Energy Technology Systems Analysis Programme

http://www.iea-etsap.org/web/Documentation.asp

Documentation for the TIMES Model

PART I

July 2016

Author: Richard Loulou

Co-authors: Gary Goldstein
Amit Kanudia
Antti Lettila
Uwe Remme

Reviewers: Evelyn Wright
George Giannakidis
Ken Noble

General Introduction to the TIMES Documentation

This documentation is composed of five Parts.

Part I provides a general description of the TIMES paradigm, with emphasis on the model's general structure and its economic significance. Part I also includes a simplified mathematical formulation of TIMES, a chapter comparing it to the MARKAL model, pointing to similarities and differences, and chapters describing new model options.

Part II constitutes a comprehensive reference manual intended for the technically minded modeler or programmer looking for an in-depth understanding of the complete model details, in particular the relationship between the input data and the model mathematics, or contemplating making changes to the model's equations. Part II includes a full description of the sets, attributes, variables, and equations of the TIMES model.

Part III describes the organization of the TIMES modeling environment and the GAMS control statements required to run the TIMES model. GAMS is a modeling language that translates a TIMES database into the Linear Programming matrix, and then submits this LP to an optimizer and generates the result files. Part III describes how the routines comprising the TIMES source code guide the model through compilation, execution, solve, and reporting; the files produced by the run process and their use; and the various switches that control the execution of the TIMES code according to the model instance, formulation options, and run options selected by the user. It also includes a section on identifying and resolving errors that may occur during the run process.

Part IV provides a step-by-step introduction to building a TIMES model in the VEDA-Front End (VEDA-FE) model management software. It first offers an orientation to the basic features of VEDA-FE, including software layout, data files and tables, and model management features. It then describes in detail twelve Demo models (available for download from the ETSAP website) that progressively introduce VEDA-TIMES principles and modeling techniques.

Part V describes the VEDA Back-End (VEDA-BE) software, which is widely used for analyzing results from TIMES models. It provides a complete guide to using VEDA-BE, including how to get started, import model results, create and view tables, create and modify user sets, and step through results in the model Reference Energy System. It also describes advanced features and provides suggestions for best practices.

PART I: TIMES CONCEPTS AND THEORY

TABLE OF CONTENTS FOR PART I

Organization of PART I

Part I comprises five divisions, each containing a number of chapters:

- Chapters 1 and 2 provide a general overview of the representation in TIMES of the Reference Energy System (RES) of a typical region or country, focusing on its basic elements, namely technologies and commodities.

- Chapters 3 to 7 describe the core TIMES model generator, i.e. the dynamic partial equilibrium version with perfect foresight: Chapter 3 discusses the economic rationale of the model, and Chapter 4 describes in more detail than chapter 3 the elastic demand feature and other economic and mathematical properties of the TIMES equilibrium. Chapter 5 presents a streamlined representation of the Linear Program used by TIMES to compute the equilibrium. Chapter 6 describes a new TIMES feature for conducting systematic sensitivity analyses. Chapter 7 describes the Climate Module of TIMES.

- Chapters 8 to 11 contain descriptions of 4 extensions or variants that, if used, depart from the assumptions of the core model in a way that alters the nature of the equilibrium: Chapter 8 covers the stochastic programming variant, which no longer assumes perfect foresight, but rather imperfect foresight; Chapter 9 describes the myopic use of TIMES, which violates the perfect foresight property and replaces it with limited foresight; Chapter 10 describes the lumpy investment variant where some decisions are discrete rather than continuous, and thus violate the convexity property; Chapter 11 describes the endogenous technology learning extension, also involving non-convex elements.

- Chapter 12 is devoted to two extensions that make TIMES into a General Equilibrium model, namely ES-MACRO and TIMES-MERGE-MACRO.

- Chapters 13 and 14 constitute appendices that may be of interest to readers at any point in their use of the rest of the text. Chapter 13 provides a brief history and comparison of TIMES and MARKAL, the modeling framework that preceded TIMES. Chapter 14 provides a short review of the theoretical foundation of Linear Programming and the interpretation of the dual solution of a linear program.

1 Introduction to the TIMES model

1.1 *A brief summary*

TIMES (an acronym for The Integrated MARKAL-EFOM[1] System) is an economic model generator for local, national, multi-regional, or global energy systems, which provides a technology-rich basis for representing energy dynamics over a multi-period time horizon. It is usually applied to the analysis of the entire energy sector, but may also be applied to study single sectors such as the electricity and district heat sector. Estimates of end-use energy service demands (e.g., car road travel; residential lighting; steam heat requirements in the paper industry; etc.) are provided by the user for each region to drive the reference scenario. In addition, the user provides estimates of the existing stocks of energy related equipment in all sectors, and the characteristics of available future technologies, as well as present and future sources of primary energy supply and their potentials.

Using these as inputs, the TIMES model aims to supply energy services at minimum global cost (more accurately at minimum loss of total surplus) by simultaneously making decisions on equipment investment and operation; primary energy supply; and energy tradefor each region. For example, if there is an increase in residential lighting energy service relative to the reference scenario (perhaps due to a decline in the cost of residential lighting, or due to a different assumption on GDP growth), either existing generation equipment must be used more intensively or new – possibly more efficient – equipment must be installed. The choice by the model of the generation equipment (type and fuel) is based on the analysis of the characteristics of alternative generation technologies, on the economics of the energy supply, and on environmental criteria. TIMES is thus a vertically integrated model of the entire extended energy system.

The scope of the model extends beyond purely energy-oriented issues, to the representation of environmental emissions, and perhaps materials, related to the energy system. In addition, the model is suited to the analysis of energy-environmental policies, which may be represented with accuracy thanks to the explicitness of the representation of technologies and fuels in all sectors.

In TIMES – like in its MARKAL forebear – the quantities and prices of the various commodities are in equilibrium, i.e. their prices and quantities in each time period are

---

[1]MARKAL (MARket ALlocation model, Fishbone et al, 1981, 1983, Berger et al. 1992) and EFOM (Van Voort et al, 1984) are two bottom-up energy models that inspired the structure of TIMES.

such that the suppliers produce exactly the quantities demanded by the consumers. This equilibrium has the property that the total economic surplus is maximized.

1.2 *Driving a TIMES model via scenarios*

The TIMES model is particularly suited to the *exploration* of possible energy futures based on contrasted *scenarios*. Given the long horizons that are usually simulated with TIMES, the scenario approach is really the only choice (whereas for the shorter term, econometric methods may provide useful projections). Scenarios, unlike forecasts, do not pre-suppose knowledge of the main drivers of the energy system. Instead, a scenario consists of a set of *coherent assumptions* about the future trajectories of these drivers, leading to a coherent organization of the system under study. A scenario builder must therefore carefully test the scenario assumptions for internal coherence, via a credible *storyline*.

In TIMES, a complete scenario consists of four types of inputs: energy service demand curves, primary resource supply curves, a policy setting, and the descriptions of a complete set of technologies. We now present a few comments on each of these four components.

1.2.1 The Demand component of a TIMES scenario

In the case of the TIMES model, demand drivers (population, GDP, households, etc.) are obtained externally, via other models or from accepted other sources. As one example, several global instances of TIMES (e.g. Loulou, 2007) use the GEM-E3[2] to generate a set of *coherent* (national and sectoral) output growth rates in the various regions. Note that GEM-E3 or GEMINI-E3 themselves use other drivers as inputs, in order to derive GDP trajectories. These drivers consist of measures of technological progress, population, degree of market competitiveness, and a few other (perhaps qualitative) assumptions. For population and household projections, TIMES instances use the same exogenous sources (IPCC, Nakicenovic 2000, Moomaw and Moreira, 2001). Other approaches may be used to derive TIMES drivers, whether via models or other means.

---

[2]European Commission, *The GEM-E3 Model*, *General Equilibrium Model for Economy, Energy and Environment,* https://ec.europa.eu/jrc/en/gem-e3/model.

For the global versions of TIMES, the main drivers are: Population, GDP, GDP per capita, number of households, and sectoral outputs. For sectoral TIMES models, the demand drivers may be different depending on the system boundaries.

Once the drivers for a TIMES model are determined and quantified the construction of the reference demand scenario requires computing a set of energy service demands over the horizon. This is done by choosing elasticities of demands to their respective drivers, in each region, using the following general formula:

$$Demand = Driver^{Elasticity}$$

As mentioned above, the demands are user provided for the reference scenario only. When the model is run for alternate scenarios (for instance for an emission constrained case, or for a set of alternate technological assumptions), it is likely that the demands will be affected. TIMES has the capability of estimating the response of the demands to the changing conditions of an alternate scenario. To do this, the model requires still another set of inputs, namely the assumed elasticities of the demands to their own prices. TIMES is then able to endogenously adjust the demands to the alternate cases without exogenous intervention. In fact, the TIMES model is driven not by demands but by *demand curves*.

To summarize: the TIMES demand scenario components consist of a set of assumptions on the drivers (GDP, population, households, outputs) and on the elasticities of the demands to the drivers and to their own prices.

1.2.2 The Supply component of a TIMES scenario

The second constituent of a scenario is a set of *supply curves* for primary energy and material resources. Multi-stepped supply curves are easily modeled in TIMES, each step representing a certain potential of the resource available at a particular cost. In some cases, the potential may be expressed as a cumulative potential over the model horizon (e.g. reserves of gas, crude oil, etc.), as a cumulative potential over the resource base (e.g. available areas for wind converters differentiated by velocities, available farmland for biocrops, roof areas for PV installations) and in others as an annual potential (e.g. maximum extraction rates, or for renewable resources the available wind, biomass, or hydro potentials). Note that the supply component also includes the identification of trading possibilities, where the amounts and prices of the traded commodities are determined endogenously (optionally within user imposed limits).

1.2.3 The Policy component of a TIMES scenario

Insofar as some policies impact on the energy system, they become an integral part of the scenario definition. For instance, a reference scenario may perfectly ignore emissions of various pollutants, while alternate policy scenarios may enforce emission restrictions, or emission taxes, etc. The detailed technological nature of TIMES allows the simulation of a wide variety of both micro measures (e.g. technology portfolios, or targeted subsidies to groups of technologies), and broader policy targets (such as general carbon tax, or permit trading system on air contaminants). A simpler example might be a nuclear policy that limits the future capacity of nuclear plants. Another example might be the imposition of fuel taxes, or of targeted capital subsidies, etc.

1.2.4 The Techno-economic component of a TIMES scenario

The fourth and last constituent of a scenario is the set of technical and economic parameters assumed for the transformation of primary resources into energy services. In TIMES, these techno-economic parameters are described in the form of *technologies* (or processes) that transform some commodities into others (fuels, materials, energy services, emissions). In TIMES, some technologies may be user imposed and others may simply be available for the model to choose from. The quality of a TIMES model rests on a rich, well developed set of technologies, both current and future, for the model to choose from. The emphasis put on the technological database is one of the main distinguishing factors of the class of Bottom-up models, to which TIMES belongs. Other classes of models will tend to emphasize other aspects of the system (e.g. interactions with the rest of the economy) and treat the technical system in a more succinct manner via aggregate production functions.

Remark: Two scenarios may differ in some or all of their components. For instance, the same demand scenario may very well lead to multiple scenarios by varying the primary resource potentials and/or technologies and/or policies, insofar as the alternative scenario assumptions do not alter the basic demand inputs (drivers and elasticities). The scenario builder must always be careful about the overall coherence of the various assumptions made on the four components of a scenario.

1.3 *Selected scenario types*

The purpose of this section is to show how certain policies may be simulated in a TIMES model. The enormous flexibility of TIMES, especially at the technology level, allows the representation of almost any policy, be it at the national, sector, or subsector level.

Policy 1: Carbon tax

A tax is levied on emissions of CO2 at point of source.

This policy is easily represented in TIMES a) making sure that all technologies that emit CO2 have an emission coefficient, and then defining a tax on these emissions (see 2.6.1.2). The policy may indicate that the tax be levied upstream for some end-use sectors (e.g. automobiles), in which case the emission coefficient is defined at the oil refinery level rather than at the level of individual car types.

Policy 2: Cap-and-trade on CO2

An upper limit on CO2 emissions is imposed at the national level (alternatively, separate upper limits are imposed at the sector level). If the model is multi-country, trade of emission permits is allowed between countries (and/or between sectors). The trade may also be upper bounded by a maximum percentage of the actual emissions, thus representing a form of the subsidiarity principle.

This type of policy is simulated by defining upper bounds on emissions, a straightforward feature in TIMES (sections2.6.1.3 and 2.6.2.3). By defining total sector emissions as a new commodity, the sector-restricted cap is just as easily implemented. The trade of national emissions makes use of the standard trade variables of TIMES (section 5.2).

Policy 3: Portfolio standard

A sector is submitted to a lower limit on its efficiency. For instance, the electricity subsector using fossil fuels must have an overall efficiency of 50%[3]. A similar example is an overall lower limit on the efficiency of light road vehicles.

[3]This standard may also be imposed on the entire electricity generation sector, in which case renewable electricity plants are assumed to have zero energy input.

This type of policy requires the definition of a new constraint that expresses that the ratio of electricity produced (via fossil fueled plants) over the amount of fuel used be more than 0.5. TIMES allows the modeller to define such new constraints via the user constraints (section 5.4.9).

Policy 4: Subsidies for some classes of technologies

The representation of this policy requires defining a capital subsidy for every new capacity of a class of technologies. This is quite straightforward in TIMES using the subsidy parameters (section 2.6.1.2.)

A more elaborate form of the subsidy might be to first levy an emission tax, and then use the proceeds of the tax to subsidize low-emitting and non-emitting technologies. Such a compound policy requires several sequential runs of TIMES, the first run establishing the proceeds of the carbon tax, followed by subsequent runs that distribute the proceeds among the targeted technologies. Several passes of these two runs may well be required in order to balance exactly the proceeds of the tax and the use of them as subsidies.

Assessing the robustness of policies

An important aspect of any policy is whether it will stay effective under various conditions. Examples of such conditions are oil prices, climate parameters, availability of certain resources, key technology costs or efficiency, etc. A policy that remains effective under a range of values for such conditions, is said to be ***robust***. In TIMES, robustness may be assessed using a variety of features, ranging from sensitivity analysis (chapter 6) to Stochastic Programming (chapter 8).

2 The basic structure of the core TIMES model

2.1 *The TIMES economy*

The TIMES energy economy is made up of producers and consumers of *commodities* such as energy carriers, materials, energy services, and emissions. By default, TIMES assumes competitive markets for all commodities, unless the modeler voluntarily imposes regulatory or other constraints on some parts of the energy system, in which case the equilibrium is (partially) regulated. The result is a supply-demand equilibrium that maximizes the *net total surplus* (i.e. the sum of producers' and consumers' surpluses) as fully discussed in chapters 3 and 4. TIMES may however depart from perfectly competitive market assumptions by the introduction of user-defined explicit constraints, such as limits to technological penetration, constraints on emissions, exogenous oil price, etc. Market imperfections can also be introduced in the form of taxes, subsidies and hurdle rates.

While computing the equilibrium, a TIMES run configures the *energy system* of a *set of regions*, over a certain *time horizon,* in such a way as to *minimize the net total cost* (or equivalently *maximize the net total surplus*) of the system, while satisfying a number of *constraints*. TIMES is run in a dynamic manner, which is to say that all investment decisions are made in each period with full knowledge of future events. The model is said to have *perfect foresight*[4] (or to be *clairvoyant*). The next subsection describes in detail the time dimension of the model.

2.2 *Time horizon*

The time horizon is divided into a user-chosen number of time-periods, each period containing a (possibly different) number of years.
In the standard version of TIMES each year in a given period is considered identical, except for the cost objective function which differentiates between payments in each year of a period. For all other quantities (capacities, commodity flows, operating levels, etc.) any model input or output related to period ***t*** applies to each of the years in that period,

[4] However, there are TIMES variants – discussed in chapters 8 to 12, that depart significantly from these assumptions

with the exception of investment variables, which are usually made only once in a period[5].

Another version of TIMES is available, in which the TIMES variables (capacities and flows) are defined at some year in the midst of each period (called milestone year), and are assumed to evolve linearly between the successive milestone years. This option emulates that of the EFOM model and is discussed in section 5.5.

The initial period is usually considered a past period, over which the model has no freedom, and for which the quantities of interest are all fixed by the user at their historical values. It is often advisable to choose an initial period consisting of a single year, in order to facilitate calibration to standard energy statistics. Calibration to the initial period is one of the more important tasks required when setting up a TIMES model. The main variables to be calibrated are: the capacities and operating levels of all technologies, as well as the extracted, exported, imported, produced, and consumed quantities for all energy carriers, and the emissions if modeled.

In TIMES, years preceding the first period also play a role. Although no explicit variables are defined for these years, data may be provided by the modeler on past investments. Note carefully that the specification of past investments influences not only the initial period's calibration, but also partially determines the model's behavior over several future periods, since the past investments provide residual capacity in several years within the modeling horizon proper.

In addition to time-periods (which may be of variable length), there are time divisions within a year, also called *time-slices,* which may be defined at will by the user (see Figure 2.1). For instance, the user may want to define seasons, portions of the day/night, and/or weekdays/weekends. Time-slices are especially important whenever the mode and cost of production of an energy carrier at different times of the year are significantly different. This is the case for instance when the some energy commodity is expensive to store so that the matching of production and consumption of that commodity is itself an issue to be resolved by the model. The production technologies for the commodity may themselves have different characteristics depending on the time of year (e.g. wind turbines or run-of-the-river hydro plants). In such cases, the matching of supply and demand requires that the activities of the technologies producing and consuming the

---

[5] There are exceptional cases when an investment must be repeated more than once in a period, namely when the period is so long that it exceeds the technical life of the investment. These cases are described in detail in section 6.2.2 of PART II.

commodity be tracked for each time slice. Examples of commodities requiring time-slicing may include electricity, district heat, natural gas, industrial steam, and hydrogen. An additional reason for defining sub yearly time slices is the requirement of an expensive infrastructure whose capacity should be sufficient to allow the peak demand for the commodity to be satisfied. Technologies that store a commodity in one time slice, at a cost, for discharge in another time slice, may also be defined and modeled.
The net result of these conditions is that the deployment in time of the various production technologies may be very different in different time slices, and furthermore that specific investment decisions will be taken to insure adequate reserve capacity at peak.

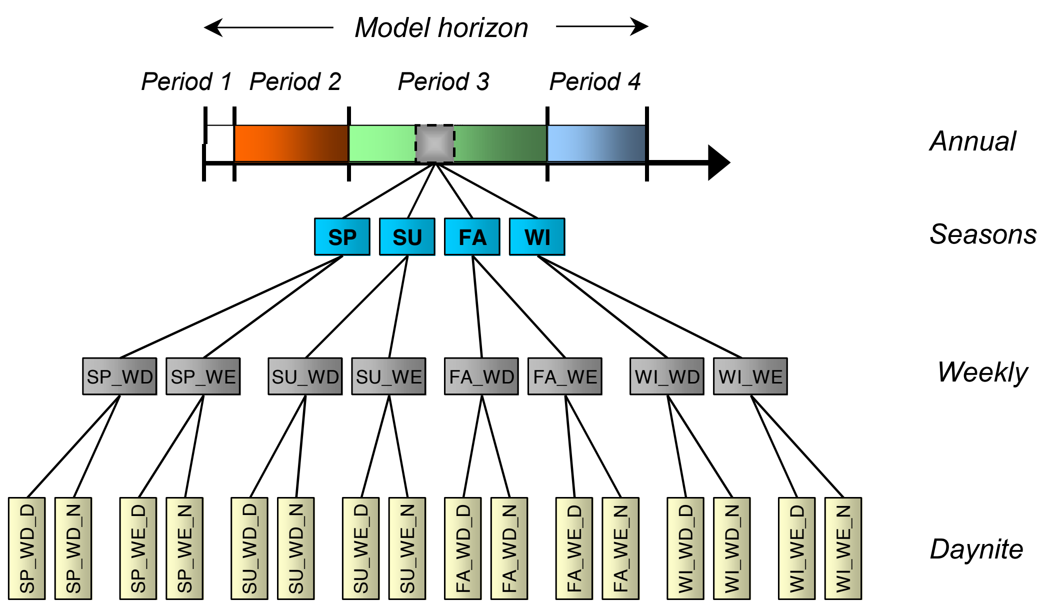


Figure 2.1: Example of a time-slice tree

2.3 *Decoupling of data and model horizon*

In TIMES, special efforts have been made to decouple the specification of data from the definition of the time periods for which a model is run. Two TIMES features facilitate this decoupling.

First, the fact that investments made in past years are recognized by TIMES makes it much easier to modify the choice of the initial and subsequent periods without major revisions of the database.

Second, the specification of process and demand input data in TIMES is made by specifying the *calendar years* when the data apply, irrespective of how the model time periods have been defined. The model then takes care of interpolating and extrapolating the data for the *periods* chosen by the modeler for a particular model run. TIMES offers a particularly rich range of interpolation/extrapolation modes adapted to each type of data and freely overridden by the user. Section 3.1.1of Part II discusses this feature.

These two features combine to make a change in the definition of periods quite easy and error-free. For instance, if a modeler decides to change the initial year from 2010 to 2015, and perhaps change the number and durations of all other periods as well, only one type of data change is needed, namely to define the investments made from 2011 to 2015 as past investments. All other data specifications need not be altered[6]. This feature represents a great simplification of the modeler's work. In particular, it enables the user to define time periods that have varying lengths, without changing the input data.

2.4 ***The components of a Reference Energy System (RES): processes, commodities, flows***

The TIMES energy economy consists of three types of entities:

- *Technologies* (also called *processes*) are representations of physical plants, vehicles, or other devices that transform some commodities into other commodities. They may be primary sources of commodities (e.g. mining processes, import processes), or transformation activities such as conversion plants that produce electricity, energy-processing plants such as refineries, or end-use demand devices such as cars and heating systems, that transform energy into a demand service;
- *Commodities* consisting of energy carriers, energy services, materials, monetary flows, and emissions. A commodity is produced by one or more processes and/or consumed by other processes; and
- *Commodity flows* are the links between processes and commodities. A flow is of the same nature as a commodity but is attached to a particular process, and represents one input or one output of that process. For instance, electricity produced by wind turbine type A at period *p*, time-slice *s*, in region *r*, is a commodity flow.

[6] However, if the horizon has been lengthened beyond the years already covered by the data, additional data for the new years at the end of the horizon must of course be provided.

2.4.1 The RES

It is helpful to picture the relationships among these various entities using a network diagram, referred to as a *Reference Energy System* (RES). In a TIMES RES, processes are represented as boxes and commodities as vertical lines. Commodity flows are represented as links between process boxes and commodity lines.

Figure 2.2 depicts a small portion of a hypothetical RES containing a single energy service demand, namely residential space heating. There are three end-use space heating technologies using the gas, electricity, and heating oil energy carriers (commodities), respectively. These energy carriers in turn are produced by other technologies, represented in the diagram by one gas plant, three electricity-generating plants (gas fired, coal fired, oil fired), and one oil refinery.

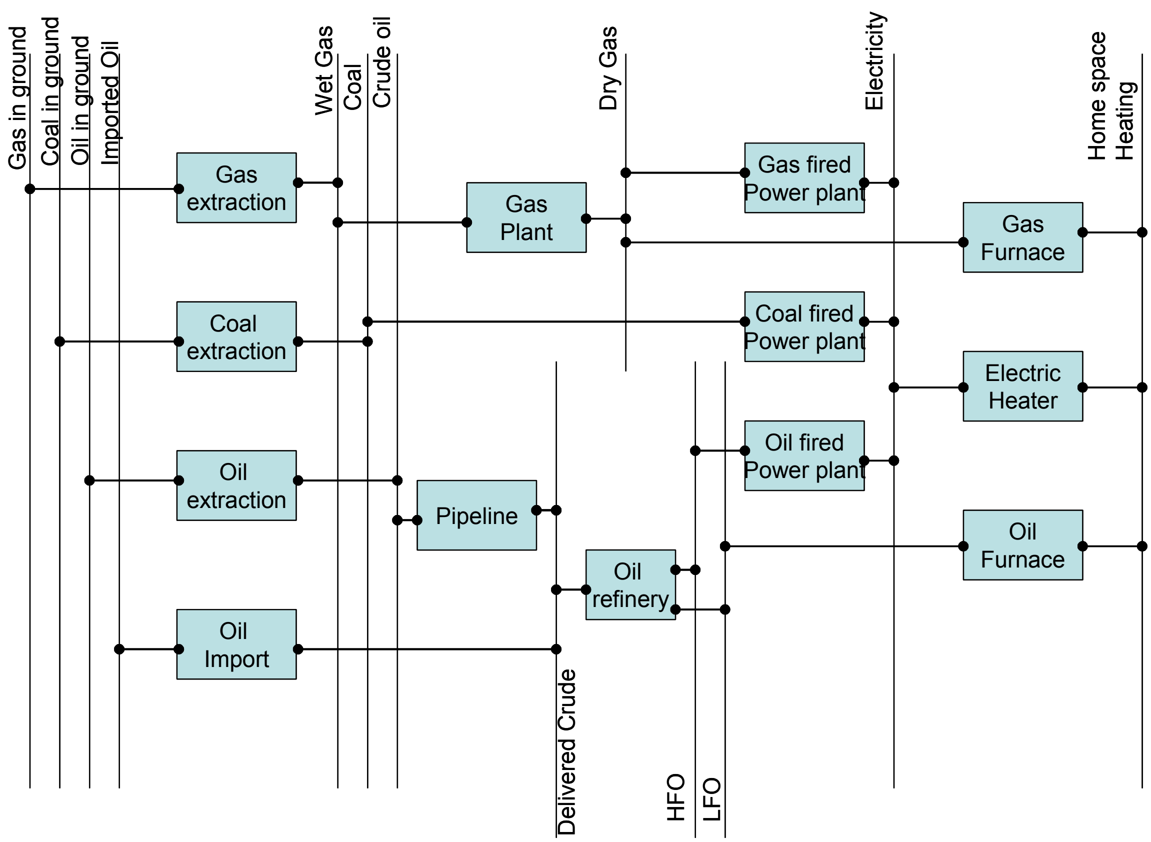


Figure 2.2. Partial view of a Reference Energy System (links are oriented left to right)

To complete the production chain on the primary energy side, the diagram also represents an extraction source for natural gas, an extraction source for coal, and two sources of crude oil (one extracted domestically and then transported by pipeline, and the other one imported). This simple RES has a total of 13 commodities and 13 processes. Note that in the RES every time a commodity enters/leaves a process (via a particular flow) its name is changed (e.g., wet gas becomes dry gas, crude becomes pipeline crude). This simple rule enables the inter-connections between the processes to be properly maintained throughout the network.

To organize the RES, and inform the modeling system of the nature of its components, the various technologies, commodities, and flows may be classified into *sets*. Each set regroups components of a similar nature. The entities belonging to a set are referred to as *members, items* or *elements* of that set. The same item may appear in multiple technology or commodity sets. While the topology of the RES can be represented by a multi-dimensional network, which maps the flow of the commodities to and from the various technologies, the set membership conveys the nature of the individual components and is often most relevant to post-processing (reporting) rather than influencing the model structure itself. However, the TIMES commodities are still classified into several *Major Groups*. There are five such groups: energy carriers, materials, energy services, emissions, and monetary flows. The use of these groups is essential in the definition of some TIMES constraints, as discussed in chapter 5.

2.4.2 Three classes of processes

We now give a brief overview of three classes of processes that need to be distinguished: Processes are *general processes, storage processe*s, and *inter-regional trading processes* (also called *inter-regional exchange processes*). The latter two classes need to be distinguished from general processes due to their special function requiring special rules and sometimes a different set of indices.

2.4.2.1 General processes

In TIMES most processes are endowed with essentially the same attributes (with the exceptions of storage and inter-regional exchange processes, see below), and unless the user decides otherwise (e.g. by providing values for some attributes and ignoring others), they have the same variables attached to them, and must obey similar constraints. Therefore, the differentiation between the various species of processes (or commodities) is made through data specification only, thus eliminating the need to define specialized membership sets, unless desired for processing results. Most of the TIMES features (e.g. sub-annual time-slice resolution, vintaging) are available for all processes and the modeler chooses the features being assigned to a particular process by specifying a corresponding indicator set (e.g. PRC_TSL, PRC_VINT).

A general process receives one or more commodity inputs (inflows) and produces one or more commodity outputs (outflows) in the same time-slice, period, and region.
As already mentioned, two classes of process do not follow these rules and deserve separate descriptions, namely *storage processes* and i*nter-regional exchange processes*.

2.4.2.2 Storage processes (class STG)

This advanced feature of TIMES allows the modeller to represent very intricate storage activities from real life energy systems. Storage processes are used to store a commodity either between periods or between time-slices in the same period. A process ***p*** is specified to be an ***inter-period storage (IPS) process*** for commodity ***c,*** or as ***general time-slice storage (TSS)***. A special case of time-slice storage is a so-called ***night-storage device (NST)*** where the commodity for charging and the one for discharging the storage are different.

An example of a night storage device is an electric heating technology that is charged during the night using electricity and produces heat during the day. Several time-slices may be specified as charging time-slices, the non-specified time-slices are assumed to be discharging time-slices. However, when the process is an end-use process that satisfies a service demand, the discharging occurs according to the load curve of the corresponding demand, and the charging is freely optimized by TIMES across time-slices. Such an exception for demand processes only exists if the demand is at the ANNUAL level. But if the demand is not ANNUAL, discharging can only occur in the non-charging time-slices.

An example of general time-slice storage is a pumped storage reservoir, where electricity is consumed during the night to store water in a reservoir, water which is then used to activate a turbine and produce electricity at a different time-slice.

An example of an inter-period storage process is a plant that accumulates organic refuse in order to produce methane some years later.

Besides the commodity being stored, other (auxiliary) commodity flows are also permitted and may be defined in relation to the stored commodity using the FLO_FUNC and/or the ACT_FLO parameters. The activity of a storage process is interpreted as the amount of the commodity being stored in the storage process. Accordingly the capacity of a storage process describes the maximum commodity amount that can be kept in storage.

2.4.2.3 *Inter-regional exchange processes (class IRE)*

Inter-regional exchange (IRE) processes are used for trading commodities between regions. Note that the name of the traded commodity is allowed to be different in both regions, depending on the chosen commodity names in both regions. There are two types of trade in TIMES, bi-lateral or multi-lateral.

Bi-lateral trade is the most detailed way to specify trade between regions. It takes place between specific pairs of regions. A pair of regions together with an exchange process and the direction of the commodity flow is first identified, and the model ensures that trade through the exchange process is balanced between these two regions (the amount is exported from region A to region B must be imported by region B from region A, possibly adjusted for transportation losses). If trade should occur only in one direction then only that direction is provided by the proper ordinal attribute. The process capacity and the process related costs (e.g. activity costs, investment costs) of the exchange process may be described individually for both regions by specifying the corresponding parameters in each region. This allows for instance the investment cost of a trade process to be shared between regions in user chosen proportions.

There are cases when it is not important to fully specify the pair of trading regions. An example is the trading of greenhouse gas (GHG) emission permits in a global market. In such cases, the *multi-lateral trade* option decreases the size of the model. Multi-lateral trade is based on the idea that a common marketplace exists for a traded commodity with several selling and several buying regions for the commodity (e.g. GHG emission permits). To model a marketplace the user must first identify (or create) one region that participates both in the production and consumption of the traded commodity. Then a single exchange process is used to link all regions with the marketplace region. Note however that some flexibility is lost when using multilateral trade. For instance, it is not possible to express transportation costs in a fully accurate manner, if such cost depends upon the precise pair of trading regions in a specific way.

2.5 ***Data-driven model structure***

It is useful to distinguish between a model's *structure* and a particular *instance* of its implementation. A model's structure exemplifies its fundamental approach for representing a problem—it does not change from one implementation to the next. All TIMES models exploit an identical underlying structure. However, because TIMES is *data*[7] *driven*, the *effective structure* of a particular instance of a model will vary according to the data inputs. This means that some of the TIMES features will not be activated if the corresponding data is not specified. For example, in a multi-region model one region may, as a matter of user data input, have undiscovered domestic oil reserves. Accordingly, TIMES automatically generates technologies and processes that account for the cost of discovery and field development. If, alternatively, user supplied data indicate that a region does not have undiscovered oil reserves no such technologies and processes would be included in the representation of that region's Reference Energy System (RES, see section 2.4). Due to this property TIMES may also be called a *model generator* that, based on the input information provided by the modeler, generates an instance of a model. In the following, if not stated otherwise, the word 'model' is used with two meanings indifferently: the instance of a TIMES model or more generally the model generator TIMES.

Thus, the structure of a TIMES model is ultimately defined by variables and equations created from the union of the underlying TIMES equations and the data input provided by the user. This information collectively defines each TIMES regional model database, and therefore the resulting mathematical representation of the RES for each region. The database itself contains both qualitative and quantitative data.

The *qualitative data* includes, for example, the list of commodities, and the list of those technologies that the modeler feels are applicable (to each region) over a specified time horizon. This information may be further classified into subgroups, for example commodities may include energy carriers (themselves split by type --e.g., fossil, nuclear, renewable, etc.), materials, emissions, energy services.

Quantitative data, in contrast, contains the technological and economic parameter assumptions specific to each technology, region, and time period. When constructing multi-region models it is often the case that a given technology is available for use in two or more regions; however, cost and performance assumptions may be quite different. The word ***attribute*** designates both qualitative and quantitative elements of the TIMES modeling system.

[7] Data in this context refers to parameter assumptions, technology characteristics, projections of energy service demands, etc. It does not refer to historical data series.

2.6 *A brief overview of the TIMES attributes*

Due to the data driven nature of TIMES (see section 2.5), all TIMES constraints are activated and defined by specifying some attributes. Attributes are attached to processes, to commodities, to flows, or to special variables that have been created to define new TIMES features. Indeed, TIMES has many new attributes that were not available in earlier versions, corresponding to powerful new features that confer additional modeling flexibility. The complete list of attributes is fully described in section 3 of PART II, and we provide below only succinct comments on the types of attribute attached to each entity of the RES or to the RES as a whole. Additional attribute definitions may also be included in the chapters describing new features or variants of the TIMES generator.

Attributes may be *cardinal* (numbers) or *ordinal* (lists, sets). For example, some ordinal attributes are defined for processes to describe subsets of flows that are then used to construct specific flow constraints as described in section 5.4. PART II, section 2 shows the complete list of TIMES sets.

The cardinal attributes are usually called *parameters*. We give below a brief idea of the main types of parameters available in the TIMES model generator.

2.6.1 Parameters attached to processes

TIMES process-oriented parameters fall into several general categories.

2.6.1.1 Technical parameters

Technical parameters include process efficiency, availability factor(s)[8], commodity consumptions per unit of activity, shares of fuels per unit activity, technical life of the process, construction lead time, dismantling lead-time and duration, amounts of the commodities consumed (respectively released) by the construction (respectively dismantling) of one unit of the process, and contribution to the peak equations. The efficiency, availability factors, and commodity inputs and outputs of a process may be defined in several flexible ways depending on the desired process flexibility, on the time-slice resolution chosen for the process and on the time-slice resolution of the

[8] There are a variety of availability factors: annual or seasonal. Each may be specified as a maximum factor (the most frequent case), an exact factor, or even a minimum factor (in order to force some minimum utilization of the capacity of some equipment, as in a backup gas turbine for instance).

commodities involved. Certain parameters are only relevant to special processes, such as storage processes or processes that implement trade between regions.

2.6.1.2 Economic and policy parameters

A second class of process parameters comprises *economic and policy parameters* that include a variety of costs attached to the investment, dismantling, maintenance, and operation of a process. The investment cost of the technology is incurred once at the time of acquisition; the fixed annual cost is incurred each year per unit of the capacity of the technology, as long as the technology is kept alive (even if it is not actively functioning); the annual variable cost is incurred per unit of the activity of the technology. In addition to costs, taxes and subsidies (on investment and/or on activity) may be defined in a very flexible manner. Other economic parameters are: the economic life of a process (the time during which the investment cost of a process is amortized, which may differ from the operational lifetime) and the process specific discount rate, also called *hurdle rate*. Both these parameters serve to calculate the annualized payments on the process investment cost, which enters the expression for the total cost of the run (section 5.2).

2.6.1.3 Bounds

Another class of parameter is used to define the right-hand-side of some constraint. Such a parameter represents a ***bound*** and its specification triggers the constraint on the quantity concerned. Most frequently used bounds are those imposed on period investment, capacity, or activity of a process. Newly defined bounds allow the user to impose limits on the annual or annualized payments at some period or set of consecutive years.

A special type of bounding consists in imposing upper or lower limits on the ***growth rate*** of technologies. The most frequently quantities thus bounded are investment, capacity and activity of a process, for which a simplified formulation has been devised.

The growth constraints belong to the class of ***dynamic bounds*** that involve multiple periods. Many other dynamic bounds may be defined by the user. ***Bounds on cumulative quantities*** are also very useful. The accumulation may be over the entire horizon or over some user defined set of consecutive years. The variables on which such bounds apply may quite varied, such as: process capacity, process investment, process activity, annual or annuity payments, etc.

All bounds may be of four types: lower (LO), upper (UP), equality (FX), or neutral (N). The latter case does not introduce any restriction on the optimization, and is used only to generate a new reporting quantity.

2.6.1.4 Other parameters

Features that were added to TIMES over the years require new parameters. For instance, the Climate Module of TIMES (chapter 7), the Lumpy Investment feature (chapter 10), and several others. These will be alluded to in the corresponding chapters of this Part I, and more completely described in section 2 and Appendices of Part II.

An advanced feature allows the user to define certain process parameters as *vintaged* (i.e. dependent upon the date of installation of new capacity). For instance, the investment cost and fuel efficiency of a specific type of automobile will depend on the model year[9].

Finally, another advanced TIMES feature renders some parameters dependent *also on the age* of the technology. For instance, the annual maintenance cost of an automobile could be defined to remain constant for say 3 years and then increase in a specified manner each year after the third year.

2.6.2 Parameters attached to commodities

This subsection concerns parameters attached to each commodity, irrespective of how the commodity is produced or consumed. The next subsection concerns commodity flows. Commodity-oriented parameters fall into the same categories as those attached to processes.

2.6.2.1 Technical parameters

Technical parameters associated with commodities include overall efficiency (for instance the overall electric grid efficiency), and the time-slices over which that commodity is to be tracked. For demand commodities, in addition, the annual projected demand and load curves (if the commodity has a sub-annual time-slice resolution) can be specified.

---

[9] Vintaging could also be introduced by defining a new technology for each vintage year, but this approach would be wasteful, as many parameters remain the same across all vintages.

2.6.2.2 Economic and policy parameters

Economic parameters include additional costs, taxes, and subsidies on the overall or net production of a commodity. These cost elements are then added to all other (implicit) costs of that commodity. In the case of a demand service, additional parameters define the demand curve (i.e. the relationship between the quantity of demand and its price). These parameters are: the demand's own-price elasticity, the total allowed range of variation of the demand value, and the number of steps to use for the discrete approximation of the curve.

Policy based parameters include bounds (at each period or cumulative over user defined years) on the gross or net production of a commodity, or on the imports or exports of a commodity by a region.

2.6.2.3 Bounds

In TIMES the net or the total production of each commodity may be explicitly represented by a variable, if needed for imposing a bound or a tax. A similar variety of bounding parameters exists for commodities as for processes.

2.6.3 Parameters attached to commodity flows

A *commodity flow* (more simply, a *flow*) is an amount of a given commodity produced or consumed by a given process. Some processes have several flows entering or leaving them, perhaps of different types (fuels, materials, demands, or emissions). In TIMES, each flow has a variable attached to it, as well as several attributes (parameters or sets). Flow related parameters confer enormous flexibility for modeling a large spectrum of conditions.

2.6.3.1 Technical parameters

Technical parameters, along with some set attributes, permit full control over the maximum and/or minimum share a given input or output flow may take within the same commodity group. For instance, a flexible turbine may accept oil and/or gas as input, and the modeler may use a parameter to limit the share of oil to, say, at most 40% of the total fuel input. Other parameters and sets define the amount of certain outflows in relation to certain inflows (e.g., efficiency, emission rate by fuel). For instance, in an oil refinery a parameter may be used to set the total amount of refined products equal to 92% of the total amount of crude oils (s) entering the refinery, or to calculate certain emissions as a fixed proportion of the amount of oil consumed. If a flow has a sub-annual time-slice resolution, a load curve can be specified for the flow. It is possible to define not only load curves for a flow, but also bounds on the share of a flow in a specific time-slice relative to the annual flow, e.g. the flow in the time-slice "Winter-Day" has to be at least 10 % of the total annual flow. Refer to section 5.4 describing TIMES constraints for details. Cumulative bounds on a process flow are also allowed.

2.6.3.2 Economic and policy parameters

Economic or policy parameters include delivery and other variable costs, taxes and subsidies attached to an individual process flow.

2.6.3.3 Bounds

Bounds may be defined for flows in similar variety that exists for commodities.

2.6.4 Parameters attached to the entire RES

These parameters include currency conversion factors (in a multi-regional model), region-specific time-slice definitions, a region-specific general discount rate, and reference year for calculating the discounted total cost (objective function). In addition, certain switches are needed to control the activation of the data interpolation procedure as well as special model features to be used. The complete set of switches is described in Part III.

2.7 ***Process and commodity classification***

Although TIMES does not explicitly differentiate processes or commodities that belong to different portions of the RES (with the notable exceptions of storage and trading processes), there are three ways in which some differentiation does occur.

First, TIMES requires the definition of Primary Commodity Groups (*pcg*), i.e. subsets of commodities *of the same nature* entering or leaving a process. TIMES utilizes the pcg to define the activity of the process, and also its capacity. For instance, the *pcg* of an oil refinery is defined as the set of energy forms produced by the plant; and the activity of the refinery is thus simply the sum of all its energy outputs (excluding any outputs that are non energy).

Besides establishing the process activity and capacity, these groups are convenient aids for defining certain complex quantities related to process flows, as discussed in chapter 5 and in PART II, section 2.1.

Even though TIMES *does not require* that the user provide many set memberships, the TIMES reporting step does pass some set declarations to the VEDA-BE result-processing system[10]to facilitate construction of results analysis tables. These include process subsets to distinguish demand devices, energy processes, material processes (by weight or volume), refineries, electric production plants, coupled heat and power plants, heating plants, storage technologies and distribution (link) technologies; and commodity subsets for energy, useful energy demands (split into six aggregate sub-sectors), environmental indicators, currencies, and materials.

Besides the definition of *pcg*'s and that of VEDA reporting sets, there is a third instance of commodity or process differentiation which is not embedded in TIMES, but rests entirely on the modeler. A modeler may well want to choose process and commodity names in a judicious manner so as to more easily identify them when browsing through the input database or when examining results. As an example, the TIAM-World multi-regional TIMES model (Loulou, 2007) adopts a naming convention whereby the first three characters of a commodity's name denote the sector and the next three the fuel (e.g., light fuel oil used in the residential sector is denoted RESLFO). Similarly, process names are chosen so as to identify the sub-sector or end-use (first three characters), the main fuel used (next three), and the specific technology (last four). For instance, a standard (0001) residential water heater (RWH) using electricity (ELC) is named RWHELC0001. Naming conventions may thus play a critical role in allowing the easy identification of an element's position in the RES and thus facilitate the analysis and reporting of results.

Similarly, energy services may be labeled so that they are more easily recognized. For instance, the first letter may indicate the broad sector (e.g. 'T' for transport) and the second letter designate any homogenous sub-sectors (e.g. 'R' for road transport), the third character being free.

In the same fashion, fuels, materials, and emissions may be identified so as to immediately designate the sector and sub-sector where they are produced or consumed. To achieve this, some fuels have to change names when they change sectors. This is accomplished via processes whose primary role is to change the name of a fuel. In

[10]See Appendix A for the VEDA-FE, VEDA-BE, and ANSWER modeling and analysis systems, used to maintain and manage TIMES databases, conduct model runs, and organize results.

addition, such a process may serve as a bearer of sector wide parameters such as distribution cost, price markup, tax, that are specific to that sector and fuel. For instance, a tax may be levied on industrial distillate use but not on agricultural distillate use, even though the two commodities are physically identical.

3 Economic rationale of the TIMES modeling approach

This chapter provides a detailed economic interpretation of TIMES and other partial equilibrium models based on maximizing total surplus. Partial equilibrium models have one common feature – they simultaneously configure the production and consumption of commodities (i.e. fuels, materials, and energy services) and their prices. The price of producing a commodity affects the demand for that commodity, while at the same time the demand affects the commodity's price. A market is said to have reached an equilibrium at prices p^* and quantities q^* when no consumer wishes to purchase less than q^* and no producer wishes to produce more than q^* at price p^*. Both p^* and q^* are vectors whose dimension is equal to the number of different commodities being modeled. As will be explained below, when all markets are in equilibrium the total economic surplus is maximized.

The concept of total surplus maximization extends the direct cost minimization approach upon which earlier bottom-up energy system models were based. These simpler models had fixed energy service demands, and thus were limited to minimizing the cost of supplying these demands. In contrast, the TIMES demands for energy services are themselves elastic to their own prices, thus allowing the model to compute a *bona fide* supply-demand equilibrium. This feature is a fundamental step toward capturing the main feedback from the economy to the energy system.

Section 3.1 provides a brief review of different types of energy models. Section 3.2 discusses the economic rationale of the TIMES model with emphasis on the features that distinguish TIMES from other bottom-up models (such as the early incarnations of MARKAL, see Fishbone and Abilock, 1981 and Berger et al., 1992, though MARKAL has since been extended beyond these early versions). Section 3.3 describes the details of how price elastic demands are modeled in TIMES, and section 3.4 provides additional discussion of the economic properties of the model.

3.1. A brief classification of energy models

Many energy models are in current use around the world, each designed to emphasize a particular facet of interest. Differences include: economic rationale, level of disaggregation of the variables, time horizon over which decisions are made (which is closely related to the type of decisions, i.e., only operational planning or also investment decisions), and geographic scope. One of the most significant differentiating features among energy models is the degree of detail with which commodities and technologies are represented, which will guide our classification of models into two major classes, as explained in the following very streamlined classification.

3.1.1 'Top-down' models

At one end of the spectrum are aggregated *General Equilibrium* (GE) models. In these each sector is represented by a production function designed to simulate the potential substitutions between the main factors of production (also highly aggregated into a few variables such as: energy, capital, and labor) in the production of each sector's output. In this model category are found a number of models of national or global energy systems. These models are usually called "top-down", because they represent an entire economy via a relatively small number of aggregate variables and equations. In these models, production function parameters are calculated for each sector such that inputs and outputs reproduce a single base historical year.[11] In policy runs, the mix of inputs[12] required to produce one unit of a sector's output is allowed to vary according to user-selected elasticities of substitution. Sectoral production functions most typically have the following general form:

$$\boldsymbol{X_s = A_0\left(B_K \cdot K_s^{\rho} + B_L \cdot L_S^{\rho} + B_E \cdot E_S^{\rho}\right)^{1/\rho}} \tag{3-1}$$

where $\boldsymbol{X_S}$ is the output of sector *S,*
$\boldsymbol{K_S}$*,* $\boldsymbol{L_S}$*, and* $\boldsymbol{E_S}$ are the inputs of capital, labor and energy needed to produce one unit of output in sector *S,*
$\boldsymbol{\rho}$ is the elasticity of substitution parameter,
$\boldsymbol{A_0}$ and the $\boldsymbol{B}$'s are scaling coefficients.

The choice of ρ determines the ease or difficulty with which one production factor may be substituted for another: the smaller ρ is (but still greater than or equal to 1), the easier it is to substitute the factors to produce the same amount of output from sector *S*. Also note that the degree of factor substitutability does not vary among the factors of production — the ease with which capital can be substituted for labor is equal to the ease

[11] These models assume that the relationships (as defined by the form of the production functions as well as the calculated parameters) between sector level inputs and outputs are in equilibrium in the base year.
[12] Most models use inputs such as labor, energy, and capital, but other input factors may conceivably be added, such as arable land, water, or even technical know-how. Similarly, labor may be further subdivided into several categories.

with which capital can be substituted for energy, while maintaining the same level of output. GE models may also use alternate forms of production function (3-1), but retain the basic idea of an explicit substitutability of production factors.

3.1.2 'Bottom-up' models

At the other end of the spectrum are the very detailed, *technology explicit* models that focus primarily on the energy sector of an economy. In these models, each important energy-using technology is identified by a detailed description of its inputs, outputs, unit costs, and several other technical and economic characteristics. In these so-called 'bottom-up' models, a sector is constituted by a (usually large) number of logically arranged technologies, linked together by their inputs and outputs (*commodities*, which may be energy forms or carriers, materials, emissions and/or demand services). Some bottom-up models compute a partial equilibrium via maximization of the total net (consumer and producer) surplus, while others simulate other types of behavior by economic agents, as will be discussed below. In bottom-up models, one unit of sectoral output (e.g., a billion vehicle kilometers, one billion tonnes transported by heavy trucks or one petajoule of residential cooling service) is produced using a mix of individual technologies' outputs. Thus the production function of a sector is *implicitly* constructed, rather than explicitly specified as in more aggregated models. Such implicit production functions may be quite complex, depending on the complexity of the reference energy system of each sector (sub-RES).

3.1.3 Hybrid approaches

While the above dichotomy applied fairly well to earlier models, these distinctions now tend to be somewhat blurred by advances in both categories of model. In the case of aggregate top-down models, several general equilibrium models now include a fair amount of fuel and technology disaggregation in the key energy producing sectors (for instance: electricity production, oil and gas supply). This is the case with MERGE[13] and SGM[14], among others.

In the other direction, the more advanced bottom-up models are 'reaching up' to capture some of the effects of the entire economy on the energy system. The TIMES model has

---

[13] Model for Evaluating Regional and Global Effects (Manne et al., 1995)
[14] Second Generation Model (Edmonds et al., 1991)

end-use demands (including demands for industrial output) that are sensitive to their own prices, and thus captures the impact of rising energy prices on economic output and *vice versa*. Recent incarnations of technology-rich models (including TIMES) are multi-regional, and thus are able to consider the impacts of energy-related decisions on trade. It is worth noting that while the multi-regional top-down models have always represented trade, they have done so with a very limited set of traded commodities – typically one or two, whereas there may be quite a number of traded energy forms and materials in multi-regional bottom-up models.

MARKAL-MACRO (Manne and Wene, 1992) and TIMES-MACRO (Kypreos and Lehtila, 2013) are hybrid models combining the technological detail of MARKAL with a succinct representation of the macro-economy consisting of a single producing sector in a single region. Because of its succinct single-sector production function, MARKAL-MACRO is able to compute a general equilibrium in a single optimization step. More recently, TIMES_MACRO-MSA (section 12.2) is based on the computation of a multi-regional global equilibrium, but requires an iterative process to do so. MESSAGE (Messner and Strubegger, 1995) links a bottom-up model based on the EFOM paradigm with a macro module, and computes a global, multi-regional equilibrium iteratively. The NEMS (US EIA, 2000) model is another example of a full linkage between several technology rich modules of the various energy subsectors and a set of macro-economic equations, and requires iterative resolution methods.

In spite of these advances in both classes of models, there remain important differences. Specifically:

- Top-down models encompass macroeconomic variables beyond the energy sector proper, such as wages, consumption, and interest rates, and
- Bottom-up models have a rich representation of the variety of technologies (existing and/or future) available to meet energy needs, and, they often have the capability to track a much wider variety of traded commodities. They are also more adapted to the representation of micro policies targeting specific technologies or commodities.

The top-down vs. bottom-up approach is not the only relevant difference among energy models. Among top-down models, the so-called Computable General Equilibrium models (CGE) described above differ markedly from the *macro econometric models*. The latter do not compute equilibrium solutions, but rather simulate the flows of capital and other monetized quantities between sectors (see, e.g., Meade, 1996 on the LIFT model). They use econometrically derived input-output coefficients to compute the impacts of these flows on the main sectoral indicators, including economic output (GDP) and other variables (labor, investments). The sector variables are then aggregated into national indicators of consumption, interest rate, GDP, labor, and wages.

Among technology explicit models also, two main classes are usually distinguished: the first class is that of the partial equilibrium models such as MARKAL, MESSAGE, and TIMES, that use optimization techniques to compute a least cost (or maximum surplus) path for the energy system. The second class is that of *simulation* models, where the emphasis is on representing a system not governed purely by financial costs and profits. In these simulation models (e.g., CIMS, Jaccard et al. 2003), investment decisions taken by a representative agent (firm or consumer) are only partially based on profit maximization, and technologies may capture a share of the market even though their life-cycle cost may be higher than that of other technologies. Simulation models use market-sharing formulas that preclude the easy computation of equilibrium – at least not in a single pass. The SAGE (US EIA, 2002) incarnation of the MARKAL model possesses a market sharing mechanism that allows it to reproduce certain behavioral characteristics of observed markets.

3.2 *The core TIMES paradigm*

In the rest of this chapter, we present the properties of the **core TIMES** paradigm. As will be seen in chapters8 to 12, some of these properties are not applicable to several important TIMES variants. The reader should keep this caveat in mind when contemplating the use of some features that are described in these 5 chapters.

Since certain portions of this and the next sections require an understanding of the concepts and terminology of Linear Programming, the reader requiring a brush-up on this topic may first read Appendix B, and then, if needed, some standard textbook on LP, such as Hillier and Lieberman (2009), Chvàtal (1983), or Schrijver (1986). The application of Linear Programming to microeconomic theory is covered in two historically important references, Gale (1960 and 11th edition 1989), and in Dorfman, Samuelson, and Solow (1958, and 1987 reprint).

A brief description of the core TIMES model generator would express that it is:

- *Technologically explicit, integrated*;
- *Multi-regional*; and

- *Partial equilibrium* (with *price elastic* demands for energy services) in *competitive markets* with *perfect foresight*. It will be seen that such an equilibrium entails *marginal value pricing* of all commodities.

We now proceed to flesh out each of these properties.

3.2.1 A technologically explicit integrated model

As already presented in chapter 2 (and described in much more detail in Part II, section 3), each technology is described in TIMES by a number of technical and economic parameters. Thus each technology is explicitly identified (given a unique name) and distinguished from all others in the model. A mature TIMES model may include several thousand technologies in all sectors of the energy system (energy procurement, conversion, processing, transmission, and end-uses) in each region. Thus TIMES is not only technologically explicit, it is technology rich and it is integrated as well. Furthermore, the number of technologies and their relative topology may be changed at will, purely via data input specification, without the user ever having to modify the model's equations. The model is thus to a large extent *data driven*.

3.2.2 Multi-regional

Some existing TIMES models comprise several dozen regional modules, or more. The number of regions in a model is limited only by the difficulty of solving LP's of very large size. The individual regional modules are linked by energy and material trading variables, and by emission permit trading variables, if desired. The linking variables transform the set of regional modules into a *single* multi-regional (possibly global) energy model, where actions taken in one region may affect all other regions. This feature is essential when global as well as regional energy and emission policies are being simulated. Thus a multi-regional TIMES model is geographically integrated.

3.2.3 Partial equilibrium

The core version of TIMES computes a partial equilibrium on energy markets. This means that the model computes both the *flows* of energy forms and materials as well as their *prices*, in such a way that, at the prices computed by the model, the suppliers of energy produce exactly the amounts that the consumers are willing to buy. This equilibrium feature is present at every stage of the energy system: primary energy forms, secondary energy forms, and energy services[15]. A supply-demand equilibrium model has as its economic rationale the maximization of the total surplus, defined as the sum of all suppliers' and consumers' surpluses. The mathematical method used to maximize the surplus must be adapted to the particular mathematical properties of the model. In TIMES, these properties are as follows:

- Outputs of a technology are linear functions of its inputs (subsection 3.2.3.1)[16];
- Total economic surplus is maximized over the entire horizon (3.2.3.2); and
- Energy markets are competitive, with perfect foresight (3.2.3.3)[17].

As a result of these assumptions the following additional properties hold:

- The market price of each commodity is equal to its marginal value in the overall system (3.2.4); and
- Each economic agent maximizes its own profit or utility (3.2.5).

3.2.3.1 Linearity

A linear input-to-output relationship first means that each technology represented may be implemented at any capacity, from zero to some upper limit, without economies or dis-economies of scale. In a real economy, a given technology is usually available in discrete sizes, rather than on a continuum. In particular, for some real life technologies, there may be a minimum size below which the technology may not be implemented (or else at a prohibitive cost), as for instance a nuclear power plant, or a hydroelectric project. In such cases, because TIMES assumes that all technologies may be implemented in any size, it may happen that the model's solution shows some technology's capacity at an unrealistically small size. It should however be noted that in most applications, such a situation is relatively infrequent and often innocuous, since the scope of application is at the country or region's level, and thus large enough so that small capacities are unlikely to occur.

---

[15]It has been argued, based on strong experimental evidence, that the change in demands for energy services indeed captures the main economic impact of energy system policies on the economy at large (Loulou and Kanudia, 2000)

[16] This property does not hold in three TIMES extensions presented in Chapters 10-12.

[17] These two properties do not hold in the time-stepped extension of TIMES (chapter 9) and in Stochastic TIMES (Chapter 8.)

On the other hand, there may be situations where plant size matters, for instance when the region being modeled is very small. In such cases, it is possible to enforce a rule by which certain capacities are allowed only in multiples of a given size (e.g., build or not a gas pipeline), by introducing *integer variables*. This option, referred to as lumpy investment (LI), is available in TIMES and is discussed in chapter 10. This approach should, however, be used sparingly because it may greatly increase solution time.
It is the linearity property that allows the TIMES equilibrium to be computed using Linear Programming techniques. In the case where economies of scale or some other non-convex relationship is important to the problem being investigated, the optimization program would no longer be linear or even convex. We shall examine such cases in chapters 9 to 12.

We must now mention a common misconception regarding linearity: the fact that TIMES equations are linear *does not mean that production functions behave in a linear fashion; far from it*. Indeed, the TIMES production functions are usually highly non-linear (although convex), consisting of a stepped sequence of linear functions. As a simple example, a supply of some resource is almost always represented as a sequence of segments, each with rising (but constant within an interval) unit cost. The modeler defines the 'width' of each interval so that the resulting supply curve may simulate any non-linear convex function. In brief, diseconomies of scale are easily represented in linear models.

3.2.3.2 Maximization of total surplus: Price equals marginal value

The *total surplus* of an economy is the sum of the suppliers' and the consumers' surpluses. The term *supplier* designates any economic agent that produces (and/or sells) one or more commodities i.e., in TIMES, an energy form, a material, an emission permit, and/or an energy service. A *consumer* is a buyer of one or more commodities. In TIMES, the suppliers of a commodity are technologies that procure a given commodity, and the consumers of a commodity are technologies or service segments that consume a given commodity. Some (indeed most) technologies are both suppliers and consumers. Therefore, for each commodity the RES defines a complex set of suppliers and consumers.

It is customary in microeconomics to represent the set of suppliers of a commodity by their *inverse production function*, that plots the marginal production cost of the commodity (vertical axis) as a function of the quantity supplied (horizontal axis). In TIMES, as in other linear optimization models, the supply curve of a commodity, with the exception of end-use demands, is entirely determined endogenously by the model. It is a standard result of Linear Programming theory that the inverse supply function is step-wise constant and increasing in each factor (see Figures 3.1 and 3.3 for the case of a single commodity[18]). Each horizontal step of the inverse supply function indicates that the commodity is produced by a certain technology or set of technologies in a strictly linear fashion. As the quantity produced increases, one or more resources in the mix (either a technological potential or some resource's availability) is exhausted, and therefore the system must start using a different (more expensive) technology or set of technologies in order to produce additional units of the commodity, albeit at higher unit cost. Thus, each change in production mix generates one step of the staircase production function with a value higher than the preceding step. The width of any particular step depends upon the technological potential and/or resource availability associated with the set of technologies represented by that step.

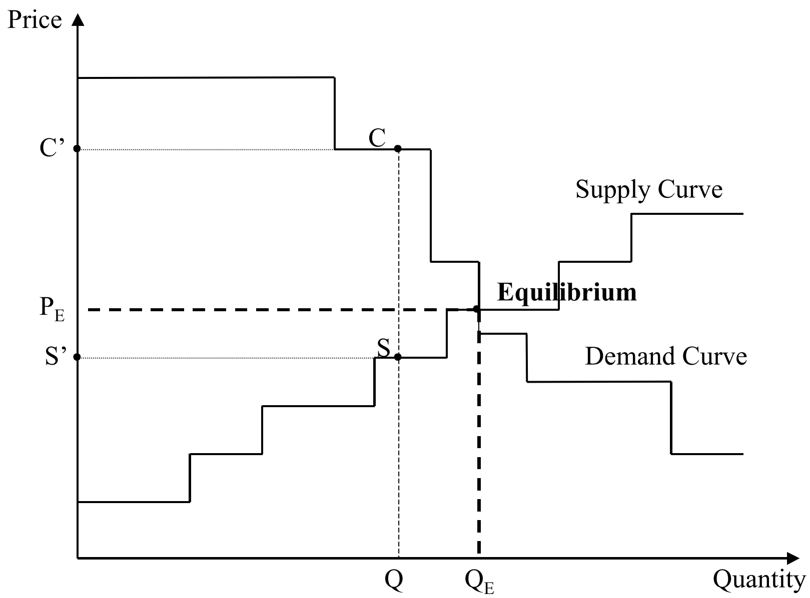


Figure 3.1. Equilibrium in the case of an energy form: the model implicitly constructs both the supply and the demand curves (note that the equilibrium is multiple in this configuration)

In a similar manner, each TIMES model instance defines a series of inverse demand functions. In the case of demands, two cases are distinguished. First, if the commodity in question is an energy carrier whose production and consumption are endogenous to the model, then its demand function is *implicitly* constructed within TIMES, and is a step-

[18] This is so because in Linear Programming the shadow price of a constraint remains constant over a certain interval, and then changes abruptly, giving rise to a stepwise constant functional shape.

wise constant, decreasing function of the quantity demanded, as illustrated in Figure 3.1 for a single commodity. If on the other hand the commodity is a demand for an energy service, then its demand curve is *defined by the user* via the specification of the own-price elasticity of that demand, and the curve is in this instance a smoothly decreasing curve as illustrated in Figure 3.2[19]. In both cases, the supply-demand equilibrium is at the intersection of the supply function and the demand function, and corresponds to an equilibrium quantity Q_E and an equilibrium price P_E[20]. At price P_E, suppliers are willing to supply the quantity Q_E and consumers are willing to buy exactly that same quantity Q_E. Of course, the TIMES equilibrium concerns a large number of commodities simultaneously, and thus the equilibrium is a multi-dimensional analog of the above, where Q_E and P_E are now vectors rather than scalars.

As already mentioned, the demand curves of most TIMES commodities (i.e. energy carriers, materials, emission permits) are implicitly constructed endogenously as an integral part of the solution of the LP. For each commodity that is an energy service, the user *explicitly* defines the demand *function* by specifying its own price elasticity. In TIMES, each energy service demand is assumed to have a constant own price elasticity function of the form (see Figure 3.2):

$$DM/DM_0 = (P/P_0)^E \qquad (3\text{-}2)$$

Where $\{DM_0, P_0\}$ is a reference pair of demand and price values for that energy service over the forecast horizon, and *E* is the (negative) own price elasticity of that energy service demand, as specified by the user (note that although not obvious from the notation, this price elasticity may vary over time). The pair $\{DM_0, P_0\}$ is obtained by solving TIMES for a reference scenario. More precisely, $DM_0$is the demand projection estimated by the user in the reference scenario (usually based upon explicitly defined relationships to economic and demographic drivers), and P_0 is the shadow price of that energy service demand in the dual solution of the reference case scenario. The precise manner in which the demand functions are discretized and incorporated in the TIMES objective function is explained in chapter 4.

Using Figure 3.1 as an example, the definition of the suppliers' surplus corresponding to a certain point S on the inverse supply curve is the difference between the total revenue

---

[19] This smooth curve will be discretized later for computational purposes, and thus become a staircase function, as described in section 4.2

[20] As may be seen in figure 3.1, the equilibrium is not necessarily unique. In the case shown, any point on the vertical segment containing the equilibrium is also an equilibrium, with the same quantity Q_E but a different price. In other situations, the multiple equilibria may have a single price but multiple quantities.

and the total cost of supplying a commodity, i.e. the gross profit. In Figure 3.1, the surplus is thus the area between the horizontal segment SS' and the inverse supply curve. Similarly, the consumers' surplus for a point C on the inverse demand curve, is defined as the area between line segment CC' and the inverse demand curve. This area is a consumer's analog to a producer's profit; more precisely it is the cumulative opportunity gain of all consumers who purchase the commodity at a price lower than the price they would have been willing to pay. Thus, for a given quantity Q, the total surplus (suppliers' plus consumers') is simply the area between the two inverse curves situated at the left of Q. It should be clear from Figure 3.1 that the total surplus is maximized when Q is exactly equal to the equilibrium quantity Q_E. Therefore, we may state (in the single commodity case) the following Equivalence Principle:

"The supply-demand equilibrium is reached when the total surplus is maximized."

This is a remarkably useful result, as it leads to a method for computing the equilibrium, as will be see in much detail in Chapter 4.

In the multi-dimensional case, the proof of the above statement is less obvious, and requires a certain qualifying property (called the integrability property) to hold (Samuelson, 1952, Takayama and Judge, 1972). One sufficient condition for the integrability property to be satisfied is realized when the cross-price elasticities of any two energy forms are equal, viz.

$$\partial \boldsymbol{P_j} / \partial \boldsymbol{Q_i} = \partial \boldsymbol{P_i} / \partial \boldsymbol{Q_j} \quad \text{for all } \boldsymbol{i}, \boldsymbol{j}$$

In the case of commodities that are end-use energy services, these conditions are trivially satisfied in TIMES because we have assumed zero cross price elasticities. In the case of an endogenous energy carrier, where the demand curve is implicitly derived, it is also easy to show that the integrability property is always satisfied[21]. Thus the equivalence principle is valid in all cases.

[21] This results from the fact that in TIMES each price $\boldsymbol{P_i}$ is the shadow price of a balance constraint (see section 5.4.4), and may thus be (loosely) expressed as the derivative of the objective function $\boldsymbol{F}$ with respect to the right-hand-side of a balance constraint, i.e. $\partial \boldsymbol{F} / \partial \boldsymbol{Q_i}$. When that price is further differentiated with respect to another quantity $\boldsymbol{Q_j}$, one gets $\partial^2 \boldsymbol{F} / \partial \boldsymbol{Q_i} \bullet \partial \boldsymbol{Q_j}$, which, under mild conditions is always equal to $\partial^2 \boldsymbol{F} / \partial \boldsymbol{Q_j} \bullet \partial \boldsymbol{Q_i}$, as desired.

In summary, the equivalence principle guarantees that the TIMES supply-demand equilibrium maximizes total surplus. The total surplus concept has long been a mainstay of social welfare economics because it takes into account both the surpluses of consumers and of producers.[22]

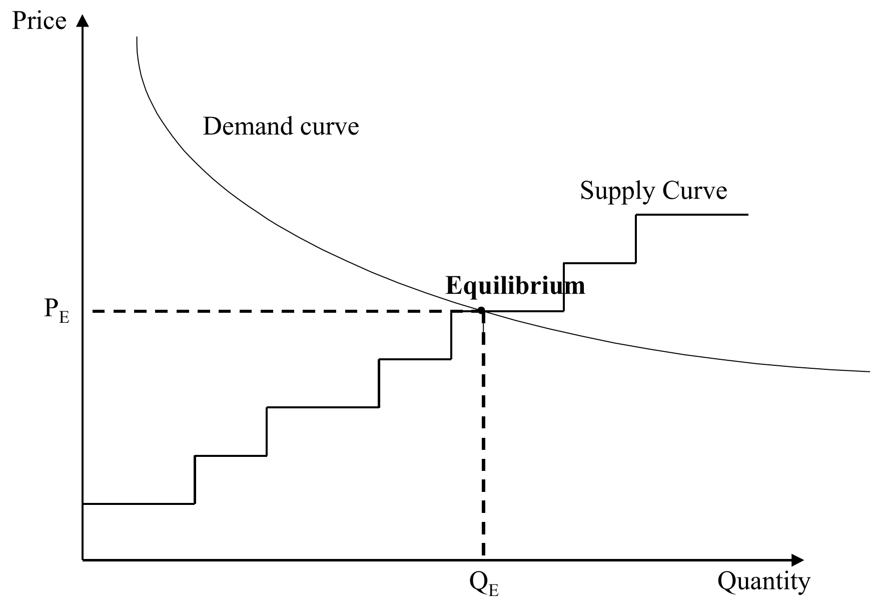


Figure 3.2. Equilibrium in the case of an energy service: the user explicitly provides the demand curve, usually using a simple functional form (see text for details)

Remark: In older versions of MARKAL, and in several other least-cost bottom-up models, energy service demands are exogenously specified by the modeler, and only the cost of supplying these energy services is minimized. Such a case is illustrated in Figure 3.3 where the "inverse demand curve" is a vertical line. The objective of such models was simply the minimization of the total cost of meeting exogenously specified levels of energy service.

---

[22] See e.g. Samuelson and Nordhaus (1977)

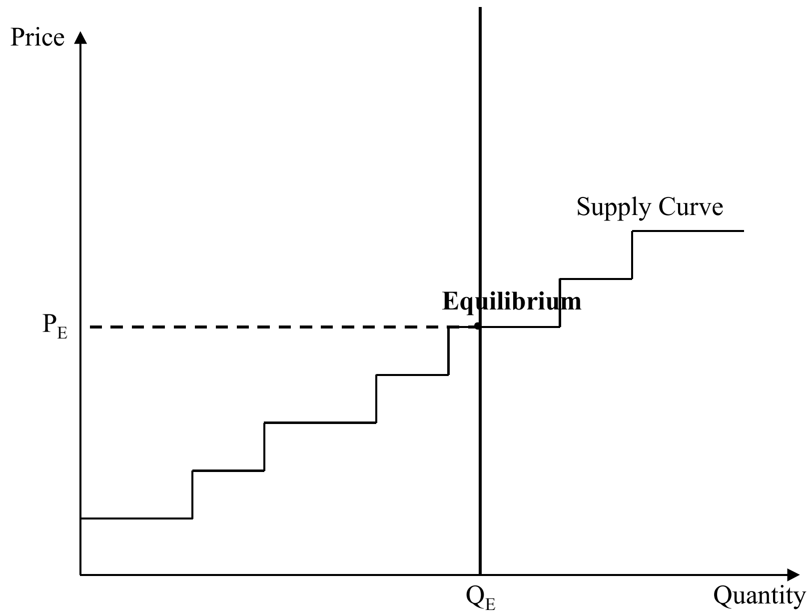


Figure 3.3. Equilibrium when an energy service demand is fixed

3.2.3.3 Competitive energy markets with perfect foresight

Competitive energy markets are characterized by perfect information and atomic economic agents, which together preclude any of them from exercising market power. That is, neither the level at which any individual producer supplies, nor the level any individual consumer acquires, affects the equilibrium market price (because there are many other buyers and sellers to replace them). It is a standard result of microeconomic theory that the assumption of competitive markets entails that the market price of a commodity is equal to its marginal value in the economy (Samuelson, 1952). This is of course also verified in the TIMES economy, as discussed in the next subsection.

Of course, real world energy markets are not always competitive. For instance, an electric utility company may be a (regulated) monopoly within an entire country, or a cartel of oil producing countries may have market power on oil markets. There are ways around these so-called “market imperfections”. For instance, concerning the monopolistic utility, a socially desirable approach would be to first use the assumption of marginal cost pricing, so as to determine a socially optimal plan for the monopoly, and then to have the regulatory agency enforce such a plan, including the principle of marginal cost pricing. The case of the oil producers’ cartel is less simple, since there is no global regulatory agency to ensure that oil producers act in a socially optimal fashion. There are however ways to use equilibrium models such as TIMES in order to faithfully represent the market power of certain economic agents, as exemplified in (Loulou et al., 2007).

In the core version of TIMES, the perfect information assumption extends to the entire planning horizon, so that each agent has perfect foresight, i.e. complete knowledge of the market's parameters, present and future. Hence, the equilibrium is computed by maximizing total surplus in one pass for the entire set of periods. Such a farsighted equilibrium is also called an *inter-temporal dynamic equilibrium*.

Note that there are at least two ways in which the perfect foresight assumption may be voided: in one variant, agents are assumed to have foresight over a limited portion of the horizon, say one or a few periods. Such an assumption of limited foresight is embodied in the TIMES feature discussed in chapter 9, as well as in the SAGE variant of MARKAL (US EIA, 2002). In another variant, foresight is assumed to be imperfect, meaning that agents may only *probabilistically* know certain key future events. This assumption is at the basis of the TIMES Stochastic Programming option, discussed in chapter 8.

3.2.4 Marginal value pricing

We have seen in the preceding subsections that the TIMES equilibrium occurs at the intersection of the inverse supply and inverse demand curves. It follows that the equilibrium prices are equal to the marginal system values of the various commodities. From a different angle, the duality theory of Linear Programming (chapter 14) indicates that for each constraint of the TIMES linear program there is a *dual variable.* This dual variable (when an optimal solution is reached) is also called the constraint's *shadow price*[23]*,* and is equal to the marginal change of the objective function per unit increase of the constraint's right-hand-side. For instance, the shadow price of the balance constraint of a commodity (whether it be an energy form, material, a service demand, or an emission) represents the competitive market price of the commodity.

---

[23] The term *shadow price* is often used in the mathematical economics literature, whenever the price is derived from the marginal value of a commodity. The qualifier 'shadow' is used to distinguish the competitive market price from the price observed in the real world, which may be different, as is the case in regulated industries or in sectors where either consumers or producers exercise market power, or again when other market imperfections exist. When the equilibrium is computed using LP optimization, as is the case for TIMES, the shadow price of each commodity is computed as the dual variable of that commodity's balance constraint, see chapter 14

The fact that the price of a commodity is equal to its marginal value is an important feature of competitive markets. Duality theory does not necessarily indicate that the marginal value of a commodity is equal to the marginal cost of *producing* that commodity. For instance, in the equilibrium shown in Figure 3.4 the price does not correspond to *any* marginal supply cost, since it is situated at a discontinuity of the inverse supply curve. In this case, the price is precisely determined by demand rather than by supply, and the term *marginal cost pricing* (so often used in the context of optimizing models) is *sensu stricto* incorrect. The term *marginal value pricing* is a more appropriate term to use.

It is important to reiterate that marginal value pricing *does not imply that suppliers have zero profit*. Profit is exactly equal to the suppliers' surplus, and is generally positive. Only the last few units produced may have zero profit, if, and when, their production cost equals the equilibrium price.

In TIMES the shadow prices of commodities play a very important diagnostic role. If some shadow price is clearly out of line (i.e. if it seems much too small or too large compared to the anticipated market prices), this indicates that the model's database may contain some errors. The examination of shadow prices is just as important as the analysis of the quantities produced and consumed of each commodity and of the technological investments.

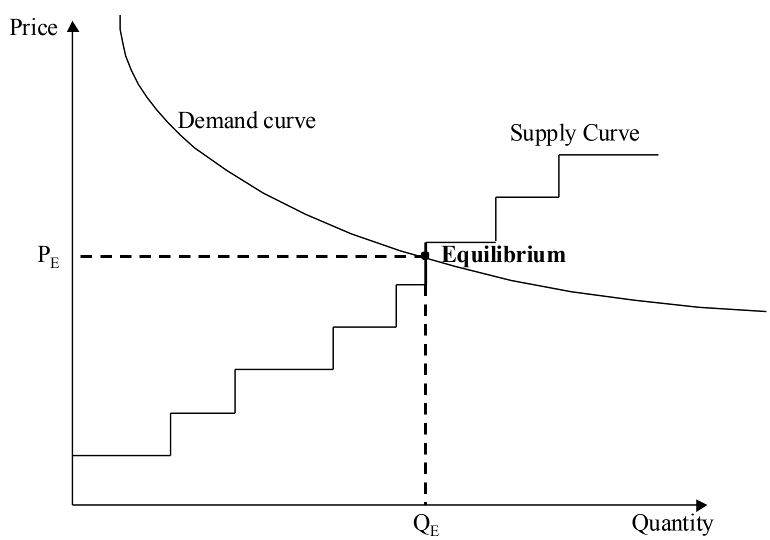


Figure 3.4. Case where the equilibrium price is not equal to any marginal supply cost.

3.2.5 Profit maximization: the Invisible Hand

An interesting property may be derived from the assumptions of competitiveness. While the avowed objective of the TIMES model is to maximize the overall surplus, it is also true that each economic agent in TIMES maximizes its own surplus. This property is akin to the famous 'invisible hand' property of competitive markets, and may be established rigorously by the following theorem that we state in an informal manner:

> *Theorem: Let (p*,q*) be a pair of equilibrium vectors that maximize total surplus. If we now replace the original TIMES linear program by one where all commodity prices are fixed at value p*, and we let each agent maximize its own surplus, the vector of optimal quantities produced or purchased by the agents also maximizes the total surplus*[24].

This property is important inasmuch as it provides an alternative justification for the class of equilibria based on the maximization of total surplus. It is now possible to shift the model's rationale from a global, societal one (total surplus maximization), to a local, decentralized one (individual utility maximization). Of course, the equivalence suggested by the theorem is valid only insofar as the marginal value pricing mechanism is strictly enforced—that is, neither an individual producer nor an individual consumer may affect market prices—both are price takers. Clearly, some markets are not competitive in the sense the term has been used here. For example, the behavior of a few oil producers has a dramatic impact on world oil prices, which then depart from their marginal system value. Market power[25] may also exist in cases where a few consumers dominate a market.

---

[24] However, the resulting Linear Program has multiple optimal solutions. Therefore, although q^* is an optimal solution, it is not necessarily the one found when the modified LP is solved.

[25] An agent has market power if its decisions, all other things being equal, have an impact on the market price. Monopolies and oligopolies are example of markets where one or several agents have market power.

4 Core TIMES model: Mathematics of the computation of the supply-demand equilibrium

In the preceding chapter, we have seen that TIMES does more than minimize the cost of supplying energy services. Instead, it computes a supply-demand equilibrium where both the energy supplies and the energy service demands are endogenously determined by the model. The equilibrium is driven by the user-defined specification of demand functions, which determine how each energy service demand varies as a function of the current market price of that energy service. The TIMES code assumes that each demand has constant own-price elasticity in a given time period, and that cross price elasticities are zero. We have also seen that economic theory establishes that the equilibrium thus computed corresponds to the maximization of the net total surplus, defined as the sum of the suppliers' and consumers' surpluses. We have argued in section 3.2 that the total net surplus has often been considered a valid metric of societal welfare in microeconomic literature, and this fact confers strong validity to the equilibrium computed by TIMES. Thus although TIMES falls short of computing a general equilibrium, it does capture a major element of the feedback effects not previously accounted for in bottom-up energy models.

In this chapter we provide the details on how the equilibrium is transformed into an optimization problem and solved accordingly.

Historically, the approach was first used in the Project Independence Energy System (PIES, see Hogan, 1975), although in the context of demands for final energy rather than for energy services as in TIMES or MARKAL. It was then proposed for MARKAL model by Tosato (1980) and Altdorfer (1982), and later made available as a standard MARKAL option by Loulou and Lavigne (1995). The TIMES implementation is identical to the MARKAL one.

4.1 *Theoretical considerations: the Equivalence Theorem*

The computational method is based on the equivalence theorem presented in chapter 3, which we restate here:

"A supply/demand economic equilibrium is reached when the sum of the producers and the consumers surpluses is maximized"

Figure 3.2 of Chapter 3 provides a graphical illustration of this theorem in a case where only one commodity is considered.

4.2 *Mathematics of the TIMES equilibrium*

4.2.1 Defining demand functions

From chapter 3, we have the following demand function for each demand category ***i***

$$DM_i / DM_i^0 = (p_i / p_i^0)^{E_i} \qquad (4-1)$$

Or its inverse:

$$\boldsymbol{p_i = p_i^0 \cdot (DM_i / DM_i^0)^{1/E_i}}$$

where the superscript ‘0’ indicates the reference case, and the elasticity $\boldsymbol{E_i}$ is negative. Note also that the elasticity may have two different values, one for upward changes in demand, the other for downward changes.

4.2.2 Formulating the TIMES equilibrium

With inelastic demands (i.e. pure cost minimization), the TIMES model may be written as the following Linear Program:

$$\begin{array}{lll} Min & c \cdot X & (4-2) \\ s.t. & \sum_k VAR\_ACT_{k,i}(t) \geq DM_i(t) \qquad i = 1,2,..,I;\ t = 1,..,T & (4-3) \\ and & B \cdot X \geq b & (4-4) \end{array}$$

where ***X*** is the vector of all TIMES variables and ***I*** is the number of demand categories. In words:

- (4-2) expresses the total discounted cost to be minimized. See chapter 5 for details on the list of TIMES variables ***X***, and on the cost vector ***c.***

- (4-3) is the set of demand satisfaction constraints (where the ***VAR_ACT*** variables are the activity levels of end-use technologies, and the ***DM*** right-hand-sides are the exogenous demands to satisfy).
- (4-4) is the set of all other TIMES constraints, which need not be explicated here, and are presented in chapter 5.

When demand are elastic, TIMES must compute a supply/demand equilibrium of the optimization problem (4-2) through (4-4), where the demand side adjusts to changes in prices, and the prevailing demand prices are the marginal costs of the demand categories (i.e. p_i is the marginal cost of producing demand DM_i). *A priori* this seems to be a difficult task, because the demand prices are computed as part of the dual solution to that optimization problem. The Equivalence Theorem, however, states that the equilibrium is reached as the solution of the following mathematical program, where the objective is to maximize the net total surplus:

$$Max \sum_i \sum_t \left(p_i^0(t) \bullet \left[DM_i^0(t) \right]^{-1/E_i} \bullet \int_a^{DM_i(t)} q^{1/E_i} \cdot dq \right) - c \cdot X \qquad (4-5)$$

$$s.t. \quad \sum_k VAR\_ACT_{k,i}(t) - DM_i(t) \geq 0 \qquad i = 1,..,I;\; t = 1,..,T \qquad (4-6)$$

$$and \quad B \cdot X \geq b \qquad (4-7)$$

where ***X*** is the vector of all TIMES variables, (4-5) expresses the total net surplus, and ***DM(t)***is now a vector of *variables* in (4-6), rather than fixed demands. The integral in (4-5) is easily computed, yielding the following maximization program:

$$Max \sum_i \sum_t \left(p_i^0(t) \cdot \left[DM_i^0(t) \right]^{-1/E_i} \bullet DM_i(t)^{1+1/E_i} / (1 + 1/E_i) \right) - c \cdot X \qquad (4-5)'$$

$$s.t. \quad \sum_k VAR\_ACT_{k,i}(t) \geq DM_i(t) \qquad i = 1,..,I;\; t = 1,..,T \qquad (4-6)'$$

$$B \cdot X \geq b \qquad (4-7)'$$

We are almost there, but not quite, since the $\boldsymbol{[DM_i(t)]^{1/Ei}}$ are non linear expressions and thus not directly usable in an L.P.

4.2.3 Linearization of the Mathematical Program

The Mathematical Program embodied in (4-5)', (4-6)' and (4-7)' has a non-linear objective function. Because the latter is separable (i.e. does not include cross terms) and concave in the DM_i variables, each of its terms is easily linearized by piece-wise linear functions which approximate the integrals in (4-5). This is the same as saying that the inverse demand curves are approximated by staircase functions, as illustrated in figure 4.1. By so doing, the resulting optimization problem becomes linear again. The linearization proceeds as follows.

a) For each demand category $\boldsymbol{i}$, and each time period $\boldsymbol{t}$, the user selects a range $\boldsymbol{R(t)_i}$, i.e. the distance between some values $\boldsymbol{DM_i(t)_{min}}$ and $\boldsymbol{DM_i(t)_{max}}$. The user estimates that the demand value $\boldsymbol{DM_i(t)}$ will always remain within such a range, even after adjustment for price effects (for instance the range could be equal to the reference demand $\boldsymbol{DM^o{}_i(t)}$ plus or minus 50%).

b) Select a grid that divides each range into a number n of equal width intervals. Let $ß_i(t)$ be the resulting common width of the grid, $\boldsymbol{\beta_i(t)= R_i(t)/n}$. See Figure 4.1 for a sketch of the non-linear expression and of its step-wise constant approximation. The number of steps, n, should be chosen so that the step-wise constant approximation remains close to the exact value of the function.

c) For each demand segment $\boldsymbol{DM_i(t)}$ define n step-variables (one per grid interval), denoted $s_{1,i}(t),\ s_{2,i}(t),\ \ldots,\ s_{n,i}(t)$. Each s variable is bounded below by 0 and above by $\beta_i(t)$. One may now replace in equations (4-5)' and (4-6)' each $\boldsymbol{DM_i(t)}$ variable by the sum of the n-step variables, and each non-linear term in the objective function by a weighted sum of the n step-variables, as follows:

$$DM_i(t) = DM(t)_{\min} + \sum_{j=1}^{n} s_{j,i}(t) \qquad 4-8$$

and

$$DM_i(t)^{1+1/E_i} \cong DM(t)_{\min}^{1+1/E_i} + \sum_{j=1}^{n} A_{j,s,i}(t) \bullet s_{j,i}(t) \qquad 4-9$$

The $\boldsymbol{A_{j,i,t}}$ term is equal to the value of the inverse demand function of the $\boldsymbol{j^{th}}$ demand at the mid-point of the $\boldsymbol{i^{th}}$ interval. The resulting Mathematical Program is now fully linearized.

Since the $\boldsymbol{A_{j,i,t}}$ terms have decreasing values (due to the concavity of the curve), the optimization will always make sure that the $\boldsymbol{s_{j,I}}$ variables are increased consecutively and in the correct order, thus respecting the step-wise constant approximation described above.

Remark: Instead of maximizing the linearized objective function, TIMES minimizes its negative, which then has the dimension of a cost. The portion of that cost representing the negative of the consumer surplus is akin to a *welfare loss.*

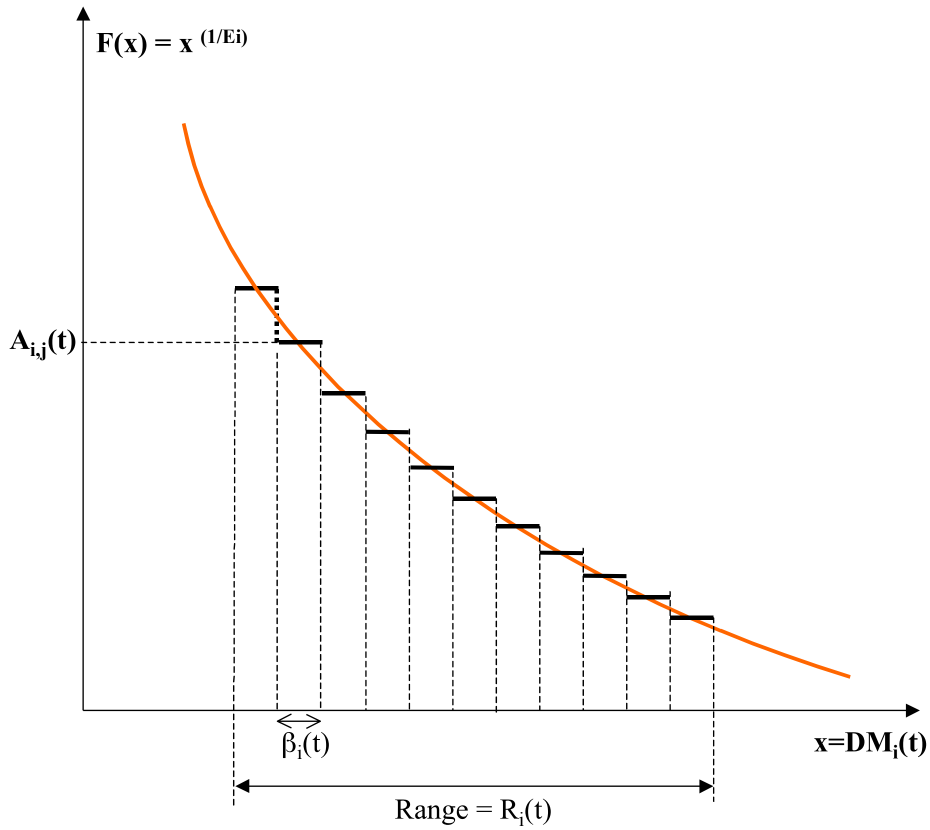


Figure 4.1. Step-wise constant approximation of the non-linear terms in the objective function

4.2.4 Calibration of the demand functions

Besides selecting elasticities for the various demand categories, the user must evaluate each constant $K_i(t)$. To do so, we have seen that one needs to know one point on each demand function in each time period, $\{ p^0_i(t), DM^0_i(t) \}$. To determine such a point, we perform a single preliminary run of the inelastic TIMES model (with exogenous $DM^0_i(t)$), and use the resulting shadow prices $p^0_i(t)$ for all demand constraints, in all time periods for each region.

4.2.5 Computational considerations

Each demand segment that is elastic to its own price requires the definition of as many variables as there are steps in the discrete representation of the demand curve (both upward and down if desired), for each period and region. Each such variable has an upper bound, but is otherwise involved in no new constraint. Therefore, the linear program is augmented by a number of variables, but does not have more constraints than the initial inelastic LP (with the exception of the upper bounds). It is well known that with modern LP codes the number of variables has little or no impact on computational time in Linear Programming, whether the variables are upper bounded or not. Therefore, the inclusion in TIMES of elastic demands has a very minor impact on computational time or on the tractability of the resulting LP. This is an important observation in view of the very large LP's that result from representing multi-regional and global models in TIMES.

4.2.6 Interpreting TIMES costs, surplus, and prices

It is important to note that, instead of maximizing the net total surplus, TIMES minimizes its negative (plus a constant), obtained by changing the signs in expression (4-5). For this and other reasons, it is inappropriate to pay too much attention to the meaning of the *absolute* objective function values. Rather, examining the difference between the objective function values of two scenarios is a far more useful exercise. That difference is of course, the negative of the difference between the net total surpluses of the two scenario runs.

Note again that the popular interpretation of shadow prices as the *marginal costs* of model constraints is inaccurate. Rather, the shadow price of a constraint is, by definition, the incremental value of the objective function per unit of that constraint's right hand side (RHS). The interpretation is that of an amount of *surplus loss* per unit of the constraint's RHS. The difference is subtle but nevertheless important. For instance, the shadow price of the electricity balance constraint is not necessarily the marginal cost of producing electricity. Indeed, when the RHS of the balanced constraint is increased by one unit, one of two things may occur: either the system *produces* one more unit of electricity, or else the system *consumes* one unit less of electricity (perhaps by choosing more efficient end-use devices or by reducing an electricity-intensive energy service, etc.) It is therefore correct to speak of shadow prices as the marginal *system value* of a resource, rather than the marginal *cost* of procuring that resource.

5 Core TIMES Model: A simplified description of the Optimization Program (variables, objective, constraints)

This chapter contains a simplified formulation of the core TIMES Linear Program.

Mathematically, a TIMES instance is a Linear Program, as was mentioned in the previous chapter. A Linear Program (LP for short) consists in the minimization or maximization of an *objective function* (defined as a linear mathematical expression of *decision variables)*, subject to linear *constraints,* also called *equations*[26].

Very large instances of Linear Programs involving sometimes millions of constraints and variables may be formulated using modern modeling languages such as GAMS (http://www.gams.com/help/index.jsp), and solved via powerful Linear Programming *optimizers*[27]. The Linear Program described in this chapter is much simplified, since it ignores many exceptions and complexities that are not essential to a basic understanding of the principles of the model. Chapter 14 gives additional details on general Linear Programming concepts. The full details of the parameters, variables, objective function, and constraints of TIMES are given in Part II of this documentation (sections 3, 5, and 6).

A linear optimization problem formulation consists of three types of entities:

- *the decision variables:* i.e. the unknowns, or endogenous quantities, to be determined by the optimization;
- *the objective function*: expressing the criterion to be minimized or maximized; and;
- *the constraints*: equations or inequalities involving the decision variables that must be satisfied by the optimal solution.

5.1 *Indices*

The model data structures (sets and parameters), variables and equations use the following indices:

[26] This rather improper term includes equality as well as inequality relationships between mathematical expressions.

[27] For more information on optimizers see Brooke et al., 1998

r: indicates the region

t or v: time period; ***t*** corresponds to the current period, and ***v*** is used to indicate the vintage year of an investment. When a process is not vintaged then ***v*** = ***t***.

p: process (technology)

s: time-slice; this index is relevant only for user-designated commodities and processes that are tracked at finer than annual level (e.g. electricity, low-temperature heat, and perhaps natural gas, etc.). Time-slice defaults to "ANNUAL", indicating that a commodity is tracked only annually.

c: commodity (energy, material, emission, demand).

5.2 *Decision variables*

The decision variables represent the *choices* to be made by the model, i.e. the *unknowns*. All TIMES variables are prefixed with the three letters VAR followed by an underscore.

Important remark: There are two possible choices concerning the very *meaning* of some decision variables, namely those variables that represent yearly flows or process activities. In the original TIMES formulation, the activity of a process during some period ***t*** is considered to be constant in all years constituting the period. This is illustrated in panel M.a of Figure 5.1).In the alternative option the activity variable is considered to represent the value *in a milestone year* of each period, and the values at all other years is linearly interpolated between the consecutive milestone year values, as illustrated in panel M.b). A milestone year is chosen close to the middle of a period. This second option is similar to that of the EFOM and the MESSAGE models. The user is free to choose either option. The constraints and objective function presented below apply to the first option (constant value of activity variables within a period). Appropriate changes in constraints and objective function are made for the alternative option, as explained in section 5.5, and more completely in Part II, section 6.

The main kinds of decision variables in a TIMES model are:

VAR_NCAP(r,v,p)**:** new capacity addition (investment) for technology ***p***, in period ***v*** and region ***r***. For all technologies the ***v*** value corresponds to the vintage of the process, i.e. year in which it is invested in. For vintaged technologies (declared as such by the user) the vintage (***v***) information is reflected in other process variables, discussed below. Typical units are PJ/year for most energy technologies, Million tonnes per year (for steel, aluminum, and paper industries), Billion vehicle-kilometers per year (B-vkm/year) or million cars for road vehicles, and GW for electricity equipment (1GW=31.536 PJ/year), etc.

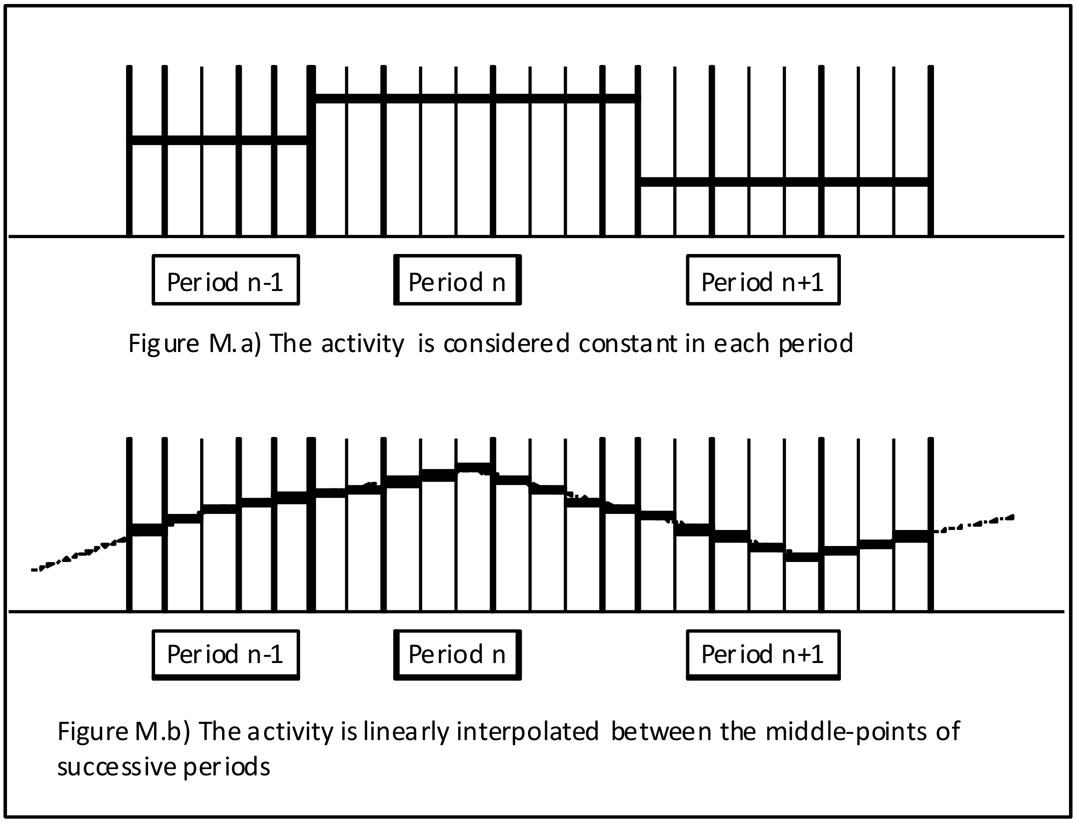


Figure 5.1.Process activity in the original TIMES formulation (top) and Linear variant (bottom)

VAR_RCAP(r,v,t,p): Amount of capacity that is newly retired at period ***t***. The new retirements will reduce the available capacity of vintage ***v*** in period ***t*** and in all successive periods $t_i > t$ by the value of the variable. This new feature was not available in early versions of TIMES. Note carefully that the feature must be activated by a special switch in order to become effective. Note also that additional a new advanced feature allows the user to specify that capacity retirement may only occur in lump amounts that are either equal to the entire remaining capacity or equal to a multiple of some user defined block. Consult the separate technical note *TIMES Early Retirement of Capacity* for details.

VAR_DRCAP(r,v,t,p,j): Binary variables used in formulating the special early retirement equations. Two variables may be defined, one when retirement must be for the entire remaining capacity **(j=1**), another when retirement must be a multiple of some block defined by the user via parameter RCAP_BLK **(j=2**).

VAR_SCAP(r,v,t,p):Total amount of capacity that has been retired at period ***t*** and periods preceding ***t*** (see above ***VAR_RCAP***paragraph).

CAP(r,v,t,p): installed capacity of process ***p***, in region ***r*** and period ***t,*** optionally with vintage ***v.*** It represents the total capacity available at period ***t***, considering the residual capacity at the beginning of the modeling horizon and adding to it new investments made prior to and including period ***t*** that have not reached their technical lifetime, and subtracting retired capacity. Typical units: same as investments. The ***CAP*** quantity, although convenient for formulation and reporting purposes, is in fact *not explicitly defined in the model*, but is derived from the ***VAR_NCAP*** variables and from data on past investments, process lifetimes, and any retirements.

VAR_CAP(r,t,p): total installed capacity of technology ***p***, in region ***r*** andperiod ***t***, all vintages together. The ***VAR_CAP*** variables are only defined when some bounds or user-constraints are specified for them. They do not enter any other equation.

Remark: The lumpy investment option. There is a TIMES feature that allows the user to impose that new additions to capacity may only be done in predefined blocks. This feature may be useful for technologies that are implementable only in discrete sizes such as a nuclear plant, or a large hydroelectric project. The user should however be aware that using this option voids some of the economic properties of the equilibrium. This feature is described in Chapter 10 of this part of the documentation.

VAR_ACT(r,v,t,p,s): activity level of technology ***p***, in region ***r*** and period ***t*** (optionally vintage ***v*** and time-slice ***s*)*.** Typical units: PJ for all energy technologies. The ***s*** index is relevant only for processes that produce or consume commodities specifically declared as time-sliced. Moreover, it is the process that determines which time slices prevail. By default, only annual activity is tracked.

VAR_FLO(r,v,t,p,c,s): the quantity of commodity ***c*** consumed or produced by process ***p***, in region ***r*** and period ***t*** (optionally with vintage ***v*** and time-slice ***s***). Typical units: PJ for all energy technologies. The ***VAR_FLO*** variables confer considerable flexibility to the processes modeled in TIMES, as they allow the user to define flexible processes for which input and/or output flows are not rigidly linked to the process activity.

VAR_SIN(r,v,t,p,c,s)/VAR_SOUT(r,v,t,p,c,s): the quantity of commodity ***c*** stored or discharged by storage process ***p,*** in time-slice ***s***, period ***t*** (optionally with vintage ***v)***, and region ***r***.

VAR_IRE(r,v,t,p,c,s,exp) and VAR_IRE(r,v,t,p,c,s,imp)*[28]*: quantity of commodity ***c*** (PJ per year) sold (***exp***) or purchased (***imp***) by region ***r*** through export (resp. import) process ***p*** in period ***t***(optionally in time-slice ***s***). Note that the topology defined for the exchange process ***p*** specifies the traded commodity ***c***, the region ***r***, and the regions ***r'*** with which region ***r*** is trading commodity ***c***. In the case of bi-lateral trading, if it is desired that region ***r*** should trade with several other regions, each such trade requires the definition of a separate bi-lateral exchange process. Note that it is also possible to define multi-lateral trading relationships between region ***r*** and several other regions ***r'*** by defining one of the regions as the common market for trade in commodity ***c***. In this case, the commodity is 'put on the market' and may be bought by any other region participating in the market. This approach is convenient for global commodities such as emission permits or crude oil. Finally, exogenous trading may also be modeled by specifying the ***r'*** region as an external region. Exogenous trading is required for models that are not global, since exchanges with non-modeled regions cannot be considered endogenous.

VAR_DEM(r,t,d): demand for end-use energy service ***d*** in region ***r*** and period ***t.*** It is a true variable, even though in the reference scenario, this variable is fixed by the user. In alternate scenarios however, ***VAR_DEM(r,t,d)*** may differ from the reference case demand due to the responsiveness of demands to their own prices (based on each service demand's own-price elasticity). Note that in this simplified formulation, we do not show the variables used to decompose ***DEM(r,t,d)*** into a sum of step-wise quantities, as was presented in chapter 4.

Other variables: Several options that have been added to TIMES over the successive versions require the definition of additional variables. They are alluded to in the sections describing the new options, and described more precisely in Part II, and in additional technical notes. Also, TIMES has a number of commodity related variables that are not strictly needed but are convenient for reporting purposes and/or for applying certain bounds to them. Examples of such variables are: the total amount produced of a commodity (***VAR_COMPRD***), or the total amount consumed of a commodity (***VAR_COMCON***).

Important remark: It is useful to know that many variables (for instance the above two accounting variables, but also the flow variables described earlier) add only a moderate computational burden to the optimization process, thanks to the use of a *reduction algorithm* to detect and eliminate redundant variables and constraints before solving the

[28] IRE stands for Inter-Regional Exchange

LP. These variables and constraints are later reinstated in the solution file for reporting purposes.

5.3 ***TIMES objective function: discounted total system cost***

5.3.1 The costs accounted for in the objective function

The Surplus Maximization objective is first transformed into an equivalent Cost Minimization objective by taking the negative of the surplus, and calling this value the *total system cost*. This practice is in part inspired from historical custom from the days of the fixed demand MARKAL model. The TIMES objective is therefore to minimize the total 'cost' of the system, properly augmented by the 'cost' of lost demand. All cost elements are appropriately discounted to a user-selected year.

In TIMES, the cost elements are defined at a finer level than the period. While the TIMES constraints and variables are linked to a *period*, the components of the system cost are expressed for each *year* of the horizon (and even for some years outside the horizon). This choice is meant to provide a smoother, more realistic rendition of the stream of cost payments in the energy system, as discussed below. Each year, the total cost includes the following elements:

- *Capital Costs* incurred for *investing* into and/or *dismantling* processes.
- Fixed and variable annual *Operation and Maintenance (O&M) Costs,* and other annual costs occurring during the dismantling of technologies.
- Costs incurred for *exogenous imports* and for domestic resource *extraction* and *production*. An exogenous import is one that imported from a non-specified entity, i.e. not from another modeled region. Exogenous imports are not relevant in global TIMES instances.
- Revenues from exogenous *export.* An exogenous export is one that is exported to a non-specified entity, i.e. not to another modeled region. Exogenous exports are irrelevant in global TIMES instances. Exogenous export earnings are revenues and appear with a negative sign in the cost expressions.
- *Delivery* costs for commodities consumed by the processes. These costs are attached to commodity flows.
- *Taxes* and *subsidies* associated with commodity flows and process activities or investments. A tax is not a cost *per se*. However, since the tax is intended to influence the optimization, it is considered as an integral part of the objective

function. It is however reported separately from regular costs. Similarly for subsidies.

- *Revenues from recuperation of embedded commodities,* accrued when a process's dismantling releases some valuable commodities.
- *Damage costs* (if defined) due to emissions of certain pollutants. Several assumptions are made: the damage costs in region ***r*** result from emissions in ***r*** and possibly in other regions; damage cost is imputed to the emitting region (polluter pays); emissions in period ***t*** entail damages in period ***t*** only; the damage cost from several types of emission is assumed to be the sum of the costs from each emission type (no cross-effect); and the damage function linking cost DAM to emissions EM is a power function of the form:

$$DAM(EM) = MC_0 \cdot \frac{EM^{\beta+1}}{(\beta+1)\cdot EM_0^{\beta}}$$

  Where β is non-negative (i.e. marginal damage costs are non decreasing). Hence, the damage cost function is linear (β=0) or non linear but convex (β >0). Therefore, the same linearization procedure that was used for the surplus may be applied here in order to linearize the damage cost[29]. Appendix B of Part II and Technical note "TIMES Damage", explain how to declare the various parameters required to define the damage functions, to specify the linearization parameters, and to define the switches used to control the optimization. It should be noted that global emissions such as GHG's should not be treated via this feature but rather should make use of the Climate Module option described in chapter 7.
- ***Salvage value*** of processes and embedded commodities at the end of the planning horizon. This revenue appears with a negative sign in the cost expressions. It should also be stressed that the calculation of the salvage value at the end of the planning horizon is very complex and that the original TIMES expressions accounting for it contained some biases (over- or under-estimations of the salvage values in some cases). These biases have been corrected in the present version of TIMES as explained in sections 5.3.4 and 5.5.
- *Welfare loss* resulting from reduced end-use demands. Chapter 4has presented the mathematical derivation of this quantity.

[29] Alternatively, one may use a convex programming code to solve the entire TIMES LP.

5.3.2 Cash flow tracking

As already mentioned, in TIMES, special care is taken to precisely track the cash flows related to process investments and dismantling in each year of the horizon. Such tracking is made complex by several factors:

- First, TIMES recognizes that there may be a lead-time (ILED) between the beginning and the end of the construction of some large processes, thus spreading the investment installments over several years. A recent TIMES feature allows the definition of a negative lead-time, with the meaning that the construction of the technology starts before the year the investment decision is made (this is useful for properly accounting for interest during construction, and is especially needed when using the time-stepped version of TIMES described in chapter 9.)
- Second, TIMES also recognizes that for some other processes (e.g. new cars), the investment in new capacity occurs *progressivel*y over the years constituting the time period (whose length is denoted by D(t)), rather than in one lumped amount.
- Third, there is the possibility that a certain investment decision made at period ***t*** will have to be repeated more than once during that same period. (This will occur if the period is long compared to the process technical life.)
- Fourth, TIMES recognizes that there may be dismantling capital costs at the end-of-life of some processes (e.g. a nuclear plant), and that these costs, while attached to the investment variable indexed by period ***t***, are actually incurred much later.
- Finally, TIMES permits the payment of any capital cost to be spread over an *economic life (ELIFE)* that is different from the *technical life (TLIFE)* of the process. Furthermore it may be annualized at a different rate than the overall discount rate.

To illustrate the above complexities, we present a diagram taken from Part II that pictures the yearly investments and yearly outlays of capital in one particular instance where there is no lead time and no dismantling of the technology, and the technical life of the technology does not exceed the period length. There are 4 distinct such instances, discussed in detail in section 6.2 of Part II.

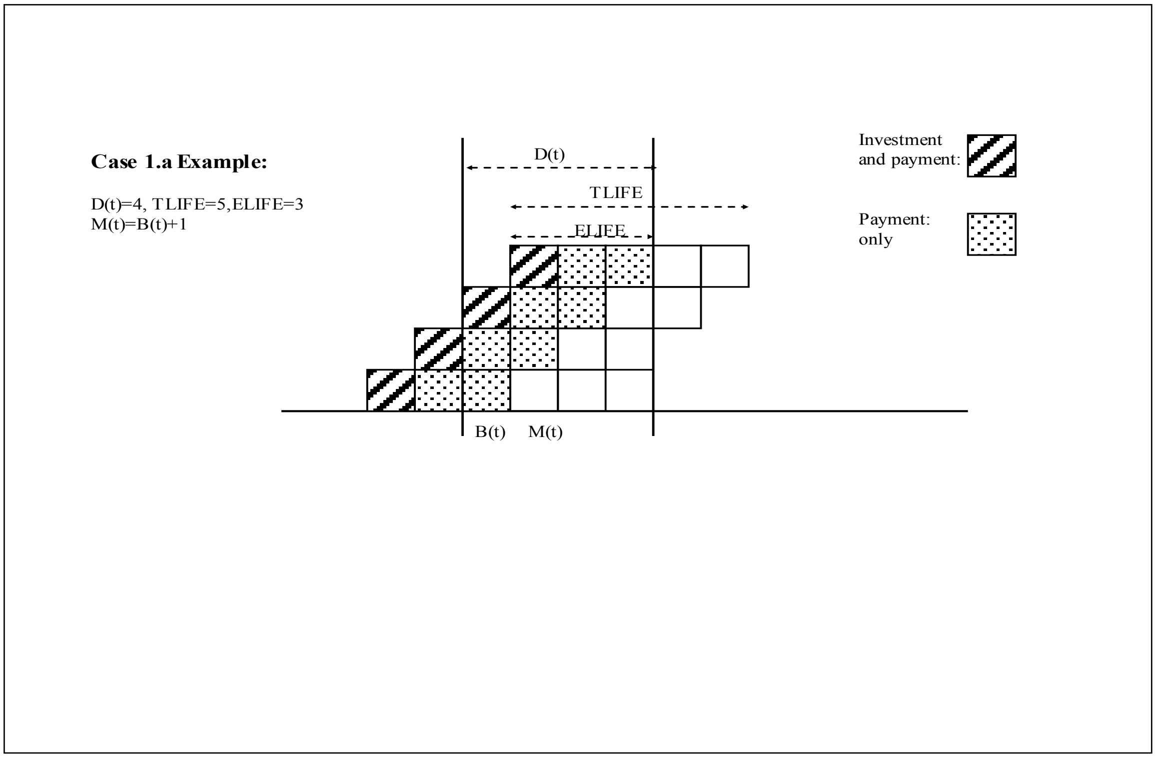


Figure 5.2. Illustration of yearly investments and payments for one of four investment tracking cases

5.3.3 Aggregating the various costs

The above considerations, while adding precision and realism to the cost profile, also introduce complex mathematical expressions into the objective function. In this simplified formulation, we do not provide much detail on these complex expressions, which are fully described in section 6.2 of Part II. We limit our description to giving general indications on the cost elements comprising the objective function, as follows:

- The capital costs (investment and dismantling) are first transformed into streams of annual payments, computed for each year of the horizon (and beyond, in the case of dismantling costs and recycling revenues), along the lines presented above.
- A *salvage value* of all investments still active at the end of the horizon (EOH) is calculated as a lump sum revenue which is subtracted from the other costs and

assumed to be accrued in the (single) year following the EOH.[30] It is then discounted to the user selected reference year.

- The other costs listed above, which are all annual costs, are added to the annualized capital cost payments, to form the ***ANNCOST*** quantity below.

TIMES then computes for each region a total net present value of the stream of annual costs, discounted to a user selected reference year. These regional discounted costs are then aggregated into a single total cost, which constitutes the objective function to be minimized by the model in its equilibrium computation.

$$NPV = \sum_{r=1}^{R} \sum_{y \in YEARS} (1 + d_{r,y})^{REFYR-y} \bullet ANNCOST(r,y)$$

where:

NPV	is the net present value of the total cost for all regions (the TIMES objective function);
ANNCOST(r,y)	is the total annual cost in region ***r*** and year ***y***;
$d_{r,y}$	is the general discount rate;
REFYR	is the reference year for discounting;
YEARS	is the set of years for which there are costs, including all years in the horizon, plus past years (before the initial period) if costs have been defined for past investments, plus a number of years after EOH where some investment and dismantling costs are still being incurred, as well as the Salvage Value; and
R	is the set of regions in the area of study.

As already mentioned, the exact computation of ***ANNCOST*** is quite complex and is postponed until section 6.2 of PART II

5.3.4 Variants for the objective function

There are some cases where the standard formulation described above leads to small distortions in the cost accounting between capacity-related costs and the corresponding activity-related costs. This occurs even without discounting but may be increased by

[30] The salvage value is thus the only cost element that remains lumped in the TIMES objective function. All other costs are annualized.

discounting. These distortions may occur at the end of the model horizon, either due to excessive or deficient salvage value.

In addition to these cost accounting problems at the end of horizon, the investment spreads used in the standard formulation can also lead to other cost distortions, regardless of discounting. In very long periods, the investment spreads are divided into D_t successive steps, each amounting to $1/D_t$ of the total capacity to be invested in the period. Recall that the full capacity must be in place by the milestone year, in order to allow activity to be constant over the period. For example, if the period length D_t is 20 years, the investments start already 19 years before the milestone year, and can thus start *even before the previous milestone year*. *If the investment costs are changing over time, it is clear that in such cases the costs will not be accounted in a realistic way, because the investment cost data is taken from the start year of each investment step.*

Similarly, in short periods the investment costs are spread over only a few years, and if the previous period is much longer, this can leave a considerable gap in the investment years between successive periods. Here again, if the investment costs are changing over time, this would lead to a distortion in the cost accounting.

Unfortunately, it is a well-known fact that the original choice of defining milestone years at or near the middle of each period limits the choice of milestone years, and furthermore tends to induce periods that may be very unequal in length, thus exacerbating the anomalies mentioned above. Such variability in period length can increase the cost distortions under discounting due to the larger differences in the timing of the available capacity (as defined by the investments) and the assumed constant activity levels in each period in the original definition of TIMES variables.

These were remedied by making changes in parts of the **OBJ** cost representation. Four options are now available, three of which apply to the original definition of TIMES variables, the fourth one applying to the alternate definition of TIMES variables. The fourth option (named **LIN**) is discussed separately in section 5.5, since it concerns not only the objective function but also several constraints.

The three options are as follows:

- The original OBJ with minor changes made to it, activated via the **OBLONG** switch.

- The modified objective function (**MOD**). The **MOD** formulation adds only a few modifications to the standard formulation:
  - The model periods are defined in a different way; and
  - The investment spreads in the investment Cases 1a and 1b (see section 6.2 of Part II for a list of all cases) are defined slightly differently.

- The **ALT** formulation includes all the modifications made in the MOD formulation. In addition, it includes the following further modifications that eliminate basically all of the remaining problems in the standard formulation:
  - The investment spreads in the investment Case 1b are defined slightly differently;
  - The capacity transfer coefficients for newly installed capacities are defined slightly differently, so that the effective lifetime of technologies is calculated taking into account discounting;
  - Variable costs are adjusted to be in sync with the available capacity.

It has been observed that these three options yield results that have practically the same degree of accuracy and reliability. There is however an advantage to the MOD and ALT options, as the milestone years need no longer be at the middle of a period.

Additional details and comments are provided on all three options in technical note "TIMES Objective Variants"

<u>Conclusion on the variants</u>: The multiplicity of options may confuse the modeler. Extensive experience with their use has shown that the distortions discussed above remain quite small. In practice, old TIMES users seem to stick to the classical OBJ with the OBLONG switch. And, as mentioned above, using MOD allows the further flexibility of freely choosing milestone years. Finally, using the LIN option (described in section 5.5) is a more serious decision, since it implies a different meaning for the TIMES variables; some modelers are more comfortable with this choice, which has also implications for the reporting of results.

5.4 *Constraints*

While minimizing total discounted cost, the TIMES model must satisfy a large number of constraints (the so-called *equations* of the model) which express the physical and logical relationships that must be satisfied in order to properly depict the associated energy system. TIMES constraints are of several kinds. Here we list and briefly discuss the main types of constraints. A full, mathematically more precise description is given in Part II. If any constraint is not satisfied, the model is said to be *infeasible*, a condition caused by a data error or an over-specification of some requirement.

In the descriptions of the equations that follow, the equation and variable names (and their indexes) are in ***bold italic*** type, and the parameters (and their indexes), corresponding to the input data, are in regular *italic* typeset. Furthermore, some parameter indexes have been omitted in order to provide a streamlined presentation.

5.4.1 Capacity transfer (conservation of investments)

Investing in a particular technology increases its installed capacity for the duration of the physical life of the technology. At the end of that life, the total capacity for this technology is decreased by the same amount. When computing the available capacity in some time period, the model takes into account the capacity resulting from all investments up to that period, some of which may have been made prior to the initial period but are still in operating condition (embodied by the residual capacity of the technology), and others that have been decided by the model at, or after, the initial period, up to and including the period in question.

The total available capacity for each technology *p,* in region *r*, in period *t* (all vintages),is equal to the sum of investments made by the model in past and current periods, and whose physical life has not yet ended, plus capacity in place prior to the modeling horizon that is still available. The exact formulation of this constraint is made quite complex by the fact that TIMES accepts variable time periods, and therefore the end of life of an investment may well fall in the middle of a future time period. We ignore here these complexities and provide a streamlined version of this constraint. Full details are shown in section 6.3.18 of Part II.

EQ_CPT(r,t,p) - Capacity transfer

> ***VAR_CAPT(r,t,p)*** = ***Sum****{over all periods t' preceding or equal to t such that*
> *t-t'<LIFE(r,t',p) of* ***VAR_NCAP(r,t',p)****}* + *RESID(r,t,p)*
> **(5-1)**

where *RESID(r,t,p)* is the (exogenously provided) capacity of technology *p* due to investments that were made prior to the initial model period and still exist in region *r* at time *t*.

5.4.2 Definition of process activity variables

Since TIMES recognizes activity variables as well as flow variables, it is necessary to relate these two types of variables. This is done by introducing a constraint that equates an overall activity variable, ***VAR_ACT(r,v,t,p,s)***, with the appropriate set of flow variables, ***VAR_FLO(r,v,t,p,c,s)***, properly weighted. This is accomplished by first identifying the group of commodities that defines the activity (and thereby the capacity as well) of the process. In a simple process, one consuming a single commodity and producing a single commodity, the modeler simply chooses one of these two flows to define the activity, and thereby the process normalization (input or output). In more complex processes, with several commodities (perhaps of different types) as inputs and/or outputs, the definition of the activity variable requires first to choose the *primary commodity group (pcg)* that will serve as the activity-defining group. For instance, the *pcg* may be the group of energy carriers, or the group of materials of a given type, or the group of GHG emissions, etc. The modeler then identifies whether the activity is defined via inputs or via outputs that belong to the selected *pcg*. Conceptually, this leads to the following relationship:

EQ_ACTFLO(r,v,t,p,s) – Activity definition

$$\boldsymbol{VAR\_ACT(r,v,t,p,s)} = \mathbf{SUM}\left\{c \text{ in } pcg \text{ of } \boldsymbol{VAR\_FLO(r,v,t,p,c,s)} \,/\, ACTFLO(r,v\ p,c)\right\} \qquad \textbf{(5-2)}$$

where *ACTFLO(r,v,p,c)* is a conversion factor (often equal to 1) from the activity of the process to the flow of a particular commodity.

5.4.3 Use of capacity

In each time period the model may use some or all of the installed capacity according to the Availability Factor (AF) of that technology. Note that the model may decide to use *less* than the available capacity during certain time-slices, or even throughout one or more whole periods, if such a decision contributes to minimizing the overall cost. Optionally, there is a provision for the modeler to force specific technologies to use their capacity to their full potential.

For each technology p, period t, vintage v, region r, and time-slice s, the activity of the technology may not exceed its available capacity, as specified by a user defined availability factor.

EQ_CAPACT (r,v,t,p,s) - Use of capacity

$$\boldsymbol{VAR\_ACT\,(r,v,t,p,s) \le or =} \quad AF(r,v,t,p,s) * PRC\_CAPACT(r,p))*FR(r,s)\boldsymbol{*VAR\_CAP(r,v,t,p)} \qquad \textbf{(5-3)}$$

Here *PRC_CAPACT(r,p)* is the conversion factor between units of capacity and activity (often equal to 1, except for power plants). The *FR(r,s)*parameter is equal to the (fractional) duration of time-slice s. The availability factor *AF* also serves to indicate the nature of the constraint as an inequality or an equality. In the latter case the capacity is forced to be fully utilized. Note that the ***CAP(r,v,t,p)***"variable" is not explicitly defined in TIMES. Instead it is replaced in (5-3) by a fraction (less than or equal to 1) of the investment variable ***VAR_NCAP(r,v,p)***[31] sum of past investments that are still operating, as in equation (5-1).

Example: *a coal fired power plant's activity in any time-slice is bounded above by 80% of its capacity, i.e.* ***VAR_ACT (r,v,t,p,s)*** *≤0.8*31.536 ** ***CAP(r,v,t,p),*** *where PRC_CAPACT(r,p) = 31.536 is the conversion factor between the units of the capacity variable (GW) and the activity-based capacity unit (PJ/a) The activity-based capacity unit is obtained from the activity unit(PJ) by division by a denominator of one year.*

The *s* index of the *AF* coefficient in equation (5-3) indicates that the user may specify time-sliced dependency on the availability of the installed capacity of some technologies, if desirable. This is especially needed when the operation of the equipment depends on the availability of a resource that cannot be stored, such as wind and sun, or that can be only partially stored, such as water in a reservoir. In other cases, the user may provide an *AF* factor that does not depend on *s*, which is then applied to the entire year. The operation profile of a technology within a year, if the technology has a sub-annual process resolution, is determined by the optimization routine. The number of ***EQ_CAPACT*** constraints is at least equal to the number of time-slices in which the

[31] That fraction is equal to 1 if the technical life of the investment made in period ***v*** fully covers period ***t***. It is less than 1 (perhaps 0) otherwise.

equipment operates. For technologies with only an annual characterization the number of constraints is reduced to one per period (where *s="ANNUAL"*).

5.4.4 Commodity balance equation

In each time period, the production by a region plus imports from other regions of each commodity must balance the amount consumed in the region or exported to other regions. In TIMES, the sense of each balance constraint (**≥** or =) is user controlled, via a special parameter attached to each commodity. However, the constraint defaults to an equality in the case of materials (i.e. the quantity produced and imported is *exactly* equal to that consumed and exported), and to an inequality in the case of energy carriers, emissions and demands (thus allowing some surplus production). For those commodities for which time-slices have been defined, the balance constraint must be satisfied in each time-slice.

The balance constraint is very complex, due to the many terms involving production or consumption of a commodity. We present a much simplified version below, to simply indicate the basic meaning of this equation.

For each commodity ***c,*** time period ***t (***vintage ***v)***, region ***r***, and time-slice ***s*** (if necessary or "ANNUAL" if not), this constraint requires that the disposition of each commodity balances its procurement. The disposition includes consumption in the region plus exports; the procurement includes production in the region plus imports.

EQ_COMBAL(r,t,c,s) - Commodity balance

[Sum *{over all* ***p,c*** *∈TOP(r,p,c,"out")of:* **[*VAR_FLO(r,v,t,p,c,s)*** **+**
VAR_SOUT(r,v,t,p,c,s)**STG_EFF(r,v,p)*]** ***}*** **+**

Sum *{over all* ***p,c*** *∈RPC_IRE(r,p,c,"imp") of*
*:**VAR_IRE(r,t,p,c,s,"imp")**}+*

Sum *{over all* ***p*** *of: Release(r,t,p,c)****VAR_NCAP(r,t,p,c)****}**]*** *******
COMM_IE(r,t,c,s)

≥ or = **(5-4)**

Sum *{over all* ***p,c ∈ TOP(r,p,c,"in")*** *of:* ***VAR_FLO(r,v,t,p,c,s)*** **+**
VAR_SIN(r,v,t,p,c,s)} **+**

Sum {*over all* ***p,c∈ RPC_IRE(r,p,c,"exp")}*** *of:*
VAR_IRE(r,t,p,c,s,'exp") +

Sum *{over all* ***p*** *of: Sink(r,t,p,c)*****VAR_NCAP(r,t,p,c)****} +*
FR(c,s) ****VAR_DEM(c,t)***

where:

The constraint is ≥ for energy forms and = for materials and emissions (unless these defaults are overridden by the user, see Part II).
TOP(r,p,c,"in/out") identifies that there is an input/output flow of commodity c into/from process *p* in region *r;*
RPC_IRE(r,p,c,"imp/exp") identifies that there is an import/export flow into/from region *r* of commodity *c* via process *p;*
STG_EFF(r,v,p) is the efficiency of storage process p;
COM_IE(r,t,c) is the infrastructure efficiency of commodity c;
Release(r,t,p,c) is the amount of commodity ***c*** recuperated per unit of capacity of process ***p*** dismantled (useful to represent some materials or fuels that are recuperated while dismantling a facility);
Sink(r,t,p,c) is the quantity of commodity ***c*** required per unit of new capacity of process ***p*** (useful to represent some materials or fuels consumed for the construction of a facility);
FR(s) is the fraction of the year covered by time-slice ***s*** (equal to 1 for non- time-sliced commodities).

Example*: Gasoline consumed by vehicles plus gasoline exported to other regions must not exceed gasoline produced from refineries plus gasoline imported from other regions.*

5.4.5 Defining flow relationships in a process

A process with one or more (perhaps heterogeneous) commodity flows is essentially defined by one or more input and output flow variables. In the absence of relationships between these flows, the process would be completely undetermined, i.e. its outputs would be independent from its inputs. We therefore need one or more constraints stating in a most general case that the ratio of the sum of some of its output flows to the sum of some of its input flows is equal to a constant. In the case of a single commodity in, and a single commodity out of a process, this equation defines the traditional efficiency of the process. With several commodities, this constraint may leave some freedom to individual output (or input) flows, as long as their sum is in fixed proportion to the sum of input (or output) flows. An important rule for this constraint is that *each sum must be taken over commodities of the same type* (i.e. in the same group, say: energy carriers, or emissions, etc.). In TIMES, for each process the modeler identifies the input commodity group *cg*1, and the output commodity group *cg*2, and chooses a value for the efficiency ratio, named *FLO_FUNC(p,cg1,cg2).* The following equation embodies this:

EQ_PTRANS(r,v,t,p,cg1,cg2,s) –Efficiency definition

> ***SUM***$\{$*c in cg2 of :* ***VAR_FLO(r,v,t,p,c,s)***$\}$ =
>
> *FLO_FUNC(r,v,cg1,cg2,s)* * ***SUM***$\{$*c within cg1 of:*
> *COEFF(r,v,p,cg1,c,cg2,s)*****VAR_FLO(r,v,t,p,c,s)***$\}$ **(5-5)**

where *COEFF(r,v,p,cg1,c,cg2,s)* takes into account the harmonization of different time-slice resolution of the flow variables, which have been omitted here for simplicity, as well as commodity-dependent transformation efficiencies.

5.4.6 Limiting flow shares in flexible processes

When either of the commodity groups*cg*1 or *cg*2 contains more than one element, the previous constraint allows a lot of freedom on the values of flows. The process is therefore quite flexible. The flow share constraint is intended to limit the flexibility, by constraining the share of each flow within its own group. For instance, a refinery output might consist of three refined products: *c1*=light, *c2*=medium, and *c3*=heavy distillate. If losses are 9% of the input, then the user must specify *FLO_FUNC* = 0.91 to define the overall efficiency. The user may then want to limit the flexibility of the slate of outputs by means of three *FLO_SHAR(ci) coefficients, say 0.4, 0.5, 0.6,* resulting in three flow share constraints as follows (ignoring some indices for clarity):

VAR_FLO(c1) ≤ 0.4[VAR_FLO(c1) + VAR_FLO(c2) + VAR_FLO(c3)],* so that *c*1 is at most 40% of the total output,

VAR_FLO(c2) ≤ 0.5[VAR_FLO(c1) + VAR_FLO(c2) + VAR_FLO(c3)],* so that *c*2 is at most 50% of the total output,

$VAR\_FLO(c3) \leq 0.6*[VAR\_FLO(c1) + VAR\_FLO(c2) + VAR\_FLO(c3)]$, so that *c*3 is at most 60% of the total output,

The general form of this constraint is:

EQ_INSHR(c,cg,p,r,t,s) and *EQ_OUTSHR(c,cg,p,r,t,s)*

VAR_FLO(c)≤,≥, =
*FLO_SHAR(c) * Sum {over all c' in cg of:* ***VAR_FLO(c')*** *}* **(5-6)**

The commodity group *cg* may be on the input or output side of the process.

A recent modification of TIMES simplifies the above constraints by allowing the use of the ***VAR_ACT*** variable instead of the sum of ***VAR_FLO*** variables in equation (5-6) or in similar ones. This simplification is triggered when the user defines the new attribute ACT_FLO, which is a coefficient linking a flow to the activity of a process. Furthermore, commodity ***c*** appearing in left-hand-side of the constraint may even be a flow that is not part of the ***cg*** group.

Warning: It is quite possible (and regrettable) to over specify flow related equations such as (5-6), especially when the constraint is an equality. Such an over specification leads to an infeasible LP. A new feature of TIMES consists in deleting some of the flow constraints in order to re-establish feasibility, in which case a warning message is issued.

5.4.7 Peaking reserve constraint (time-sliced commodities only)

This constraint imposes that the total capacity of all processes producing a commodity at each time period and in each region must exceed the average demand in the time-slice where peaking occurs by a certain percentage. This percentage is the Peak Reserve Factor, *COM_PKRSV(r,t,c,s)*, and is chosen to insure against several contingencies, such as: possible commodity shortfall due to uncertainty regarding its supply (e.g. water availability in a reservoir); unplanned equipment down time; and random peak demand that exceeds the average demand during the time-slice when the peak occurs. This constraint is therefore akin to a safety margin to protect against random events not explicitly represented in the model. In a typical cold country the peaking time-slice for electricity (or natural gas) will be Winter-Day, and the total electric plant generating capacity (or gas supply plant) must exceed the Winter-Day demand load by a certain percentage. In a warm country the peaking time-slice may be Summer-Day for electricity (due to heavy air conditioning demand). The user keeps full control regarding which time-slices have a peaking equation.

For each time period ***t*** and for region ***r***, there must be enough installed capacity to exceed the required capacity in the season with largest demand for commodity ***c*** by a safety factor *E* called the *peak reserve factor*.

EQ_PEAK(r,t,c,s) - Commodity peak requirement

> ***Sum*** *{over all* ***p*** *producing* **c with c=pcg** *of PRC_CAPACT(r,p) * Peak(r,v,p,c,s) * FR(s)* ****VAR_CAP(r,v,t,p) * VAR_ACTFLO(r,v,p,c)*** *} +*
>
> ***Sum*** *{over all* ***p*** *producing* ***c*** *with* ***c≠pcg*** *of NCAP_PKCNT(r,v,p,c,s)* ****VAR_FLO(r,v,t,p,c,s)****} +****VAR_IRE(r,t,p,c,s,i)***
>
> **≥** **(5-7)**
>
> *[1+ COM_PKRSV(r,t,c,s)] * [* ***Sum*** *{over all* ***p consuming c*** *of* ***VAR_FLO(r,v,t,p,c,s) +VAR_IRE(r,t,p,c,s,e)****}]*

where:

COM_PKRSV(r,t,c,s) is the region-specific reserve coefficient for commodity *c* in time-slice *s*, which allows for unexpected down time of equipment, for demand at peak, and for uncertain resource availability, and

NCAP_PKCNT(r,v,p,c,s) specifies the fraction of technology *p*’s capacity in a region *r* for a period *t* and commodity *c* (electricity or heat only) that is allowed to contribute to the peak load in slice *s*; many types of supply processes are predictably available during the peak and thus have a peak coefficient equal to 1, whereas others (such as wind turbines or solar plants in the case of electricity) are attributed a peak coefficient less than 1, since they are on average only fractionally available at peak (e.g., a wind turbine typically has a peak coefficient of .25 or .3, whereas a hydroelectric plant, a gas plant, or a nuclear plant typically has a peak coefficient equal to 1).

For simplicity it has been assumed in (5-7) that the time-slice resolution of the peaking commodity and the time-slice resolution of the commodity flows (FLO, TRADE) are the same. In practice, this is not the case and additional conversion factors or summation operations are necessary to match different time-slice levels.

Remark: to establish the peak capacity, two cases must be distinguished in constraint *EQ_PEAK*.
– For production processes where the peaking commodity is the only commodity in the primary commodity group (denoted c=pcg), the capacity of the process may be assumed to contribute to the peak.
– For processes where the peaking commodity is not the only member of the pcg, there are several commodities included in the pcg. Therefore, the capacity as such cannot be used in the equation. In this case, the actual production is taken into account in the contribution to the peak, instead of the capacity. For example, in the case of CHP only the production of electricity contributes to the peak electricity supply, not the entire capacity of the plant, because the activity of the process consists of both electricity and heat generation in either fixed or flexible proportions, and, depending on the modeler's choice, the capacity may represent either the electric power of the turbine in condensing or back-pressure mode, or the sum of power and heat capacities in back-pressure mode. There is therefore a slight inconsistency between these two cases, since in the first case, a technology may contribute to the peak requirement without producing any energy, whereas this is impossible in the second case.

Note also that in the peak equation (5-7), it is assumed that imports of the commodity are contributing to the peak of the importing region (thus, exports are implicitly considered to be of the *firm power* type).

5.4.8 Constraints on commodities

In TIMES variables are optionally attached to various quantities related to commodities, such as total quantity produced. Therefore it is quite easy to put constraints on these quantities, by simply bounding the commodity variables in each period. It is also possible to impose cumulative bounds on commodities over more than one period, a particularly useful feature for cumulatively bounding emissions or modeling reserves of fossil fuels. By introducing suitable naming conventions for emissions the user may constrain emissions from specific sectors. Furthermore, the user may also impose global emission constraints that apply to several regions taken together, by allowing emissions to be traded across regions. Alternatively or concurrently a tax or penalty may be applied to each produced (or consumed) unit of a commodity (energy form, emission), via specific parameters.

A specific type of constraint may be defined to limit the share of process (p) in the total production of commodity (c). The constraint indicates that the flow of commodity (c) from/to process (p) is bounded by a given fraction of the total production of commodity (c). In the present implementation, the same given fraction is applied to all time slices.

5.4.9 User constraints

In addition to the standard TIMES constraints discussed above, the user may create a wide variety of so-called User Constraints (UC's), whose coefficients follow certain rules. Thanks to recent enhancements of the TIMES code, user defined constraints may involve virtually any TIMES variable. For example, there may a user-defined constraint limiting investment in new nuclear capacity (regardless of the type of reactor), or dictating that a certain percentage of new electricity generation capacity must be powered by a portfolio of renewable energy sources. User constraints may be employed across time periods, for example to model options for retrofitting existing processes or extending their technical lives. A frequent use of UC's involves cumulative quantities (over time) of commodities, flows, or process capacities or activities. Recent TIMES code changes make the definition of the right-hand-sides of such UC's fairly independent of the horizon chosen for the scenario, and thus make it unnecessary to redefine the RHS's when the horizon is changed.

In order to facilitate the creation of a new user constraint, TIMES provides a *template* for indicating a) the set of variables involved in the constraint, and b) the user-defined coefficients needed in the constraint.

The details of how to build different types of UC are included in section 6.4 of Part II of the documentation.

5.4.10 Growth constraints

These are special cases of UC's that are frequently used to maintain the growth (or the decay) of the capacity of a process within certain bounds, thus avoiding excessive abrupt investment in new capacity. Such bounding of the growth is often justified by the reality of real life constraints on technological adoption and evolution. The user is however advised to exert caution on the choice of the maximum rates of technological change, the risk being to restrict it too much and thus "railroad" the model.

Typically, a growth constraint is of the following generic form (ignoring several indices for clarity:

$$\boldsymbol{VAR\_CAP}(\boldsymbol{t}+\mathbf{1}) \leq \left(\mathbf{1}+\boldsymbol{GROWTH}^{\boldsymbol{M}(\boldsymbol{t}+\mathbf{1})-\boldsymbol{M}(\boldsymbol{t})}\right)\times\boldsymbol{VAR\_CAP}(\boldsymbol{t})+\boldsymbol{K} \qquad (5\text{-}8)$$

The GROWTH coefficient is defined as a new attribute of the technology, and represents the maximum annual growth allowed for the capacity. The quantity M(t+1)-M(t) is the number of years between the milestones of periods t and t+1. The constant K is useful whenever the technology has no capacity initially, in order to allow capacity to build over time (if K were absent and initial capacity is zero, the technology would never acquire any capacity)

Note that the sign of the constraint may also be of the "larger than or equal to" type to express a maximum rate of abandonment, in which case the "+" sign is replaced by a "–" sign in the right-hand-side of the constraint. Equality is also allowed, but must be used only exceptionally in order to avoid railroading of the model.

5.4.11 Early retirement of capacity

With this new TIMES feature the user may allow the model to retire some technologies before the end of their technical lives. The retirement may be continuous or discrete. In the former case, the model may retire any amount of the remaining capacity (if any) at each period. In the latter case, the retirement may be effected by the model either in a single block (i.e. the remaining capacity is completely retired) or in multiples of a user chosen block. Please refer to chapter 10 of this document *The lumpy investment option*, for additional discussion of the mathematical formulation of MIP problems.

This feature requires the definition of three new constraints, as listed and briefly described in table 5.1, as well as the alteration of many existing constraints and the objective function, as described in table 5.2 Part II and the special separate note *TIMES Early Retirement of Capacity* provide additional detail.

The user is advised to use the discrete early retirement feature sparingly, as it implies the use of mixed integer programming optimizer, rather than the computationally much more efficient linear programming optimizer. The user should also be aware that using the discrete option voids some of the economic properties of the equilibrium, as discussed in section 10.3.

New Equation	Description
EQ_DSCRET(r,v,t,p)	Discrete retirement equation for process **p** and vintage **v** in region **r** and period **t**. Plays an analogous role to equation EQ_DSCNCAP in the Discrete Capacity Investment Extension.
EQ_CUMRET(r,v,t,p)	Cumulative retirement equation for process **p** and vintage **v** in region **r** and period **t**.
EQL_SCAP(r,t,p,ips)	Maximum salvage capacity constraint for process **p** in region **r** and period **t**, defined for **ips** = N (unless NCAP_OLIFE is specified).

Table 5.1. The new constraints required to implement early retirement of capacity

Existing Equation	Equation Description	Purpose of Modification
EQ_OBJFIX	Fixed cost component of objective function	To credit back the fixed costs of the capacity that is retired early
EQ_OBJVAR	Variable cost component of objective function	To reflect the effect of capacity that is retired early in the costs of capacity-related flows
EQ_OBJSALV	Salvage cost component of objective function	To subtract the salvage value (if any) of capacity that is retired early
EQ(**l**)_CPT for **l** = L, E, G	Capacity transfer equation	To reflect the effect of capacity that is retired early
EQ(**l**)_CAPACT for **l** = L, E, G	Capacity utilization equation	To reflect the effect of capacity that is retired early
EQ(**l**)_CAFLAC for **l** = L, E	Commodity based availability constraint	To reflect the effect of capacity that is retired early
EQ(**l**)_COMBAL for **l** = G, E	Commodity balance equation	To reflect the effect of capacity that is retired early in capacity-related flows
EQ_PEAK	Commodity peaking constraint	To subtract the peak contribution of capacity that is retired early
EQ(**l**)_UC* for **l** = L, E, G	The FLO component of all user constraints	To reflect the effect of capacity that is retired early in capacity-related flows
EQ(**l**)_MRKCON for **l** = L, E, G	Market share of flow in the consumption of a commodity	To reflect the effect of capacity that is retired early in capacity-related flows
EQ(**l**)_MRKPRD for **l** = L, E, G	Market share of flow in the production of a commodity	To reflect the effect of capacity that is retired early in capacity-related flows

Table 5.2. List of existing constraints that are affected by the early retirement option.

5.4.12 Electricity grid modeling

The electricity sector plays a central role in any energy model, and particularly so in TIMES. The electricity commodity has features that present particular challenges for its representation, in that it is difficult to store, and requires a network infrastructure to be transported and delivered. The considerable development of new renewable electricity generation technologies adds to the complexity, inasmuch as the technical requirements of integrating interruptible generation facilities (such as wind turbines and solar plants) to a set of traditional plants, must be satisfied for the integration to be feasible. Such considerations become even more relevant in large regions or countries, where the distances between potential generation areas and consumption areas are quite large.

Such considerations have led to the introduction of an optional grid modeling feature into the TIMES model's equations. A grid consists in a network of nodes linked by arcs (or branches). Each node may represent a well-defined geographic area that is deemed distinct from other areas of the region, either because of its generation potential (e.g. a windy area suitable for wind farms) and/or because of a concentration of points of consumption of electricity (e.g. a populated area separated from other populated areas or from generation areas.)

The purpose of this section is to indicate the broad principles and characteristics of the grid representation feature in TIMES. The modeler wishing to implement the feature is urged to read to the detailed Technical Note “TIMES Grid modeling feature”, which contains the complete mathematical derivations of the equations, and their implementation in TIMES. What follows is a much streamlined version outlining only the main approach and ignoring the many details of the mathematical equations.

5.4.12.1 A much simplified sketch of the grid constraints

The traditional way to represent the nodes and arcs of a grid is shown in figure 5.3, where each node is shown as a horizontal segment, and the nodes are connected via bi-directional arcs.

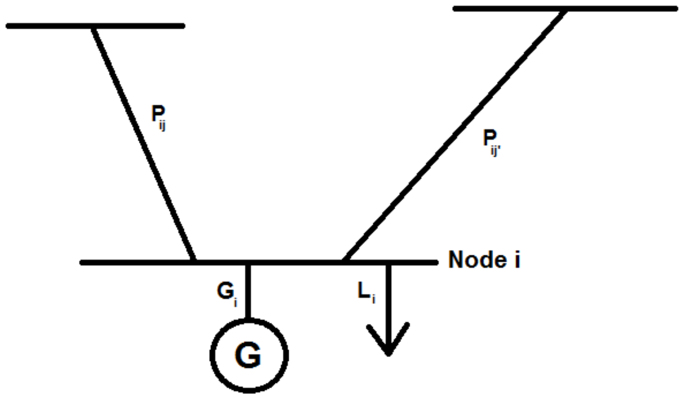


Figure 5.3. Connection of a grid node with other nodes

The basic energy conservation equation of a grid is as follows:

$$G_i - L_i = \sum_{j=1}^{M} P_{i,j} \quad \textit{for each } i=1,2,\ldots,M$$

where:

M = the number of nodes connected with node i
G_i = active power injected into node i by generators
L_i = active power withdrawn from node i by consumer loads
P_{ij} = branch flow from node i to node j

As mentioned above, these constraints are then modified so as to include important technical requirements on the electrical properties (reactance and phase angle) of each line. Suffice it to say here that the resulting new equations remain linear in the flow and other variables.

5.4.12.2 Integrating grid equations into TIMES

It should be clear that the variables G_i and L_i must be tightly related to the rest of the TIMES variables that concern the electricity commodities. In fact, the modeler must first decide on an allocation of the set of generation technologies into M subsets, each subset being attached to a node of the grid. Similarly, the set of all technologies that consume electricity must also be partitioned into M subsets, each attached to a node. These two partitions are effected via new parameters specifying the fractions of each generation type to be allocated to each grid node, and similarly for the fractions of each technology consuming electricity to be allocated to grid each node. This indeed amounts to a partial regionalization of the model concerning the electricity sector. Thus, variables G_i and L_i are defined in relation to the existing TIMES variables.

Of course, the introduction of the grid requires modifying the electricity balance equations and peak equations, via the introduction of the net total flow variables the set of grid nodes. The electricity balance equations are modified for each time slice defined for electricity.

Finally, additional a security constraint is added in the case of a multi-regional model, expressing that the total net export or import of electricity from region ***r*** does not exceed a certain (user-defined) fraction of the capacity of the portion of the grid linking region ***r*** to other regions.

5.4.12.3 Costs

New costs attached to the grid are also modeled, and form a new component of the objective function for the region. For this, a new cost coefficient is defined and attached to each node of the grid. TIMES multiplies this cost coefficient by the proper new grid variables and discounts the expression in order to form the new OBJ component.

5.4.13 Reporting "constraints"

These are not constraints proper but expressions representing certain quantities useful for reporting, after the run is completed. They have no impact on the optimization. We have already mentioned ***CAP(r,v,t,p)***, which represents the capacity of a process by vintage.

One sophisticated expression reports the *levelized cost* (LC) of a process. A process's LC is a life cycle quantity that aggregates all costs attached to a process, whether explicit or implicit. It is a useful quantity for ranking processes. However, such a ranking is dependent upon a particular model run, and may vary from run to run. This is so because several implicit costs attached to a process such as the cost of fuels used or produced, and perhaps the cost of emissions, are run dependent.

The general expression for the levelized cost of a process is as follows:

$$LEC = \frac{\sum_{t=1}^{n} \frac{IC_t}{(1+r)^{t-1}} + \frac{OC_t + VC_t + \sum_i FC_{i,t} + FD_{i,t} + \sum_j ED_{j,t}}{(1+r)^{t-0.5}} - \frac{\sum_k BD_{k,t}}{(1+r)^{t-0.5}}}{\sum_{t=1}^{n} \frac{\sum_m MO_{m,t}}{(1+r)^{t-0.5}}} \tag{5-9}$$

where

- r = discount rate (e.g. 5%)
- IC_t = investment expenditure in (the beginning of) year t
- OC_t = fixed operating expenditure in year t
- VC_t = variable operating expenditure in year t
- FC_{it} = fuel-specific operating expenditure for fuel i in year t
- FD_{it} = fuel-specific acquisition expenditure for fuel i in year t
- ED_{jt} = emission-specific allowance expenditure for emission j in year t (optional)
- BD_{kt} = revenues from commodity k produced by the process in year t (optional;)
- MO_{mt} = output of main product m in year t, i.e. a member of the *pcg*

Comments:

Each cost element listed above is obtained by multiplying a unit cost by the value of the corresponding variable indicated in the run results.

The unit values of the first four costs are simply equal the process input data, i.e. the unit investment cost, the fixed unit O&M cost, the unit variable operating cost, and the unit delivery cost. The last three costs are the shadow prices of the commodities concerned, endogenously obtained as the dual solution of the current model run.

Note also that the user may choose to ignore the last two costs or to include them. Furthermore, concerning the last cost (which is indeed a revenue), the user may decide to ignore the revenue from the main commodities produced by the process and retain only the revenues from the by-products. The choice is specified via the parameter RPT_OPT(‘NCAP’,’1’). Technical note "Levelized costs-TIMES" provides details on the parameter values.

5.5 *The 'Linear' variant of TIMES*

This alternate TIMES formulation (called the LIN variant) assumes a different meaning for the activity and flow variables of TIMES. More precisely, instead of assuming that flows and activities are constant in all years within the same period, the variant assumes that the flow and activity variables apply to only one milestone year within each period. The variables' values at other years of a period are interpolated between successive milestone years' values. See section 5.2 for a figure depicting the two alternate definitions.

Choosing the LIN formulation affects the variable costs in the objective function as well as all dynamic constraints involving activities or flows. Note also that the LIN variant avoids the cost distortions mentioned in section 5.3.1.

Significant modifications in the LIN formulation concern the variable cost accounting, since the latter are no longer constant in all years of any given period, but evolve linearly between successive milestone years. The objective function components for all variable costs have been modified accordingly.

The following further modifications are done in the LIN formulation:

- The cumulative constraints on commodity production (EQ(l)_CUMNET and EQ(l)_CUMPRD) are modified to include linear interpolation of the commodity variables involved;
- The cumulative constraints on commodity and flow taxes and subsidies (EQ(l)_CUMCST) are modified to include linear interpolation of the commodity and flow variables involved;
- The dynamic equations of the Climate module are modified to include linear interpolation of the variables involved;
- The inter-period storage equations are modified to include linear interpolation of the flow variables involved;
- The cumulative user constraints for activities and flows are also modified in a similar manner.
- Note that in the LIN formulation the activity of ***inter-period storage*** equations is measured at the milestone year (in the standard formulation it is measured at the end of each period). In addition, new EQ_STGIPS equations are added to ensure that the storage level remains non-negative at the end of each period. (Without these

additional constraints, the linear interpolation of storage could lead to a negative storage level if the period contains more than a single year.)

6 Parametric analysis with TIMES

Dealing with uncertainty in modeling is a complex endeavour that may be accomplished via a number of (sometimes widely different) approaches. In the case of TIMES, two different features are available: ***Stochastic Programming*** (treated in chapter 8) and ***parametric analysis***, also known as ***sensitivity analysis***, which is the subject of this chapter. In sensitivity analysis, the values of some important exogenous assumptions are varied, and a series of model runs is performed over a discrete set of combinations of these assumptions. Sensitivity analysis is often combined with ***tradeoff analysis***, where the tradeoff relation between several objectives is analyzed.

The uncertain attributes are similar to the corresponding standard TIMES attributes, but they may now have different values according to the different ***states-of-the-world*** (SOW), just as in the case of stochastic programming. The difference between the two approaches is that sensitivity analysis solves a sequence of instances, each assuming different values of the uncertain parameters, whereas stochastic programming solves a single instance that encompasses all potential values of the uncertain parameters simultaneously.

In TIMES, sensitivity analysis and tradeoff analysis facility are implemented using the same setup and some of the attributes of the stochastic mode of TIMES, since both approaches, although conceptually different, use the same state of the world construct.

Here are a few possible set-ups for sensitivity and tradeoff analyses in TIMES, all of which are supported by the model generator:

A. Single phase sensitivity analysis over the set of SOWs. Each run corresponds to a set of values for the uncertain parameters. The runs are mutually independent. This is the most straightforward approach;
B. Two-phase tradeoff analysis, where the model is first run using a user-defined objective function, and then the TIMES objective function is used in phase 2, while the solution from the first phase is used for defining additional constraints in a series of model runs in the second phase.
C. Multiphase tradeoff analysis over N phases, which is a generalization of the two-phase case.

Analyzing tradeoffs between the standard objective function and some other possible objectives (for which the market is not able to give a price) was not possible in an effective way with earlier versions of TIMES.

6.1 ***Two-phase tradeoff analysis***

In the ***first phase*** of the TIMES two-phase tradeoff analysis facility, the objective function is user defined as a weighted sum of any number of components, each component being a user constraint's left-hand-side. All UC's must be of the global type, (i.e. aggregated over regions and periods). Optionally, each of the component UCs may also be constrained by upper/lower bounds. The components are defined by the user, via the specification of non-zero weight coefficients for the UC's to be included in the objective. The original objective function (total discounted costs) is automatically pre-defined as a non-constraining user constraint with the name '***OBJZ***', and can therefore always be directly used as one of the component UCs, if desired.

Consequently, the first phase can be considered as representing a Utility Tradeoff Model, which can also be used as a stand-alone option. If used in a stand-alone manner, it constitutes a case of multi-criterion decision making (see e.g. Weistroffer, 2005). The resulting objective function to be minimized can be written as follows:

$$\min obj1 = \sum_{uc \in UC\_GLB} W(uc) \bullet LHS(uc)$$

where:

W(uc)	=	weight of objective component *uc* in Phase 1
LHS(uc)	=	LHS expression of user constraint *uc* according to its definition
UC_GLB	=	the set of all global UC constraints (including '***OBJZ***')

In the ***second phase*** of the TIMES two-phase tradeoff analysis facility the objective function is always the ***original objective function*** in TIMES, i.e. the total discounted system cost (this ensures that the second phase solution produce an economically meaningful set of values for the dual variables.)

In addition, in the second phase the user can specify bounds on fractional deviations in the LHS values of any or all user constraints, in comparison to the optimal LHS values obtained in the first phase. Such deviation bounds can be set for both global and non-global constraints, and for both non-constraining and constrained UCs (however, any original absolute bounds are overridden by the deviation bounds). The ***objective function***

used in Phase 1 is also available as an additional pre-defined UC, named '***OBJ1***', so that one can set either deviation bounds or absolute bounds on that as well, if desired. In addition, both the total and regional original objective functions can be referred to by using the predefined UC name '***OBJZ***' in the deviation bound parameters.

The objective function to be minimized in the second phase, and the additional bounds on the LHS values of UCs, can be written as follows:

$$\min objz \;=\; LHS('OBJZ')$$

$$\left.\begin{aligned} LHS(uc) &\leq (1 + maxdev(uc)) \bullet LHS^*(uc) \\ LHS(uc) &\geq (1 - maxdev(uc)) \bullet LHS^*(uc) \end{aligned}\right\} \begin{array}{l} \text{for each } uc \text{ for which} \\ maxdev(uc) \text{ has been specified} \end{array}$$

where:

LHS('OBJZ')	=	the standard objective function (discounted total system costs)
LHS(uc)	=	LHS expression of user constraint *uc* according to its definition
LHS(uc)*	=	optimal LHS value of user constraint *uc* in Phase 1
maxdev(uc)	=	user-specified fraction defining the maximum proportional deviation in the value of LHS(uc) compared to the solution in Phase 1

Remarks:

1. Use of the two-phase tradeoff analysis facility requires that a weight has been defined for at least one objective component in the first phase.
2. If no deviation bounds are specified, the second phase will be omitted.
3. Automatic discounting of any commodity or flow-based UC component is possible by using a new UC_ATTR option 'PERDISC' which could be applied e.g. to the user-defined objective components in Phase 1.
4. The two-phase tradeoff analysis can be carried over a set of distinct cases, each identified by a unique SOW index.

6.2 *Multiphase tradeoff analysis*

The multiphase tradeoff analysis is otherwise similar to the two-phase analysis, but in this case the objective function can be defined in the same way as in the Phase 1 described above also in all subsequent phases. The different objective functions in each phase are distinguished by using an additional phase index (the SOW index). Deviation bounds can be specified in each phase, such that they will be in force over all subsequent phases (any user constraints), or only in some of the succeeding phases (any user constraints excluding OBJ1). The deviation bounds defined on any of the user-defined objectives OBJ1 will thus always be preserved over all subsequent phases.

Remark: Although the multiphase tradeoff analysis allows the use of any user-defined objective functions in each phase, it is highly recommended that the original objective function be used in the last phase, so that the economic meaning is maintained in the final solution.

The procedure was presented in a very general form, in order to let the user exert her ingenuity at will. Typical simple examples of using the feature may be useful.

Example 1: trade-off between cost and risk.

First, a special UC (call it RISK) is defined that expresses a **global risk** measure. The successive phases consist in minimizing the following parameterized objective:

$$Min\ OBJZ + \alpha \cdot RISK$$

where α is a user chosen coefficient that may be varied within a range to explore an entire trade-off curve such as illustrated in figure 6.1, where the vertical axis represents the values of the cost objective function, and the horizontal axis the risk measure.

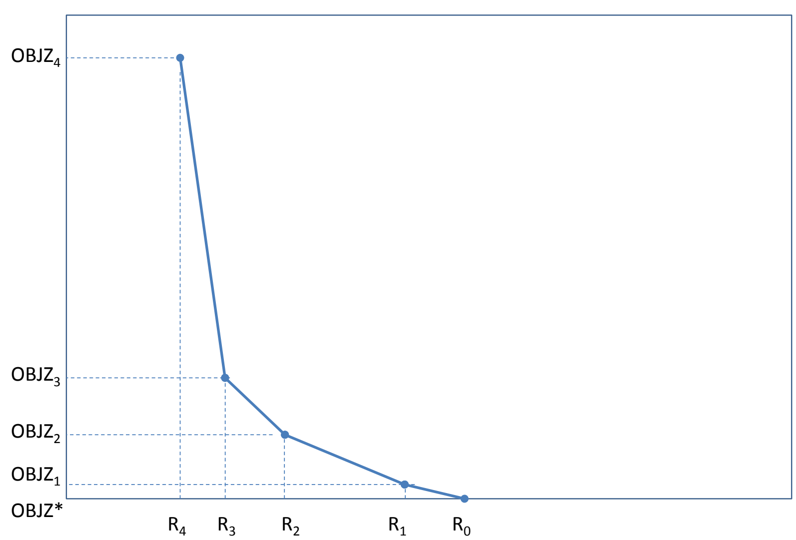


Figure 6.1. Trade-off between Risk and Cost

OBJZ* is the lowest value for OBJZ, corresponding to a relatively large value R_0 for RISK, i.e. when $\alpha = 0$. As α increases, RISK decreases and OBJZ increases. In this example, 4 alternate values of α were chosen until the value of OBJZ becomes very large, at point (R_4,$OBJZ_4$). This would correspond to very large value for α, i.e. a point where RISK is minimized.

An example of such an analysis is fully developed in Kanudia et al (2013), where a risk index is constructed to capture an indicator of energy security for the European Union. A complex (but linear) risk measure was developed to evaluate the risk for a large number of alternative channels of energy imports into the EU, and the trade-off between risk and overall cost was explored.

Example 2: exploring the opportunity cost of the nuclear option

At phase 1, the original *OBJZ* is minimized with the habitual TIMES constraints. This results in an optimal cost *OBJZ**. At phase 2, the objective function is equal to the total nuclear capacity over the entire horizon and over all regions, and a new constraint is added as follows:

$$OBJZ \ \leq (1+\alpha) \cdot OBJZ^*$$

The α parameter may be varied to explore the entire trade-off curve. A last phase may also be added at the end, with OBJZ as objective function, and a user selected value for the maximum level of nuclear capacity.

7 The TIMES Climate Module

This chapter provides a detailed description of the theoretical approach taken to model changes in atmospheric greenhouse gas concentrations, radiative forcing, and global mean temperatures in the TIMES Climate Module. Appendix A of Part II contains a full description of the implementation of the Climate Module in TIMES, including parameters, variables, and equations, as represented in the TIMES code.

The Climate Module starts from global emissions of CO2, CH4, and N2O, as generated by the TIMES global model, and proceeds to compute successively:

- the changes in CO2, CH4, and N2O concentrations via three separate sets of equations;
- the total change (over pre-industrial times) in atmospheric radiative forcing resulting from the three gases plus an exogenously specified additional forcing resulting from other causes (other anthropogenic and/or natural causes, as defined by the user), and
- the temperature changes (over pre-industrial times) in two reservoirs (surface and deep ocean).

The climate equations used to perform these calculations were adapted from Nordhaus and Boyer (1999), who proposed a three reservoir model for the CO2 cycle only[32]. This leads to linear recursive equations for calculating CO2 concentrations in each reservoir. The temperature equations use a two-reservoir model leading also to linear equations. The forcing equation is the one used in most climate models, and is non-linear.

In TIMES, we have modeled separately the life cycles of two other GHG's besides CO2, namely methane and nitrous oxide. These linear equations give results that are good approximations of those obtained from more complex climate models (Drouet et al., 2004; Nordhaus and Boyer, 1999).

The non-linear radiative forcing equation used in virtually all climate models was replaced in TIMES by a linear approximation whose values closely approach the exact ones as long as the useful range is carefully selected. This was done in order to keep the

[32]Other important GHG's such as CH4 and N2O may either be expressed in CO2-equivalent, or a special exogenous forcing term may be added to CO2 forcing. The latter approach is not attractive as it keeps two major GHG's fully exogenous.

entire model linear, and therefore to allow the user to set constraints on forcing and on temperature as well as on concentrations and on emissions.

The temperature equations have been kept as in Nordhaus and Boyer.

We now describe the mathematical equations used at each of the three steps of the climate module.

7.1 *Concentrations (accumulation of CO2, CH4, N2O*[33]*)*

a) CO2 accumulation is represented as the linear three-reservoir model below: the atmosphere, the quickly mixing upper ocean + biosphere, and the deep ocean. CO2 flows in both directions between adjacent reservoirs. The 3-reservoir model is represented by the following 3 equations when the step of the recursion is equal to one year:

$$M_{atm}(y) = E(y) + (1 - \varphi_{atm\text{-}up})\, M_{atm}(y\text{-}1) + \varphi_{up\text{-}atm}\, M_{up}(y\text{-}1) \qquad (7\text{-}1)$$

$$M_{up}(y) = (1 - \varphi_{up\text{-}atm} - \varphi_{up\text{-}lo})\, M_{up}(y\text{-}1) + \varphi_{atm\text{-}up}\, M_{atm}(y\text{-}1) + \varphi_{lo\text{-}up}\, M_{lo}(y\text{-}1) \qquad (7\text{-}2)$$

$$M_{lo}(y) = (1 - \varphi_{lo\text{-}up})\, M_{lo}(y\text{-}1) + \varphi_{up\text{-}lo}\, M_{up}(y\text{-}1) \qquad (7\text{-}3)$$

with

- $M_{atm}(y)$, $M_{up}(y)$, $M_{lo}(y)$: Concentration (expressed in mass units) of CO_2 in atmosphere, in a quickly mixing reservoir representing the upper level of the ocean and the biosphere, and in deep oceans (GtC), respectively, in year y (GtC)
- $E(y) = CO_2$ emissions in year y (GtC)
- φ_{ij}, transport rate from reservoir i to reservoir j (i, j = atm, up, lo) from year y-1 to y

b) CH4 accumulation is represented by a so-called single-box model in which the atmospheric methane concentration obeys the following equations assuming a constant annual decay rate of the anthropogenic concentrations Φ_{CH4} (whereas the natural concentration is assumed in equilibrium):

$$CH4_{atm}(y) = (1 - \Phi_{CH4}) \cdot CH4_{atm}(y-1) + EA_{CH4}(y) \qquad (7-4)$$

$$CH4_{up}(y) = CH4_{up}(y-1) \qquad (7-5)$$

[33] In keeping with the literature, we have expressed all concentrations as masses in megatonnes.

$$CH4_{tot}(y) = CH4_{atm}(y) + CH4_{up}(y) \quad (7 - 6)$$

where

- $CH4_{atm}$,, $CH4_{up}$,$CH4_{tot}$, , and EA_{CH4} are respectively: the anthropogenic atmospheric concentration, the natural atmospheric concentration[34], the total atmospheric concentration (all three expressed in Mt), and the anthropogenic emission of CH4 (expressed in Mt/yr). The anthropogenic emissions EA_{CH4} are generated within the model and enter the dynamic equation (7-4) in order to derive the anthropogenic concentration. Note that the natural concentration $CH4_{up}$is constant at all times. (See initial values for these and other parameters in Part II, Appendix A.)
- $CH4_{tot}$ is then reported and used in the forcing equations. All quantities are indexed by year.
- $1-\Phi_{CH4}$is the one-year retention rate of CH4 in the atmosphere.
- $d_{CH4} = 2.84$ (the density of *CH4,* expressed in *Mt/ppbv*) is then used to convert concentration in Mt into ppbv for reporting purposes.

c) N2O accumulation is also represented by a single-box model in a manner entirely similar to CH4, although with different parameter values. The corresponding equations are as follows:

$$N2O_{atm}(y) = (1-\Phi_{N2O}) \cdot N2O_{atm}(y-1) + EA_{N2O}(y)$$

$$N2O_{up}(y) = N2O_{up}(y-1)$$

$$N2O_{tot}(y) = N2O_{up}(y) + N2O_{atm}(y)$$

7.2 ***Radiative forcing***

We assume, as is routinely done in atmospheric science, that the atmospheric radiative forcings caused by the various gases are additive (IPCC, 2007). Thus:

---

[34]Note that the subscripts *atm* and *up*, which for the CO2 equations referred to the atmosphere and upper reservoirs, have been reused for the CH4 and N2O equations to stand for anthropogenic and natural concentrations.

$$\Delta F(y) = \Delta F_{CO2}(y) + \Delta F_{CH4}(y) + \Delta F_{N2O}(y) + EXOFOR(y) \qquad (7-7)$$

We now explain these four terms.

a) The relationship between CO2 accumulation and increased radiative forcing, $\Delta F_{CO2}(y)$, is derived from empirical measurements and climate models (IPCC 2001 and 2007).

$$\Delta F_{CO2}(y) = \gamma * \frac{\ln(M_{atm}(y)/M_0)}{\ln 2}$$

where:

- M_0 (i.e.CO2ATM_PRE_IND) is the pre-industrial (circa 1750) reference atmospheric concentration of CO2 = 596.4 GtC
- γ is the radiative forcing sensitivity to atmospheric CO_2 concentration doubling = 3.7 W/m^2

b) The radiative forcing due to atmospheric CH4 is given by the following expression (IPCC 2007), where the subscript *tot* has been omitted

$$\Delta F_{CH4}(y) = 0.036 \cdot \left(\sqrt{CH4_y} - \sqrt{CH4_0}\right) - \left[f(CH4_y, N2O_0) - f(CH4_0, N2O_0)\right] \qquad (7-8)$$

c) The radiative forcing due to atmospheric N2O is given by the following expression (IPCC, 2007)

$$\Delta F_{N2O}(y) = 0.12 \cdot \left(\sqrt{N2O_y} - \sqrt{N2O_0}\right) - \left[f(CH4_0, N2O_y) - f(CH4_0, N2O_0)\right] \qquad (7-9)$$

where:

$$f(x, y) = 0.47 \cdot \ln\left[1 + 2.01 \cdot 10^{-5} \cdot (xy)^{0.75} + 5.31 \cdot 10^{-15} \cdot x(xy)^{1.52}\right] \qquad (7-10)$$

Note that the f(x,y) function, which quantifies the cross-effects on forcing of the presence in the atmosphere of both gases (CH4 and N2O), is not quite symmetrical in the two gases. As usual, the 0 subscript indicates the pre-industrial times (1750).

d) EXOFOR(y) is the increase in total radiative forcing at period t relative to pre-industrial level due to GHG's that are not represented explicitly in the model. Units = W/m^2. In Nordhaus and Boyer (1999), only emissions of CO2 were explicitly modeled, and therefore EXOFOR(y) accounted for all other GHG's. In TIMES, N_2O and CH_4 are fully accounted for, but some other substances are not (e.g. CFC's, aerosols, ozone, volcanic activity, etc.). Therefore, the values for EXOFOR(y) will differ from those in Nordhaus and Boyer (1999). It is the modeler's responsibility to include in the calculation of EXOFOR(y) the forcing from only those gases and other causes that are not modeled. The careful modeler may also want to adapt the EXOFOR trajectory to particular scenarios. This has been done using alternative trajectories for EXOFOR provided by other models, as was done in a multi-model, multi-scenario study conducted at the Energy Modeling Forum (Clarke et al., 2009)

The parameterization of the three forcing equations (7-8, 7-9, and 7-10) is not controversial and relies on the results reported by Working Group I of the IPCC. IPCC (2001, Table 6.2, p.358) provides a value of 3.7 for γ, smaller than the one used by Nordhaus and Boyer ($\gamma = 4.1$). We have adopted this lower value of 3.7 W/m^2 as default in TIMES. Users are free to experiment with other values of the γ parameter. The same reference provides the entire expressions for all three forcing equations.

7.3 *Linear approximations of the three forcings*

In TIMES, each of the three forcing expressions is replaced by a linear approximation, in order to preserve linearity of the entire model. All three forcing expressions are concave functions. Therefore, two linear approximations are obvious candidates. The first one is an approximation from below, consisting of the chord of the graph between two selected end-points. The second one has the same slope as the chord and is tangent to the graph, thus approximating the function from above. The final approximation is the arithmetic average of the two approximations. These linear expressions are easily derived once a range of interest is defined by the user.

As an example, we derive below the linear approximation for the CO2 forcing expression. The other approximations are obtained in a similar manner.

Linear approximation for the CO2 forcing expression (see technical note "TIMES Climate Module" for similar approximations of the other two forcings):

First, an interval of interest for the concentration M must be selected by the user. The interval should be wide enough to accommodate the anticipated values of the concentrations, but not so wide as to make the approximation inaccurate. We denote the interval (M_1,M_2).

Next, the linear forcing equation is taken as the half sum of two linear expressions, which respectively underestimate and overestimate the exact forcing value. The underestimate consists of the chord of the logarithmic curve, whereas the overestimate consists of the tangent to the logarithmic curve that is parallel to the chord. These two estimates are illustrated in Figure 7.1, where the interval (M_1,M_2) is from 375 ppm to 550 ppm.

By denoting the pre-industrial concentration level as M_0, the general formulas for the two estimates are as follows:

Overestimate:

$$F_1(M) = \frac{\gamma}{\ln 2} \cdot \left[\ln\left(\frac{\gamma}{slope \cdot \ln(2) \cdot M_0}\right) - 1 \right] + slope \cdot M$$

Underestimate:

$$F_2(M) = \gamma \cdot \ln(M_1 / M_0) / \ln 2 + slope \cdot (M - M_1)$$

where:

$$slope = \gamma \cdot \frac{\ln(M_2 / M_1) / \ln 2}{(M_2 - M_1)}$$

Final approximation:

$$F_3(M) = \frac{F_1(M) + F_2(M)}{2}$$

7.4 *Temperature increase*

In the TIMES Climate Module as in many other integrated models, climate change is represented by the global mean surface temperature. The idea behind the two-reservoir model is that a higher radiative forcing warms the atmospheric layer, which then quickly warms the upper ocean. In this model, the atmosphere and upper ocean form a single layer, which slowly warms the second layer consisting of the deep ocean.

$\Delta T_{up}(y) = \Delta T_{up}(y\text{-}1) + \sigma_1\{F(y) - \lambda \Delta T_{up}(y\text{-}1) - \sigma_2 [\Delta T_{up}(y\text{-}1) - \Delta T_{low}(y\text{-}1)]\}$ (7-11)
$\Delta T_{low}(y) = \Delta T_{low}(y\text{-}1) + \sigma_3[\Delta T_{up}(y\text{-}1) - \Delta T_{low}(y\text{-}1)]$ (7-12)

with

- ΔT_{up} = globally averaged surface temperature increase above pre-industrial level,

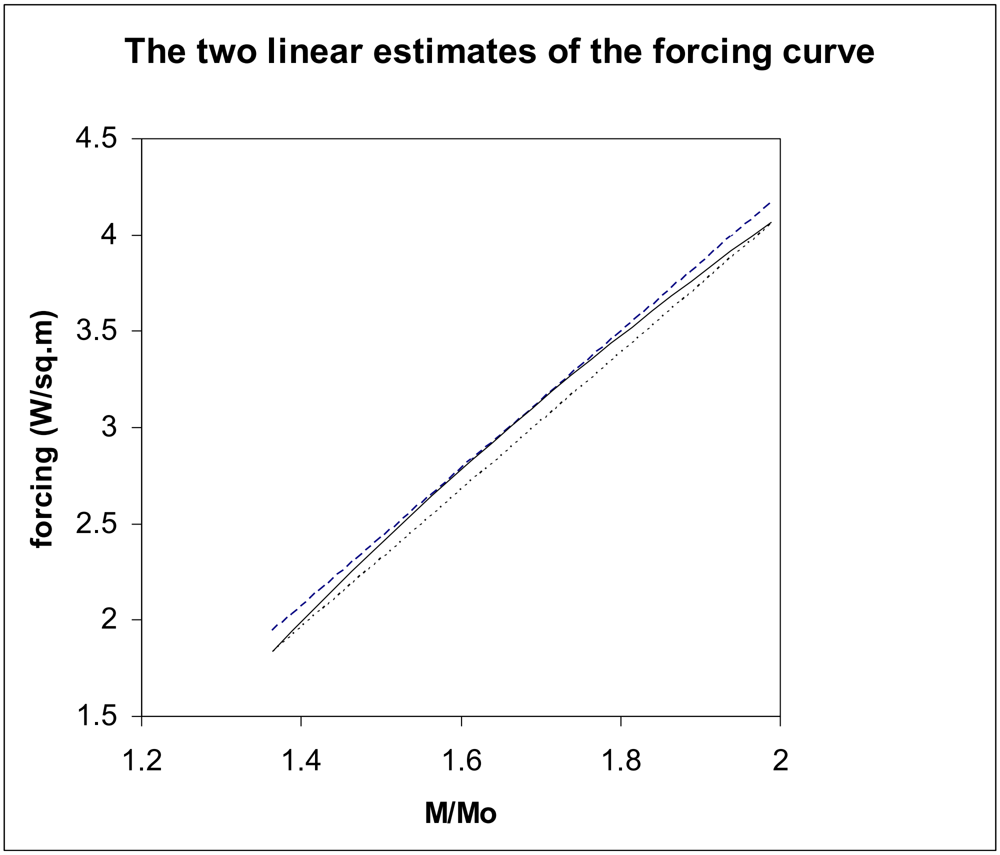


Figure 7.1. Illustration of the linearization of the CO2 radiative forcing function

- ΔT_{low}= deep-ocean temperature increase above pre-industrial level,
- σ_1= 1-year speed of adjustment parameter for atmospheric temperature (also known as the lag parameter),
- σ_2= coefficient of heat loss from atmosphere to deep oceans,
- σ_3 = 1-year coefficient of heat gain by deep oceans,
- λ= feedback parameter (climatic retroaction). It is customary to write λ as $\lambda = \gamma/C_s$, C_s being the climate sensitivity parameter, defined as the change in equilibrium atmospheric temperature induced by a doubling of CO_2 concentration. In contrast with most other parameters, the value of C_s is highly uncertain, with a possible range of values from 1°C to 10°C. This parameter is therefore a prime candidate for sensitivity analysis, or for treatment by probabilistic methods such as stochastic programming.

For more details on the implementation of the Climate Module in TIMES, including parameters, variables, and equations, as represented in the TIMES code, see Appendix A of Part II.

8 The Stochastic Programming extension

8.1 *Preamble to chapters 8 to 11*

Recall that the core TIMES paradigm described in chapters 3, 4, and 5 makes several basic assumptions:

- Linearity of the equations and objective function
- Perfect foresight of all agents over the entire horizon
- Competitive markets (i.e. no market power by any agent)

If any or all of these assumptions are violated, the properties of the resulting equilibrium are no longer entirely valid. In the following four chapters, we present four variants of the TIMES paradigm that depart from the core model. Each of these variants (extensions) departs from one or more assumptions above, as follows:

- Stochastic Programming TIMES extension: departs from the perfect foresight assumption and instead assumes that certain key model parameters are random. This extension requires the use of stochastic programming rather than the usual deterministic linear programming algorithm;

- Limited horizon TIMES extension: departs from the perfect foresight assumption and replaces it by an assumption of limited (in time) foresight. This extension requires the use of sequential linear programming rather than a single global linear optimization;

- Lumpy investments extension: departs from the linearity assumption and replaces it by the assumption that certain investments may only be made in discrete units rather than in infinitely divisible quantities. This extension requires the use of mixed integer programming (MIP) instead of Linear programming;

- The endogenous technological learning (ETL) extension: departs from the linearity assumption for the cost of technologies and replaces it by an assumption that the costs of some technologies are decreasing functions of the cumulative amounts of the technologies, i.e. a learning curve is assumed. This entails that some parts of the objective function are non-linear and non-convex, and requires the use of MIP.

Remark: None of these four extensions departs from the competitive market assumption. It is *also* possible to simulate certain types of non-competitive behavior using TIMES. For instance, it has been possible to simulate the behavior of the OPEC oil cartel by assuming that OPEC imposes an upper limit on its oil production in order to increase its long term profit (Loulou et al, 2007). Such uses of TIMES are not embodied in new extensions. Rather, they are left to the ingenuity of the user.

8.2 *Stochastic Programming concepts and formulation*

Stochastic Programming is a method for making optimal decisions under risk. The risk consists of facing uncertainty regarding the values of some (or all) of the LP parameters (cost coefficients, matrix coefficients, RHSs). Each uncertain parameter is considered to be a random variable, usually with a discrete, known probability distribution. The objective function thus becomes also a random variable and a criterion must be chosen in order to make the optimization possible. Such a criterion may be expected cost, expected utility, etc., as mentioned by Kanudia and Loulou (1998). Technical note "TIMES-Stochastic" provides a more complete description of the TIMES implementation

Uncertainty on a given parameter is said to be resolved, either fully or partially, at the *resolution time*, i.e. the time at which the actual value of the parameter is revealed. Different parameters may have different times of resolution. Both the resolution times and the probability distributions of the parameters may be represented on an event tree, such as the one of figure 8.1, depicting a typical energy/environmental situation. In figure 8.1, two parameters are uncertain: mitigation level, and demand growth rate. The first may have only two values (High and Low), and becomes known in 2010. The second also may have two values (High and Low) and becomes known in 2020. The probabilities of the outcomes are shown along the branches. This example assumes that present time is 2000. This example is said to have three stages (i.e. two resolution times). The simplest non-trivial event tree has only two stages (a single resolution time). Each pathway along the event tree, representing a different realization of the uncertain parameters is referred to as a state-of-the-world (SOW).

The **key observation** is that prior to resolution time, the decision maker (and hence the model) does not know the eventual values of the uncertain parameters, but still has to take decisions. On the contrary, after resolution, the decision maker knows with certainty the outcome of some event(s) and his subsequent decisions will be different depending on which outcome has occurred.

For the example shown in figure 8.1, in 2000 and 2010 there can be only one set of decisions, whereas in 2020 there will be two sets of decisions, contingent on which of the mitigation outcomes (High or Low) has occurred, and in 2030, 2040, 2050 and 2060, there will be four sets of contingent decisions.

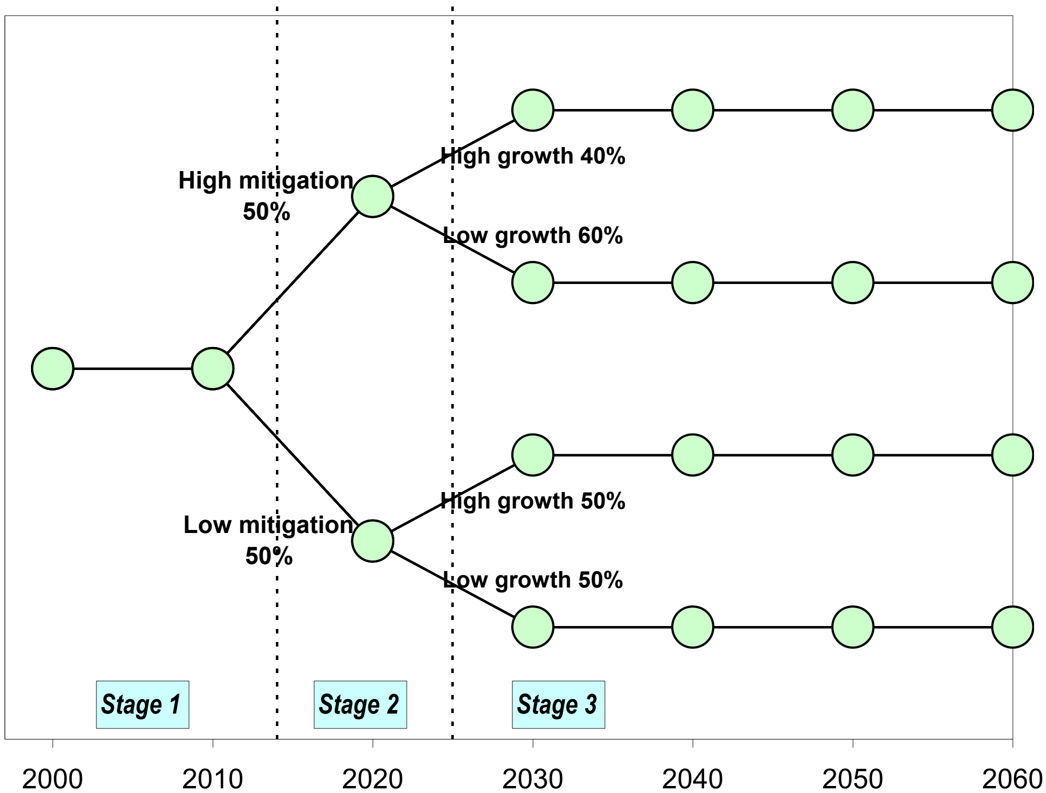


Figure 8.1. Event Tree for a three-stage stochastic TIMES Example.

This remark leads directly to the following general multi-period, multi-stage stochastic program in Equations 8-1 to 8-3 below. The formulation described here is based on Dantzig (1963, Wets (1989), or Kanudia and Loulou (1999), and uses the expected cost criterion. Note that this is a LP, but its size is much larger than that of the deterministic TIMES model.

Minimize

$$Z = \sum_{t \in T} \sum_{s \in S(t)} C(t,s) \times X(t,s) \times p(t,s) \qquad (8-1)$$

Subject to:

$$A(t,s) \times X(t,s) \geq b(t,s) \;\; \forall s \in S(T), t \in T \qquad (8\text{-}2)$$

$$\sum_{t \in T} D(t, g(t,s)) \times X(t, g(t,s)) \geq e(s) \quad \forall s \in S(T) \qquad (8\text{-}3)$$

where

t	=	time period
T	=	set of time periods
s	=	state index
S(t)	=	set of state indices for time period t;

For Figure 8.1, we have: S(2000) = 1; S(2010) = 1; S(2020) = 1,2; S(2030) = 1,2,3,4; S(2040) = 1,2,3,4; S(2050) = 1,2,3,4; S(2060) = 1,2,3,4;

S(*T*)	=	set of state indices at the last stage (the set of *scenarios*). Set *S(T)* is homeomorphic to the set of paths from period 1 to last period, in the event tree.
g(*t,s*)	=	a unique mapping from $\{(t,s) \mid s \in S(T)\}$ to *S*(*t*), according to the event tree. *g(t,s)* is the state at period *t* corresponding to scenario *s*.
X(*t,s*)	=	the column vector of decision variables in period t, under state s
C(*t,s*)	=	the cost row vector
p(*t,s*)	=	event probabilities
A(*t,s*)	=	the LP sub-matrix of single period constraints, in time period t, under state s
b(*t,s*)	=	the right hand side column vector (single period constraints) in time period t, under state s
D(*t,s*)	=	the LP sub-matrix of multi-period constraints under state s
e(*s*)	=	the right hand side column vector (multi-period constraints) under scenario s

Alternate formulation: The above formulation makes it a somewhat difficult to retrieve the strategies attached to the various scenarios. Moreover, the actual writing of the cumulative constraints (8-3) is a bit delicate. An alternate (but equivalent) formulation consists in defining one scenario per path from initial to terminal period, and to define distinct variables *X(t,s)* for each scenario and each time period. For instance, in this alternate formulation of the example, there would be four variables *X(t,s)* at every period t, (whereas there was only one variable X(2000,1) in the previous formulation).

Minimize

$$Z = \sum_{t \in T} \sum_{s \in S(t)} C(t,s) \times X(t,s) \times p(t,s) \qquad (8-1)'$$

Subject to:

$$A(t,s) \times X(t,s) \geq b(t,s) \qquad all\ t,\ all\ \mathrm{s} \qquad (8—2)'$$

$$\sum_{t \in T} D(t,s) \times X(t,s) \geq e(s) all\ t,\ all\ \mathrm{s} \qquad (8—3)'$$

Of course, in this approach we need to add equality constraints to express the fact that some scenarios are identical at some periods. In the example of Figure 8.1, we would have:

X(2000,1)=X(2000,2)=X(2000,3)=X(2000,4),
X(2010,1)=X(2010,2)=X(2010,3)=X(2010,4),
X(2020,1)=X(2020,2),
X(2020,3)=X(2020,4).

Although this formulation is less parsimonious in terms of additional variables and constraints, many of these extra variables and constraints are in fact eliminated by the pre-processor of most optimizers. The main advantage of this new formulation is the ease of producing outputs organized by scenario.

In the current implementation of stochastic TIMES, the first approach has been used (Equations 8-1 to 8-3). The results are however reported for all scenarios in the same way as in the second approach.

In addition, in TIMES there is also an experimental variant for the modeling of recurring uncertainties with stochastic programming, described in Appendix A of technical note “TIMES-Stochastic”.

8.3 *Alternative criteria for the objective function*

The preceding description of stochastic programming assumes that the policy maker accepts the expected cost as his optimizing criterion. This is equivalent to saying that he is risk neutral. In many situations, the assumption of risk neutrality is only an approximation of the true utility function of a decision maker.

Two alternative candidates for the objective function are:

- Expected utility criterion with linearized risk aversion

- Minimax Regret criterion (Raiffa,1968, applied in Loulou and Kanudia, 1999)

8.3.1 Expected utility criterion with risk aversion

The first alternative has been implemented into the stochastic version of TIMES. This provides a feature for taking into account that a decision maker may be risk averse, by defining a new utility function to replace the expected cost.

The approach is based on the classical E-V model (an abbreviation for Expected Value-Variance). In the E-V approach, it is assumed that the variance of the cost is an acceptable measure of the risk attached to a strategy in the presence of uncertainty. The variance of the cost of a given strategy k is computed as follows:

$$\boldsymbol{Var(C_k) = \sum_j p_j \bullet (Cost_{j|k} - EC_k)^2}$$

where $\boldsymbol{Cost_{j|k}}$ is the cost when strategy $\boldsymbol{k}$ is followed and the $\boldsymbol{j^{th}}$ state of nature prevails, and $\boldsymbol{EC_k}$ is the expected cost of strategy $\boldsymbol{k}$**,** defined as usual by:

$$\boldsymbol{EC_k = \sum_j p_j \bullet Cost_{j|k}}$$

An E-V approach would thus replace the expected cost criterion by the following utility function to minimize:

$$\boldsymbol{U = EC + \lambda \cdot \sqrt{Var(C)}}$$

where $\lambda>0$ is a measure of the risk aversion of the decision maker. For $\lambda=0$, the usual expected cost criterion is obtained. Larger values of $\boldsymbol{\lambda}$ indicate increasing risk aversion.

Taking risk aversion into account by this formulation would lead to a non-linear, non-convex model, with all its ensuing computational restrictions. These would impose serious limitations on model size.

8.3.2 Utility function with linearized risk aversion

To avoid non-linearities, it is possible to replace the semi-variance by the upper-absolute-deviation, defined by:

$$UpAbsDev\ (Cost_k) = \sum_j p_j \bullet \{Cost_{j|k} - EC_k\}^+$$

where $\boldsymbol{y} = \{\boldsymbol{x}\}^+$ is defined by the following two *linear* constraints: $\boldsymbol{y} \geq \boldsymbol{x}$, and $\boldsymbol{y} \geq \boldsymbol{0}$, and the utility is now written via the following *linear* expression:

$$\boldsymbol{U} = \boldsymbol{EC} + \lambda \cdot \boldsymbol{UpsAbsDev}(\boldsymbol{C})$$

This is the expected utility formulation implemented into the TIMES model generator.

8.4 ***Solving approaches***

General multi-stage stochastic programming problems of the type described above can be solved by standard deterministic algorithms by solving the deterministic equivalent of the stochastic model. This is the most straightforward approach, which may be applied to all problem instances. However, the resulting deterministic problem may become very large and thus difficult to solve, especially if integer variables are introduced, but also in the case of linear models with a large number of stochastic scenarios.

Two-stage stochastic programming problems can also be solved efficiently by using a Benders decomposition algorithm (Wets, 1989). Therefore, the classical decomposition approach to solving large multi-stage stochastic linear programs has been nested Benders decomposition. However, a multi-stage stochastic program with integer variables does not, in general, allow a nested Benders decomposition. Consequently, more complex decompositions approaches are needed in the general case (e.g. Dantzig-Wolfe decomposition with dynamic column generation, or stochastic decomposition methods).

The current version of the TIMES implementation for stochastic programming is solely based on directly solving the equivalent deterministic problem. As this may lead to very large problem instances, stochastic TIMES models are in practice limited to a relatively small number of branches of the event tree (SOW's).

8.5 ***Economic interpretation***

The introduction of uncertainty alters the economic interpretation of the TIMES solution. Over the last two decades, economic modeling paradigms have evolved to a class of equilibria called Dynamic Stochastic General Equilibria (DSGE, see references Chen and Crucuni, 2012; de Walque et al., 2005; Smets et al., 2007). In the case of Stochastic TIMES, we are in the presence of a Dynamic Stochastic Partial Equilibria (DSPE), with a much less developed literature. The complete characterization of a DSPE is beyond the scope of this documentation, but it is useful to note some of its properties, which derive from the theory of Linear Programming, as follows:

- During the first stage (i.e. before resolution of any uncertainties), the meaning of the primal solution is identical to that of a deterministic TIMES run, i.e. of a set of optimal decisions, whereas the meaning of the shadow prices is that of *expected prices*(resp. expected marginal utility changes) of the various commodities. This is so because the shadow price is the marginal change in objective function when a commodity's balance is marginally altered, and the objective function is an expected cost (resp. an expected utility function).

- During subsequent stages, the primal values of any given branch of the event tree represent the optimal decisions *conditional on the corresponding outcome being true*, and the shadow prices are the *expected*[35] *prices* of the commodities also conditional on the corresponding outcome being true.

[35] The expected prices become deterministic prices if the stage is the last one, so that there is no uncertainty remaining at or after the current period.

9 Using TIMES with limited foresight (time-stepped)

It may be useful to simulate market conditions where all agents take decisions with only a limited foresight of a few years or decades, rather than the very long term. By so doing, a modeler may attempt to simulate "real-world" decision making conditions, rather than socially optimal ones. Both objectives are valid provided the modeler is well aware of each approach's characteristics.

Be that as it may, it is possible to use TIMES in a series of time-stepped runs, each with an optimizing horizon shorter than the whole horizon. The option that enables this mode is named FIXBOH, which freezes the solution over some user chosen years, while letting the model optimize over later years. The FIXBOH feature has several applications and is first described below before a full description of the time-stepped procedure.

9.1 *The FIXBOH feature*

This feature requires that an initial run be made first, and then FIXBOH sets fixed bounds for a subsequent run according to the solution values from the initial run up to the last milestone year less than or equal to the year specified by the FIXBOH control parameter. For instance, the initial run may be a reference case, which is run from 2010 to 2100, and the FIXBOH value might be set at 2015, in which case a subsequent run would have exactly the same solution values as the reference case up to 2015. This is an extremely convenient feature to use in most situations.

As a generalization to the basic scheme described above, the user can also request fixing to the previous solution different sets of fixed years according to region.

Example: Assume that you would like to analyze the 15-region ETSAP TIAM model with some shocks after the year 2030, and you are interested in differences in the model solution only in regions that have notable gas or LNG trade with the EU. Therefore, you would like to fix the regions AUS, CAN, CHI, IND, JPN, MEX, ODA and SKO completely to the previous solution, and all other regions to the previous solution up to 2030.

9.2 *The time-stepped option (TIMESTEP)*

The purpose of the TIMESTEP option is to run the model in a stepwise manner with limited foresight. The TIMESTEP control variable specifies the number of years that should be optimized in each solution step. The total model horizon will be solved by a series of successive steps, so that in each step the periods to be optimized are advanced further in the future, and all periods before them are fixed to the solution of the previous step (using the FIXBOH feature). It is important that any two successive steps have one or more overlapping period(s), in order to insure overall continuity of the decisions between the two steps (in the absence of the overlap, decisions taken at step ***n*** would have no initial conditions and would be totally disconnected from step ***n-1*** decisions.)

Figure 9.1 illustrates the step-wise solution approach with a horizon of 8 periods and 6 successive optimization steps. Each step has a 2 period sub-horizon, and there is also an overlap of one period between a step and the next. More explicitly: at step 2, all period 2 variables are frozen at the values indicated in the solution of step 1, and period 3 is free to be optimized. At step 3, period 3 variables are frozen and period 4 is optimized, etc.

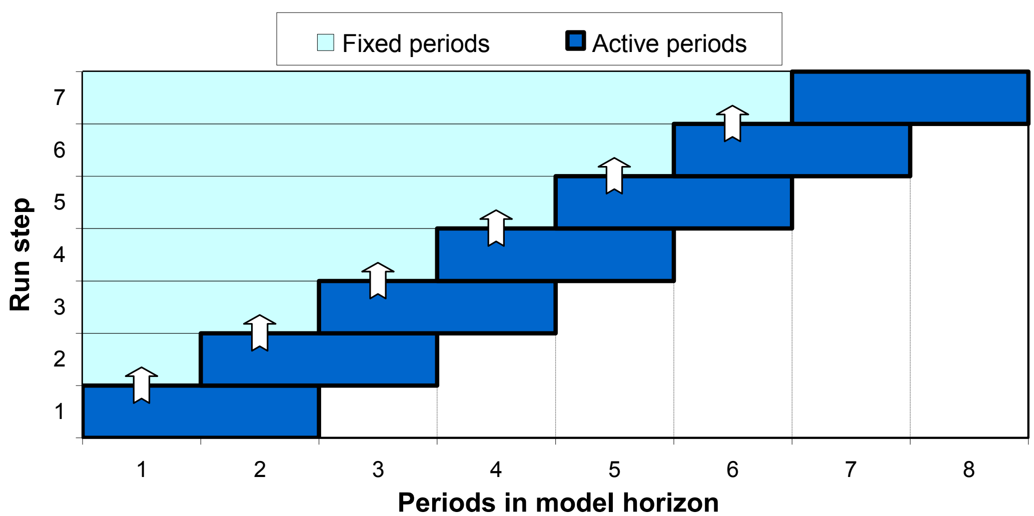


Figure 9.1. Sequence of optimized periods in the stepped TIMES solution approach. Each run includes also the fixed solution of all earlier periods.

The amount of overlapping years between successive steps is by default half of the active step length (the value of TIMESTEP), but it can be controlled by the user.

<ins>Important remark</ins>: as mentioned above, the user chooses the lengths of the sub-horizons and the length of the overlaps, *both expressed in years*. Because the time periods used in

the model may be variable and may not always exactly match with the step-length and overlap, the actual active step-lengths and overlaps may differ somewhat from the values specified by the user. At each step the model generator uses a heuristic that tries to make a best match between the remaining available periods and the prescribed step length. However, at each step it is imperative that at least one of the previously solved periods must be fixed, and at least one remaining new period is taken into the active optimization in the current step.

10 The Lumpy Investment extension

In some cases, the linearity property of the TIMES model may become a drawback for the accurate modeling of certain investment decisions. Consider for example a TIMES model for a relatively small community such as a city. For such a scope the *granularity* of some investments may have to be taken into account. For instance, the size of an electricity generation plant proposed by the model would have to conform to an implementable minimum size (it would make no sense to decide to construct a 50 MW nuclear plant). Another example for multi-region modeling might be whether or not to build cross-region electric grid(s) or gas pipeline(s) in discrete size increments. Processes subject to investments of only specific size increments are described as "lumpy" investments.

For other types of investments, size does not matter: for instance the model may decide to purchase 10,950.52 electric cars, which is easily rounded to 10,950 without any serious inconvenience, especially since this number is an annual figure. The situation is similar for a number of residential or commercial heating devices; or for the capacity of wind turbines; or of industrial boilers; in short, for any technologies with relatively small minimum feasible sizes. Such technologies would not be candidates for treatment as "lumpy" investments.

This chapter describes the basic concept and mathematics of lumpy investment option, whereas the implementation details are available in Part II, section 6.3.24. We simply note here that this option, while introducing new variables and constraints, does not affect existing TIMES constraints.

It is the user's responsibility to decide whether or not certain technologies should respect the minimum size constraint, weighing the pros and cons of so doing. This chapter explains how the TIMES LP is transformed into a Mixed Integer Program (MIP) to accommodate minimum or multiple size constraints, and states the consequences of so doing on computational time and on the interpretation of duality results.

The lumpy investment option available in TIMES is slightly more general than the one described above. It insures that investment in technology $\boldsymbol{k}$ is equal to one of a finite number $\boldsymbol{N}$ of pre-determined sizes: $\boldsymbol{0, S_1(t), S_2(t), \ldots, S_N(t)}$. This is useful when several typical plant sizes are feasible in the real world. As implied by the notation, these discrete sizes may be different at different time periods. Note that by choosing the $\boldsymbol{N}$ sizes as the successive multiples of a fixed number ***S***, it is possible to invest (perhaps many times) in a technology with fixed standard size.

Imposing such a constraint on an investment is unfortunately impossible to formulate using standard LP constraints and variables. It requires the introduction of *integer variables* into the formulation. The optimization problem resulting from the introduction of integer variables into a Linear Program is called a Mixed Integer Program (MIP).

10.1 *Formulation and solution of the Mixed Integer Linear Program*

Typically, the modeling of a lumpy investment involves Integer Variables, i.e. variables whose values may only be non-negative integers (0, 1, 2, …). The mathematical formulation is as follows:

$$VAR\_NCAP(p,t) = \sum_{i=1}^{N} S_i(p,t) \times Z_i(p,t) \qquad each\ t = 1,..,T$$

$$with$$

$$Z_i(p,t) = 0\ or\ 1$$

$$and$$

$$\sum_{i=1}^{N} Z_i(p,t) \le 1$$

The second and third constraints taken together imply that at most one of the ***Z*** variables is equal to 1 and all others are equal to zero. Therefore, the first constraint now means that ***NCAP*** is equal to one of the preset sizes or is equal to 0, which is the desired result.

Although the formulation of lumpy investments *looks* simple, it has a profound effect on the resulting optimization program. Indeed, MIP problems are notoriously more difficult to solve than LPs, and in fact many of the properties of linear programs discussed in the preceding chapters do not hold for MIPs, including duality theory, complementary slackness, etc. Note that the constraint that *Z(p,t)* should be 0 or 1 departs from the *divisibility* property of linear programs. This means that the *feasibility domain* of integer variables (and therefore of some investment variables) is no longer contiguous, thus making it vastly more difficult to apply purely algebraic methods to solve MIP's. In fact, practically all MIP solution algorithms make use (at least to some degree) of partial enumerative schemes, which tend to be time consuming and less reliable[36] than the algebraic methods used in LP.

The reader interested in more technical details on the solution of LPs and of MIPs is referred to references (Hillier and Lieberman, 1990, Nemhauser et al. 1989). In the next section we shall be content to state one important remark on the interpretation of the dual results from MIP optimization.

10.2 *Discrete early retirement of capacity*

The discrete retirement of capacity that was briefly mentioned in section 5.4.11 requires a treatment quite similar to that of discrete addition to capacity presented here. The complete mathematical formulation mimics that presented above, and is fully described in Part II, section 6.3.26, of the TIMES documentation.

10.3 *Important remark on the MIP dual solution (shadow prices)*

Using MIP rather than LP has an important impact on the interpretation of the TIMES shadow prices. Once the optimal MIP solution has been found, it is customary for MIP solvers to fix all integer variables at their optimal (integer) values, and to perform an additional iteration of the LP algorithm, so as to obtain the dual solution (i.e. the shadow prices of all constraints). However, the interpretation of these prices is different from that of a pure LP. Consider for instance the shadow price of the natural gas balance constraint: in a pure LP, this value represents the price of natural gas. In MIP, this value represents the price of gas *conditional on having fixed the lumpy investments at their optimal integer values.* What does this mean? We shall attempt an explanation via one example: suppose that one lumpy investment was the investment in a gas pipeline; then, *the gas shadow price will not include the investment cost of the pipeline, since that investment was fixed when the dual solution was computed*.

[36] A TIMES LP program of a given size tends to have fairly constant solution time, even if the database is modified. In contrast, a TIMES MIP may show some erratic solution times. One may observe reasonable solution times (although significantly longer than LP solution times) for most instances, with an occasional *very* long solution time for some instances. This phenomenon is predicted by the theory of complexity as applied to MIP, see Papadimitriou and Stieglitz (1982).

In conclusion, when using MIP, only the primal solution is fully reliable. In spite of this major caveat, modeling lumpy investments may be of paramount importance in some instances, and may thus justify the extra computing time and the partial loss of dual information.

11 The Endogenous Technological Learning extension

In a long-term dynamic model such as TIMES the characteristics of many of the future technologies are almost inevitably changing over the sequence of future periods due to *technological learning*.

In some cases it is possible to forecast such changes in characteristics as a function of time, and thus to define a time-series of values for each parameter (e.g. unit investment cost, or efficiency). In such cases, technological learning is *exogenous* since it depends only on time elapsed and may thus be established outside the model.

In other cases there is evidence that the pace at which some technological parameters change is dependent on the *experience* acquired with this technology. Such experience is not solely a function of time elapsed, but typically depends on the cumulative investment (often global) in the technology. In such a situation, technological learning is *endogenous*, since the future values of the parameters are no longer a function of time elapsed alone, but depend on the cumulative investment decisions taken by the model (which are unknown). In other words, the evolution of technological parameters may no longer be established outside the model, since it depends on the model's results.

Endogenous technological learning (ETL) is also named *Learning-By-Doing* (LBD) by some authors.

Whereas exogenous technological learning does not require any additional modeling, ETL presents a tough challenge in terms of modeling ingenuity and of solution time. In TIMES, there is a provision to represent the effects of endogenous learning on the unit investment cost of technologies. Other parameters (such as efficiency) are not treated, at this time.

11.1 *The basic ETL challenge*

Empirical studies of unit investment costs of several technologies have been undertaken in several countries. Many of these studies find an empirical relationship between the unit investment cost of a technology at time ***t***, $\boldsymbol{INVCOST_t}$, and the cumulative investment in that technology up to time ***t***, $C_t = \sum_{j=-1}^{t} VAR\_NCAP_j$.

A typical relationship between unit investment cost and cumulative investments is of the form:

$$INVCOST_t = a \cdot C_t^{-b} \qquad (11-1)$$

where

- *INVCOST*[37] is the unit cost of creating one unit of the technology, which is no longer a constant, but evolves as more units of the technology are produced;
- ***a*** is the value of I*NVCOST* for the first unit of the technology (when C_t is equal to 1) and;
- ***b*** is the learning index, representing the speed of learning[38].

As experience builds up, the unit investment cost decreases, potentially rendering investments in the technology more attractive. It should be clear that near-sighted investors will not be able to detect the advantage of investing early in learning technologies, since they will only observe the high initial investment cost and, being near-sighted, will not anticipate the future drop in investment cost resulting from early investments. In other words, tapping the full potential of technological learning requires far-sighted agents who accept making initially non-profitable investments in order to later benefit from the investment cost reduction.

With regard to actual implementation, simply using (11-1) as the objective function coefficient of VAR_NCAP_t will yield a non-linear, non-convex expression. Therefore, the resulting mathematical optimization is no longer linear, and requires special techniques for its solution. In TIMES, a Mixed Integer Programming (MIP) formulation is used, that we now describe.

11.2 *The TIMES formulation of ETL*

11.2.1 The cumulative investment cost

We follow the basic approach described in Barreto, 2001.

---

[37] The notation in this chapter is sometimes different from the standard notation for parameters and variables, in order to conform to the more detailed technical note on the subject.

[38] It is usual to define, instead of ***b***, another parameter, ***pr*** called the *progress ratio*, which is related to ***b*** via the following relationship: $pr = 2^{-b}$. Hence, ***1-pr*** is the cost reduction incurred when cumulative investment is doubled. Typical observed ***pr*** values are in a range of .75 to .95.

The first step of the formulation is to express the total investment cost, i.e. the quantity that should appear in the objective function. The cumulative investment cost $\boldsymbol{TC_t}$ of a learning technology in period $\boldsymbol{t}$ is obtained by integrating expression (11-1):

$$TC_t = \int_0^{C_t} a \cdot y^{-b} * dy = \frac{a}{1-b} \cdot C_t^{-b+1} \qquad (11-2)$$

$\boldsymbol{TC_t}$ is a concave function of $\boldsymbol{C_t}$, with a shape as shown in figure 11.1

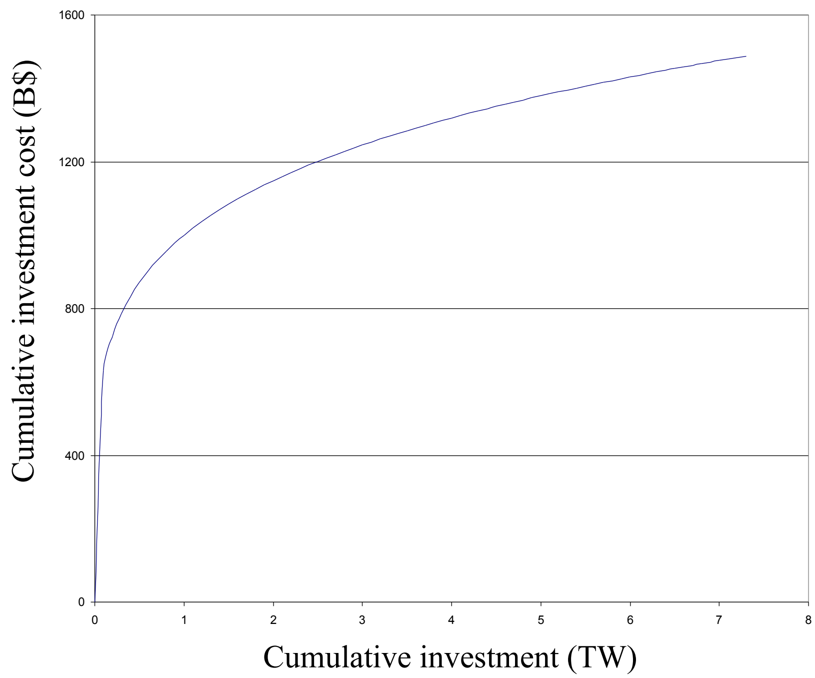


Figure 11.1. Example of a cumulative learning curve

With the Mixed Integer Programming approach implemented in TIMES, the cumulative learning curve is approximated by linear segments, and binary variables are used to represent some logical conditions. Figure 11.2 shows a possible piecewise linear approximation of the curve of Figure 11.1. The choice of the number of steps and of their respective lengths is carefully made so as to provide a good approximation of the smooth cumulative learning curve. In particular, the steps must be smaller for small values than for larger values, since the curvature of the curve diminishes as total investment increases. The formulation of the ETL variables and constraints proceeds as follows (we omit the period, region, and technology indexes for notational clarity):

1. The user specifies the set of learning technologies;
2. For each learning technology, the user provides:
    a) The progress ratio ***pr*** (from which the learning index ***b*** may be inferred)
    b) One initial point on the learning curve, denoted (C_0, TC_0)
    c) The maximum allowed cumulative investment C_{max} (from which the maximum total investment cost TC_{max} may be inferred)
    d) The number ***N*** of segments for approximating the cumulative learning curve over the (C_0, C_{max}) interval.

    Note that each of these parameters, including ***N*,** may be different for different technologies.
3. The model automatically selects appropriate values for the ***N*** step lengths, and then proceeds to generate the required new variables and constraints, and the new objective function coefficients for each learning technology. The detailed formulae are shown and briefly commented on below.

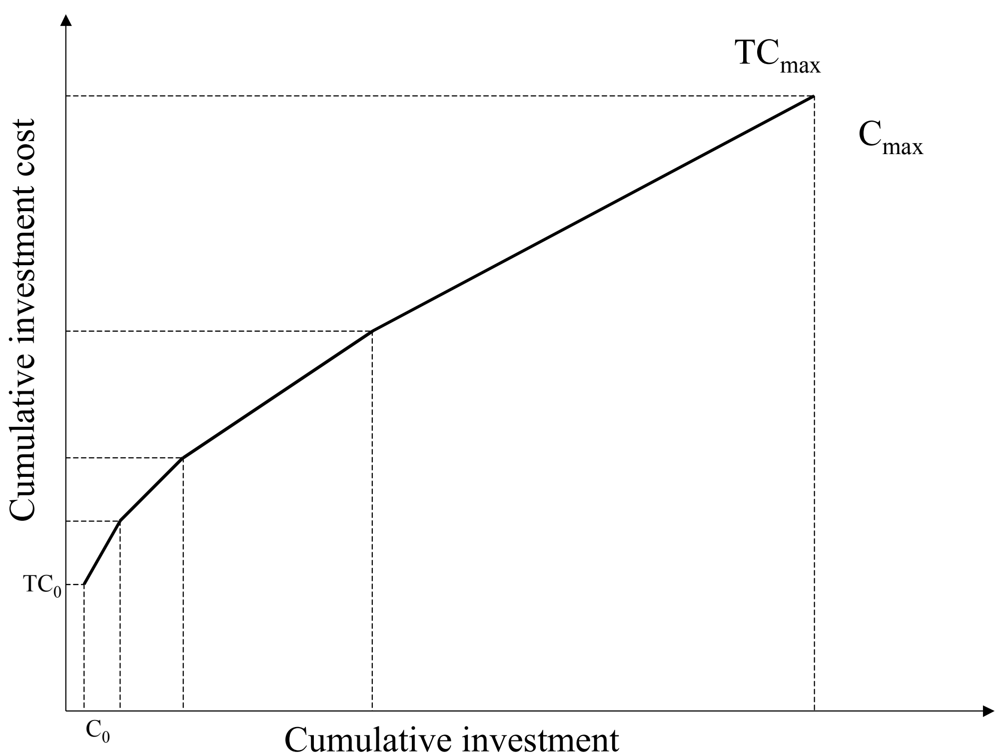


Figure 11.2. Example of a 4-segment approximation of the cumulative cost curve

11.2.2 Calculation of break points and segment lengths

The successive interval lengths on the vertical axis are chosen to be in geometric progression, each interval being twice as wide as the preceding one. In this fashion, the intervals near the low values of the curve are smaller so as to better approximate the curve in its high curvature zone. Let $\{TC_{i-1}, TC_i\}$ be the i^{th} interval on the vertical axis, for $i = 1, \ldots, N-1$. Then:

$$TC_i = TC_{i-1} + 2^{i-N-1}(TC_{\max} - TC_o)/(1-0.5^N)\ , \qquad i = 1,2,\cdots,N$$

Note that TC_{max} is equal to TC_N.

The break points on the horizontal axis are obtained by plugging the TC_i 's into expression (11-2), yielding:

$$C_i = \left(\frac{(1-b)}{a}(TC_i)\right)^{\frac{1}{1-b}} \quad , \qquad i = 1, 2, \ldots, N$$

11.2.3 New variables

Once intervals are chosen, standard approaches are available to represent a concave function by means of integer (0-1) variables. We describe the approach used in TIMES. First, we define N continuous variables x_i, $i = 1,\ldots,N$. Each x_i represents the portion of cumulative investments lying in the i^{th} interval. Therefore, the following holds:

$$C = \sum_{i=1}^{N} x_i \qquad 11-3$$

We now define N integer (0-1) variables z_i that serve as indicators of whether or not the value of C lies in the i^{th} interval. We may now write the expression for TC, as follows:

$$TC = \sum_{i=1}^{N} a_i z_i + b_i x_i \qquad 11-4$$

where b_i is the slope of the i^{th} line segment, and a_i is the value of the intercept of that segment with the vertical axis, as shown in figure 11.3. The precise expressions for a_i and b_i are:

$$b_i = \frac{TC_i - TC_{i-1}}{C_i - C_{i-1}} \qquad i = 1,2,\ldots,N$$

$$a_i = TC_{i-1} - b_i \cdot C_{i-1} \qquad i = 1,2,\ldots,N \qquad 11-5$$

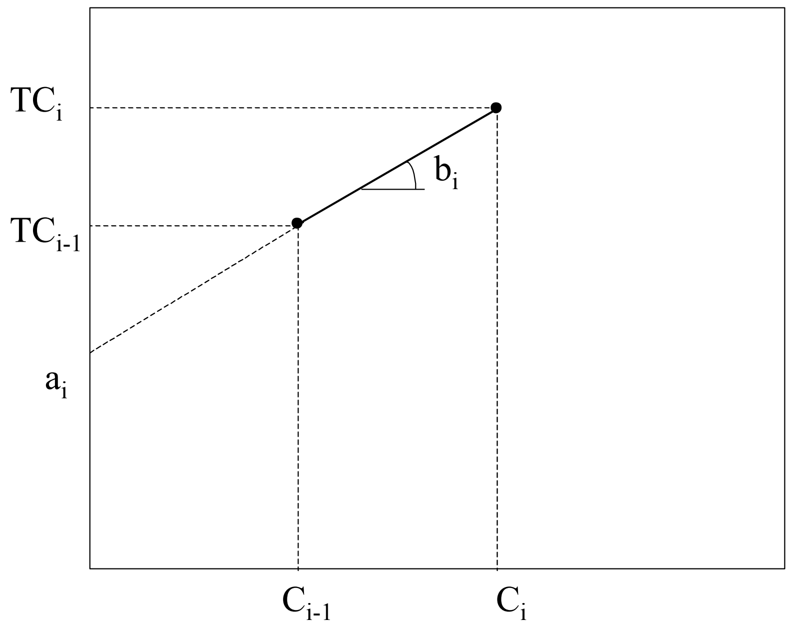


Figure 11.3. The i^{th} segment of the step-wise approximation

11.2.4 New constraints

For (11-4) to be valid we must make sure that exactly one $\boldsymbol{z_i}$ is equal to 1, and the others equal to 0. This is done (recalling that the $\boldsymbol{z_i}$ variables are 0-1) via:

$$\sum_{i=1}^{N} \boldsymbol{z_i} = 1$$

We also need to make sure that each $\boldsymbol{x_i}$ lies within the i^{th} interval whenever $\boldsymbol{z_i}$ is equal to 1 and is equal to 0 otherwise. This is done via two constraints:

$$\boldsymbol{C_{i-1} \cdot z_i \le x_i \le C_i \cdot z_i}$$

11.2.5 Objective function terms

Re-establishing the period index, we see that the objective function term at period ***t***, for a learning technology is thus equal to $\boldsymbol{TC_t - TC_{t-1}}$, which needs to be discounted like all other investment costs.

11.2.6 Additional (optional) constraints

Solving integer programming problems is facilitated if the domain of feasibility of the integer variables is reduced. This may be done via additional constraints that are not strictly needed but that are guaranteed to hold. In our application we know that experience (i.e. cumulative investment) is always increasing as time goes on. Therefore, if the cumulative investment in period t lies in segment i, it is certain that it will not lie in segments *i-1, i-2, .., 1* in time period *t+1*. This leads to two new constraints (re-establishing the period index ***t*** for the ***z*** variables):

$$\sum_{j=1}^{i} z_{j,t} \geq \sum_{j=1}^{i} z_{j,t+1}$$

$$i = 1,2,...,N-1,\ t = 1,2,...,T-1$$

$$\sum_{j=i}^{N} z_{j,t} \leq \sum_{j=i}^{N} z_{j,t+1}$$

Summarizing the above formulation, we observe that each learning technology requires the introduction of $\boldsymbol{N*T}$ integer (0-1) variables. For example, if the model has 10 periods and a 5-segment approximation is selected, 50 integer (0-1) variables are created for that learning technology, assuming that the technology is available in the first period of the model. Thus, the formulation may become very onerous in terms of solution time, if many learning technologies are envisioned, and if the model is of large size to begin with. In section 11.5 we provide some comments on ETL, as well as a word of warning.

11.3 *Clustered learning*

An interesting variation of ETL is also available in TIMES, namely the case where several technologies use the same key technology (or component), itself subject to learning. For instance, table 11.1 lists 11 technologies using the key Gas Turbine technology. As experience builds up for gas the turbine, each of the 11 technologies in the cluster benefits. The phenomenon of clustered learning is modeled in TIMES via the following modification of the formulation of the previous section.

Let ***k*** designate the key technology and let ***l* = 1, 2, …,*L*** designate the set of clustered technologies attached to ***k***. The approach consists of three steps:

i) Step 1: designate ***k*** as a learning technology, and write for it the formulation of the previous section;
ii) Step 2: subtract from each $INVCOST_l$ the initial investment cost of technology ***k*** (this will avoid double counting the investment cost of ***k***);
iii) Step 3: add the following constraint to the model, in each time period. This ensures that learning on ***k*** spreads to all members of its cluster:

$$VAR\_NCAP_k - \sum_{l=1}^{L} VAR\_NCAP_l = 0$$

Table 11.1: Cluster of gas turbine technologies
(from A. Sebregts and K. Smekens, unpublished report, 2002)

Description
Integrated Coal gasification power plant
Integrated Coal Gasification Fuel Cell plant
Gas turbine peaking plant
Existing gas Combined Cycle power plant
New gas Combined Cycle power plant
Combined cycle Fuel Cell power plant
Existing gas turbine CHP plant
Existing Combined Cycle CHP plant
Biomass gasification: small industrial cog.
Biomass gasification: Combined Cycle power plant
Biomass gasification: ISTIG+reheat

11.4 *Learning in a multiregional TIMES model*

Technological learning may be acquired via global or local experience, depending on the technology considered. There are examples of technologies that were developed and perfected in certain regions of the world, but have tended to remain regional, never fully spreading globally. Examples are found in land management, irrigation, and in household heating and cooking devices. Other technologies are truly global in the sense that the same (or close to the same) technology becomes rather rapidly commercially available globally. In the latter case, global experience benefits users of the technology worldwide. Learning is said to *spillover* globally. Examples are found in large electricity plants, in steel production, wind turbines, and many other sectors.

The first and obvious implication of these observations is that the appropriate model scope must be used to study either type of technology learning. The formulation described in the previous sections is adequate in two cases: a) learning in a single region model, and b) regional learning in a multiregional model. It does *not* directly apply to *global learning* in a multiregional global model, where the cumulative investment variable must represent the sum of all cumulative investments in all regions together. We now describe an approach to global learning that may be implemented in TIMES, using only standard TIMES entities.

The first step in modeling multiregional ETL is to create one additional region, region 0, which will play the role of the Manufacturing Region. This region's RES consists only of the set of (global) learning technologies (LT's). Each such LT has the following specifications:

a) The LT has no commodity input.
b) The LT has only one output, a new commodity ***c*** representing the 'learning'. This output is precisely equal to the investment level in the LT in each period.
c) Commodity ***c*** may be exported to all other regions.

Finally, in each 'real' region, the LT is represented with all its attributes *except the investment cost NCAP_COST.* Furthermore, the construction of one unit of the LT requires an input of one unit of the learning commodity ***c*** (using the *NCAP_ICOM* parameter see chapter 3 of PART II). This ensures that the sum of all investments in the LT in the real regions is exactly equal to the investment in the LT in region 0, as desired.

11.5 ***Endogenous vs. exogenous learning: a discussion***

In this section, we formulate a few comments and warnings that may be useful to potential users of the ETL feature.

We start by stating a very important caveat to the ETL formulation described in the previous sections: if a model is run with such a formulation, it is very likely that the model will select some technologies, and *will invest massively at some early period* in these technologies unless it is prevented from doing so by additional constraints. Why this is likely to happen may be qualitatively explained by the fact that once a learning technology is selected for investing, two opposing forces are at play in deciding the optimal timing of the investments. On the one hand, the discounting provides an incentive for postponing investments. On the other hand, investing early allows the unit investment cost to drop immediately, and thus allows much cheaper investments in the learning technologies in the current and all future periods. Given the considerable cost reduction that is usually induced by learning, the first factor (discounting) is highly unlikely to predominate, and hence the model will tend to invest massively and early in such technologies, or not at all. Of course, what we mean by "massively" depends on the other constraints of the problem (such as the extent to which the commodity produced by the learning technology is in demand, the presence of existing technologies that compete with the learning technology, etc.). However, there is a clear danger that we may observe unrealistically large investments in some learning technologies.

ETL modelers are well aware of this phenomenon, and they use additional constraints to control the penetration trajectory of learning technologies. These constraints may take the form of upper bounds on the capacity of or the investment in the learning technologies in each time period, reflecting what is considered by the user to be realistic penetrations. These upper bounds play a determining role in the solution of the problem, and it is most often observed that the capacity of a learning technology is either equal to 0 or to the upper bound. This last observation indicates that the selection of upper bounds (or capacity/investment growth rates) by the modeler is the predominant factor in controlling the penetration of successful learning technologies.

In view of the preceding discussion, a fundamental question arises: is it worthwhile for the modeler to go to the trouble of modeling e*ndogenous* learning (with all the attendant computational burdens) when the results are to a large extent conditioned by *exogenous* upper bounds? We do not have a clear and unambiguous answer to this question; that is left for each modeler to evaluate.

However, given the above caveat, a possible alternative to ETL would consist in using exogenous learning trajectories. To do so, the same sequence of 'realistic' upper bounds on capacity would be selected by the modeler, and the values of the unit investment costs (INVCOST) would be externally computed by plugging these upper bounds into the learning formula (11-1). This approach makes use of the same exogenous upper bounds as the ETL approach, but avoids the MIP computational burden of ETL. Of course, the running of exogenous learning scenarios is not entirely foolproof, since there is no absolute guarantee that the capacity of a learning technology will turn out to be exactly equal to its exogenous upper bound. If that were not the case, a modified scenario would have to be run, with upper bounds adjusted downward. This trial-and-error approach may seem inelegant, but it should be remembered that it (or some other heuristic approach) might prove to be necessary in those cases where the number of learning technologies and the model size are both large (thus making the rigorous ETL formulation computationally intractable).

12 General equilibrium extensions

12.1 *Preamble*

In order to achieve a general (as opposed to partial) equilibrium, the energy system described in TIMES must be linked to a representation of the rest of the economy. The idea of hard-linking an energy model with the economy while still keeping the resulting model as an optimization program, dates back to the ETA-MACRO model (Manne, 1977), where both the energy system and the rest of the economy were succinctly represented by a small number of equations. This approach differs from the one taken by the so-called Computable General Equilibrium (CGE), models (Johanssen 1960, Rutherford 1992), where the calculation of the equilibrium relies on the resolution of simultaneous non-linear equations. In CGE's, the use of (non-linear, non-convex) equation solvers limits the size of the problem and thus the level of detail in the energy system description. This computational difficulty is somewhat (but not completely) alleviated when the computation relies on a single non-linear optimization program. Note however that MACRO is a much simplified representation of the economy as a single producing sector and no government sector, thus precluding the endogenous representation of taxes, subsidies, multi-sector interactions, etc. Therefore, the idea of a linked TIMES-MACRO model is not to replace the CGE's but rather to create an energy model where the feedbacks from the economy goes beyond the endogenization of demands (which TIMES does) to include the endogenization of capital.

Some years after ETA-MACRO, MARKAL-MACRO (Manne-Wene, 1992) was obtained by replacing the simplified ETA energy sub-model by the much more detailed MARKAL, giving rise to a large optimization model where most, but not all equations were linear. The MERGE model (Manne et al., 1995) is a multi-region version of ETA-MACRO with much more detail on the energy side –although not as much as in MARKAL-MACRO. The TIMES-MACRO model (Remme-Blesl, 2006) is based on exactly the same approach as MARKAL-MACRO. Both MARKAL-MACRO and TIMES-MACRO were essentially single-region models, until the multi-region version of TIMES-MACRO (named TIMES-MACRO-MSA, Kypreos-Lettila, 2013) was devised as an extension that accommodates multiple regions.

In this chapter, we describe the single region and the multi-region versions of TIMES-MACRO, focusing on the concepts and mathematical representation, whereas the implementation details are left to Part II of the TIMES documentation and to technical notes.

12.2 *The single-region TIMES-MACRO model*

As was already discussed in chapter 4, the main physical link between a TIMES model and the rest of the economy occurs at the level of the consumption of energy by the end-use sectors. There are however other links, such as capital and labor, which are common to the energy system and the rest of the economy. Figure 12.1 shows the articulation of the three links in TIMES-MACRO. Energy flows from TIMES to MACRO, whereas money flows in the reverse direction. Labor would also flow from MACRO to TIMES, but here a simplification is used, namely that the representation of labor is purely exogenous in both sub-models. Thus, TIMES-MACRO is not suitable for analyzing the impact of policies on labor, or on taxation, etc.

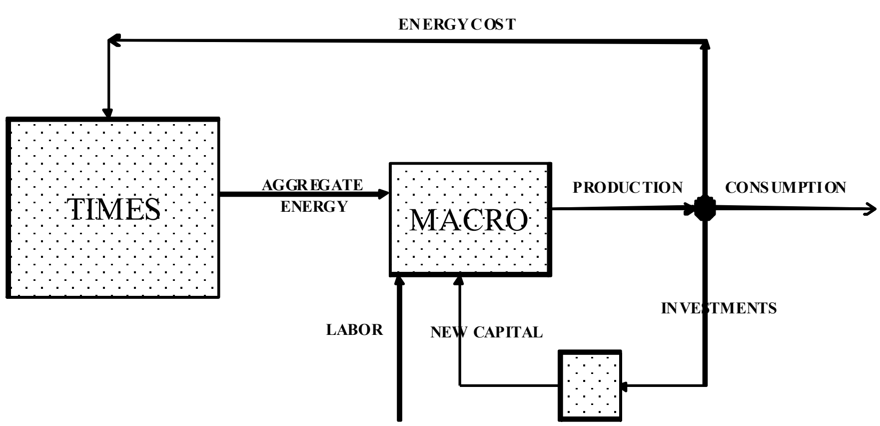


Figure 12.1. Energy, Labor, and Monetary flows between TIMES and MACRO

We now turn to the mathematical description of the above, starting with the MACRO portion of the hybrid model.

12.2.1 Formulation of the MACRO model

We start our description of the hybrid model by stating the MACRO equations (12-1) – (12-6)[39]:

$$Max \sum_{t=1}^{T-1} dfact_t \cdot \ln(C_t) + \frac{dfact_{T-1} \cdot dfactcurr_{T-1}^{\frac{d_{T-1}+d_T}{2}}}{1 - dfactcurr_T^{\frac{d_{T-1}+d_T}{2}}} \cdot \ln(C_T) \tag{12-1}$$

$$Y_t = C_t + INV_t + EC_t \tag{12-2}$$

$$Y_t = \left(akl \cdot K_r^{kpvs \cdot \rho} \cdot l_t^{(1-kpvs)\rho} + \sum_{dm} b_{dm} \cdot DEM\_M_{t,dm}^{\rho} \right)^{1/\rho} \tag{12-3}$$

$$l_1 = 1 \quad \text{and} \quad l_{t+1} = l_t \cdot (1 + growv_t)^{\frac{d_t + d_{t+1}}{2}} \tag{12-4}$$

$$K_{t+1} = tsrv_t \cdot K_t + \frac{1}{2}\left(d_t \cdot tsrv \cdot INV_t + d_{t+1} \cdot INV_{t+1}\right) \tag{12-5}$$

$$K_T \cdot (growv_T + depr) \le INV_T \tag{12-6}$$

with the model **variables**:

- C_t: annual consumption in period *t*,
- $DEM\_M_{t,dm}$: annual energy demand in MACRO for commodity *dm* in period *t*,
- Y_t: annual production in period *t*,
- INV_t: annual investments in period *t*,
- EC_t: annual energy costs in period *t*,
- K_t: total capital in period *t*

and the **exogenous parameters**:

- akl: production function constant,
- b_{dm}: demand coefficient,

[39] The concrete implementation in the TIMES-MACRO model differs in some points, e.g. the consumption variable in the utility function is substituted by equations (12-2) and (12-3).

d_t: duration of period t in years,

$depr$: depreciation rate,

$dfact_t$: utility discount factor,

$dfactcurr_t$: annual discount rate,

$growv_t$: growth rate in period t,

$kpvs$ capital value share,

l_t: annual labor index in period t,

ρ: substitution constant,

T: period index of the last period,

$tsrv_t$: capital survival factor between two periods.

The objective function (12-1) of the MACRO model is the maximization of the summation of discounted utility at each period. The utility is defined as the logarithm of consumption C_t of the households. A logarithmic utility function embodies a decreasing marginal utility property (Manne, 1977). Note that the discount factor $dfact_t$ for period t must take into account both the length of the period and the time elapsed between the period's start and the base year. Note also that the discount factor of the last period has a larger impact since it is assumed to apply to the infinite time horizon after the last model period (alternatively, the user may decide to limit the number of years in the last term, in those cases where it is deemed important to confer less weight to the indefinite future).

The national accounting equation (12-2) simply states that national production Y_t must cover national consumption C_t , plus investments INV_t , plus energy costs EC_t.

The production function (12-3) represents the entire economy. It is a nested, constant elasticity of substitution (CES) function with the three input factors capital, labor and energy. The production input factors labor l_t and capital K_t form an aggregate, in which both can be substituted by each other represented via a Cobb-Douglas function. Then, the aggregate of the energy services and the aggregate of capital and labor can substitute each other. Note that labor is not endogenous in MACRO,but is specified exogenously by the user provideda labor growth rate $growv_t$.

The energy in term in (12-3) is a weighted sum of end-use demands in all sectors dm of the economy, $DEM\_M_{t,dm}$, raised to the power ρ. We defer the definition of these quantities until the next subsection.

The lower the value of the elasticity of substitution the closer is the linkage between economic growth and increase in energy demand. For homogenous production functions with constant returns to scale[40] the substitution constant ρ in (12-3) is directly linked with the user-defined elasticity of substitution σ by the expression $\rho = 1 - 1/\sigma$.

The capital value share $kpvs$ describes the share of capital in the sum of all production factors and must be specified by the user. The parameter akl is the level constant of the production function. The parameters akl and b_{dm} of the production are determined based on the results from a TIMES model run without the MACRO module.

The capital dynamics equation (12-5) describes the capital stock in the current period K_{t+1} based on the capital stock in the previous period and on investments made in the current and the previous period. Depreciation leads to a reduction of the capital. This effect is taken into account by the capital survival factor $tsrv_t$, which describes the share of the capital or investment in period *t* that still exists in period *t+1*. It is derived from the depreciation rate $depr$ using the following expression:

$$tsrv_t = (1 - depr)^{\frac{(d_{t+1} + d_t)}{2}} \qquad (12\text{-}7)$$

Expression (12-7) calculates the capital survival factor for a period of years beginning with the end of the middle year m_t and ending with the end of the year m_{t+1}. The duration between these two middle years equals the duration $\frac{d_{t+1} + d_t}{2}$. Then, a mean investment in period *t* is calculated by weighting the investments in *t* and *t+1* with the respective period duration: $1/2\left(d_t \cdot tsrv \cdot INV_t + d_{t+1} \cdot INV_{t+1}\right)$.

For the first period it is assumed that the capital stock grows with the labor growth rate of the first period $growv_0$. Thus, the investment has to cover this growth rate plus the depreciation of capital. Since the initial capital stock is given and the depreciation and growth rates are exogenous, the investment in the first period can be calculated beforehand:

---

[40] A production function is called homogenous of degree *r*, if multiplying all production factors by a constant scalar leads λ to an increase of the function by λ^r. If *r*= 1, the production function is called linearly homogenous and leads to constant returns to scale.

$$INV_0 = K_0 \cdot (depr + growv_0) \qquad (12\text{-}8)$$

Since the model horizon is finite, one has to ensure that the capital stock is not fully exhausted (which would maximize the utility in the model horizon.) Therefore a terminal condition (12-6) is added, which guarantees that after the end of the model horizon a capital stock for the following generations exists. It is assumed that the capital stock beyond the end of horizon grows with the labor growth rate $growv_T$. This is coherent with the last term of the utility function.

12.2.2 Linking MACRO with TIMES

TIMES is represented via the following condensed LP

$$Min \sum_t dfact_t \cdot COST\_T_t \quad (x)$$

s.t.

$$E \cdot x = DEM\_T_{dm,t} \qquad (A)$$

$$A \cdot x = b \qquad (B)$$

where

- x is the vector of TIMES variables
- $COST\_T_t \quad (x)$ is the annual undiscounted cost TIMES expression
- $dfact_t$ is the discount factor for period t
- equations *(A)* express the satisfaction of demands in TIMES (and thus defines the $DEM\_T_{dm,t}$ variables), and
- equations *(B)* is the set of all other TIMES constraints

MACRO and TIMES are hard linked via two sets of variables: the energy variables $DEM\_T_{dm,t}$, and the period energy costs $COST\_T_t$.

The aggregate energy input into MACRO (see equation (12-3)), is slightly different from the TIMES variables defined above. In the linked model, each term *DEM_M* is obtained by further applying a factor $aeeifac_{t,dm}$ as shown in equation (12-9).

$$DEM\_T_{t,dm} = aeeifac_{t,dm} \cdot DEM\_M_{t,dm} \qquad (12\text{-}9)$$

Indeed, the energy demand in the TIMES model can be lower than the energy requirement of the MACRO model due to demand reductions, which are caused by autonomous energy efficiency improvements and come in addition to those captured in the energy sector of the TIMES model. The autonomous energy efficiency improvement factor $aeeifac_{t,dm}$ is determined in a calibration procedure described in technical note "Documentation of the TIMES-MACRO model", which also discusses the weighing coefficients b_{dm}.

The other link consists in accounting for the monetary flow EC_t, equal to the expenditures made in the energy sector. Precisely, EC_t is equal to the annual *undiscounted* energy system cost of the TIMES model, $COST\_T_t$, (as used in the TIMES objective function), augmented with an additional term as shown in equation (12-10):

$$COST\_T_t + \frac{1}{2} qfac \sum_p \frac{cstinv_{t,p}}{expf_t \cdot capfy_p} \cdot XCAP_{t,p}^2 = EC_t \tag{12-10}$$

with

- $XCAP_{t,p}$: portion of the capacity expansion for technology *p* in period *t* that is penalized. Constraint (12-11) below states that it is the portion exceeding a predefined tolerable expansion rate $expf_t$,
- EC_t: costs for the production factor energy in the MACRO model,
- $qfac$: trigger to activate penalty term (0 for turning-off penalty, 1 for using penalty term),
- $cstinv_{t,p}$: specific annualized investment costs of technology *p* in period *t*,
- $capfy_p$: maximum level of capacity for technology *p*,
- $expf_t$: tolerable expansion between two periods.

Just like in the pure MACRO model, the quadratic penalty term added on the left hand side of Eqn. (11) serves to slow down the penetration of technologies. This term plays a somewhat similar role as the growth constraints do in the stand-alone TIMES model. The variable $XCAP_{t,p}$ is the amount of capacity exceeding a predefined expansion level expressed by the expansion factor $expf_t$ and is determined by the following equation:

$$VAR\_CAP_{t+1,p} \leq (1 + expf_t) \cdot VAR\_CAP_{t,p} + XCAP_{t+1,p} \tag{12-11}$$

with:

$VAR\_CAP_{t,p}$: total installed capacity of technology p in period t.

As long as the total installed capacity in period $t+1$ is below $\left(1+expf_t\right)\cdot CAP_{t,p}$ no penalty costs are applied. For the capacity amount $XCAP_{t+1,p}$ exceeding this tolerated capacity level penalty costs are added to the regular costs of the TIMES model in Equation (12-10).

The quadratic term in Eqn. (11) introduces a large number of nonlinear terms (one for each technology and period) that may constitute a considerable computational burden for large models. These constraints are therefore replaced in the current implementation of TIMES, by linear piece-wise approximations in a way quite similar to what was done to linearize the surplus in chapter 4.

12.2.3 A brief comment

In spite of the linearization of the penalty terms in equation (12-10), TIMES-Macro still contains non-linearities: its objective function is a concave function, a good property when maximizing, but there are T nonlinear, non convex constraints as per equation (12-3) that introduce a non trivial computational obstacle to large size instances of the model.

Although not discussed here, the calibration of the TIMES-MACRO model is an exceedingly important task, since the model must agree with the initial state of the economy in the dimensions of labor, capital, and the links between the energy sector and the economy at large. Fuller details on calibration are provided in the above-mentioned technical note.

Overall, the experience with TIMES-MACRO has been good, with sizable model instances solved in reasonable time. But the modeler would benefit from carefully weighing the limitation of model size imposed by the non-linear nature of TIMES-MACRO, against the advantage of using a (single sector) general equilibrium model.

12.3 ***The multi-regional TIMES-MACRO model (MSA)***

In this section, we only sketch the generalization of TIMES-MACRO to a multi-regional setting. Full details, including the important calibration step and other implementation issues, appear in technical note "TIMES-Macro: Decomposition into Hard-Linked LP and NLP Problems".

12.3.1 Theoretical background

In a multi-regional setting, inter-regional trade introduces an important new complication in the calculation of the equilibrium[41]. Indeed, the fact that the utility function used in the MACRO module is highly non linear also means that the global utility is not equal to the sum of the national utilities. Also, it would be impractical and conceptually wrong to define a single consumption function for the entire set of regions, since the calibration of the model may only be done using national statistics, and furthermore, there may be large differences in the parameters of each region's production function, etc.

It follows from the above that it is not possible to use a single optimization step to calculate the global equilibrium. Instead, one must resort to more elaborate approaches in order to compute what is termed a Pareto-optimal solution to the equilibrium problem, i.e. a solution where the utility of any region may not be improved without deteriorating the utility of some other region(s).

Such a situation has been studied in the economics literature, starting with the seminal paper by Negishi (1960) that established the existence of equilibria that are Pareto-optimal in the Welfare functions. Manne (1999) applied the theory to the MACRO model, and Rutherford (1992) proposed a decomposition algorithm that makes the equilibrium computation more tractable. The Rutherford algorithm is used in the TIMES-MACRO model. An interesting review of the applications of Negishi theory to integrated assessment models appeared in Stanton (2010).

12.3.2 A sketch of the algorithm to solve TIMES-MACRO-MSA

Rutherford's procedure is an iterative decomposition algorithm. Each iteration has two steps. The first step optimizes a large TIMES LP and the second step optimizes a stand-

[41] Of course, if no trade between the regions is assumed, the global equilibrium amounts to a series of independent national equilibria, which may be calculated by the single region TIMES-MACRO.

alone *reduced* non-linear program which is an alteration of MACRO, and is named MACRO-MSA. These two steps are repeated until convergence occurs.
Because the two steps must be solved repeatedly, the iterative procedure is computationally demanding; furthermore, it is established that the speed of convergence is dependent upon the number of trade variables that link the regions. For this and other reasons, the trade between regions is limited to a single commodity, namely a *numéraire*, expressed in monetary units. The numéraire $\boldsymbol{NTX_{r,t}}$ affects the national account equation (12-2) of each region, as follows:

$$Y_{r,t} = C_{r,t} + INV_{r,t} + EC_{r,t} + NTX_{r,t}$$

and is subject to the conservation constraint: $\sum_r NTX_{r,t} = 0 \quad \forall \{t\}$,

which insures that trade is globally balanced.

First step: at each iteration, the first step is the resolution of TIMES using non-elastic demands provided by the previous solution of the non-linear program (except at iteration 1, where demands are either exogenously provided or generated by TIMES[42]).

Second step: once the TIMES solution is obtained, it is used to form a quadratic expression representing an approximation of the aggregate energy cost, to be used in MACRO-MSA. Defining this approximation is the crux of Rutherford decomposition idea. It replaces the entire TIMES model, thus greatly simplifying the resolution of Step 2. The global objective function of MACRO-MSA is a weighted sum (over all regions) of the regional MACRO welfare functions, where the weights are the Negishi weights for each region. The thus modified global objective function is maximized. Then, a convergence criterion is checked. If convergence is not observed, the new demands are fed into TIMES and a new iteration is started. The Negishi weights are also updated at each iteration, leading to a new version of the objective, until the algorithm converges to the Pareto-optimal equilibrium.

The adaptation of Rutherford algorithm to TIMES-MACRO was formalized by Kypreos (2006) and implemented by Kypreos and Lettila as the above-mentioned technical note.

[42]It may be desirable, although not required to use non-zero demand elasticities at the very first iteration.

13 Appendix A: History and comparison of MARKAL and TIMES

13.1 *A brief history of TIMES*[43] *and MARKAL*

The TIMES (The Integrated Markal-EfomSystem) and the MARKAL (MARket ALlocation) models have a common history beginning in the 1970's when a formal decision of the International Energy Agency (IEA) led to the creation of a common tool for analyzing energy systems, to be shared by the participating OECD nations. MARKAL became a reality by the year 1980 and became a common tool of the members of the Energy Technology Systems Analysis Programme (ETSAP), an IEA Implementing Agreement (IA).

Development of the new modeling paradigm was undertaken over a period of three years. First a team of national experts from more than sixteen countries met numerous times to define the data requirements and mathematics that were to underpin MARKAL. Then the actual coding and testing of the model formulation proceeded on two parallel tracks. One team at Brookhaven National Laboratory (BNL) embarked on the undertaking employing OMNI[44], a specialized programming language specifically designed for optimization modeling, that was widely used for modeling oil refinery operations. The second team at KFA Julich chose to use Fortran to code the model. While both teams initially succeeded, changes were quickly necessary that proved to be more manageable in the BNL OMNI version of MARKAL than in the KFA Fortran version – leading to the decision to formally adopt only the BNL OMNI version for general use. A full description of this initial incarnation of the model maybe be found in the MARKAL User's Guide (Fishbone, 1983).

MARKAL was used intensively by ETSAP members throughout the two decades after 1980 and beyond, undergoing many improvements. The initial mainframe OMNI version of MARKAL was in use until 1990, when BNL ported the model to the person computer that was just becoming a viable alternative. At the same time, as part of this move of MARKAL to the PC, the first model management system for MARKAL databases and model results was developed at BNL which greatly facilitated working with MARKAL and opened it up to a new class of users. This PC based shell, MUSS (MARKAL User Support System, Goldstein, 1991), provided spreadsheet-like browse/edit facilities for

---

[43] With the kind permission of Professor Stephen Hawking

[44] A product of Haverly Systems Incorporated, http://www.haverly.com/.

managing the input data, Reference Energy System (RES) network diagramming to enable viewing the underlying depiction of the energy system, scenario management and run submission, and multi-case comparison graphics that collectively greatly facilitated the ability to work effectively with MARKAL.

The next big step in the evolution of MARKAL/TIMES arose from the BNL collaboration with Professor Alan Manne of Stanford University resulting in the porting of MARKAL to the more flexible General Algebraic Modeling System (GAMS), still used for TIMES today. The driving motivation for this move to GAMS was to enable the creation of MARKAL-MACRO (see chapter 12), a major model variant enhancement resulting in a General Equilibrium version of the model. One drawback of MARKAL-MACRO is that it was implemented as a non-linear programming (NLP) optimization model, which limits its usability for large energy system models.

To overcome this shortcoming while embracing one of the main benefits arising from MARKAL-MACRO, another major model enhancement was implemented in 1995 from a proposal made in 1980 by Giancarlo Tosato (1980), to allow end-use service demands to be price sensitive, thus transforming MARKAL from a supply cost optimization model to a system computing a supply demand partial equilibrium, named MARKAL-ED (Loulou and Lavigne, 1996) while retaining its linear form. An alternative formulation using non-linear programming, MARKAL-MICRO (Van Regemorter, 1998) was also implemented. Many other enhancements were made in the late 1990's and early 2000's and are described in the second comprehensive version of the MARKAL model documentation (Loulou et al., 2004).

The development of ANSWER, the first Windows interface for MARKAL, commenced at the Australian Bureau of Agricultural and Resource Economics (ABARE) in Canberra in early 1996 with primary responsibility taken by then ABARE staff member Ken Noble. By early 1998 the first production version of ANSWER-MARKAL was in use, including by most ETSAP Partners. In late 2003 Ken Noble retired from ABARE, established Noble-Soft Systems and became the owner of the ANSWER-MARKAL software, thereby ensuring its continuing development and support.

By the late 1990's, the need to gather all the existing MARKAL features and to create many new ones was becoming pressing, and an international group of ETSAP researchers was formed to create what became the TIMES model generator. The main desired new features were as follows:

- To allow time periods to be of unequal lengths, defined by the user;

- To allow the user data to control the model structure;
- To make data as independent as possible of the choice of the model periods (data decoupling), in particular to facilitate the recalibration of the model when the initial period is changed, but also to avoid having to redefine the data when period lengths are altered;
- To formally define commodity flows as new variables (as in the EFOM model), thus making it easier to model certain complex processes;
- To define vintaged processes that allow input data to change according to the investment year;
- To enable the easy creation of flexible processes, a feature that was feasible with MARKAL only by creating multiple technologies;
- To permit time-slices to be entirely flexible with a tiered hierarchy of year/season/week/time-of-day to permit much more robust modeling of the power sector;
- To improve the representation and calculation of costs in the objective function;
- To formally identify trade processes in order to facilitate the creation of multi-regional models;
- To define storage processes that carry some commodities from one time-slice to another or from some period to another; and
- To implement more dynamic and inter-temporal user-defined constraints.

Definition and development of TIMES began in late 1998, resulting in a beta version in 1999, and the first production version in year 2000, initially used by only a small number of ETSAP members. The transition from MARKAL to TIMES was slower than anticipated, mainly because ETSAP modellers already had mature MARKAL databases that required serious time and effort to be converted into TIMES databases.

Furthermore there was a need for a TIMES specific model shell to manage the new model. Two data handling shells were created during the 2000's by two private developers closely associated with ETSAP and with partial support from ETSAP: In the early 2000’s, VEDA_FE (VErsatile Data Analysis -Front End) (http://www.kanors.com/Index.asp) and in 2008, ANSWER-TIMES (Noble-Soft Systems, 2009). Even before that, a back-end version of VEDA (VEDA_BE, Kanudia http://www.kanors.com/Index.asp) had been created to explore and exploit the results and create reports.

Following these developments, and as the merits of TIMES over MARKAL became increasingly evident, TIMES became the preferred modeling tool for most ETSAP members, old and new, as well as for energy system modellers who were not formal ETSAP members, but were either associated with ETSAP as partners in several outreach projects or on their own.

The first complete documentation of the TIMES model generator was written in 2005 and made available on the ETSAP website (http://www.iea-etsap.org/web/index.asp). It has since been replaced by this documentation.

As the number of modellers increased and they gained experience with TIMES, the model underwent many new additions and enhancements, and the number of publications based on TIMES rose sharply. One development started in 2000 and achieved by 2005 was the creation of the first world multi-regional TIMES model (Loulou, 2007) and the simultaneous creation of a Climate Module (chapter 7). Together, these two realizations allowed ETSAP to participate in the Stanford Energy Modeling Forum (EMF, https://emf.stanford.edu/) and conduct global climate change analyses alongside other modellers who were mostly using general equilibrium models. Following these developments, several ETSAP teams created multiple versions of global TIMES models.

At the same time other major new features were implemented, some of them found in MARKAL though often further advanced in TIMES, such as the Endogenous Technological Learning feature (chapter 11), the lumpy investment feature (chapter 10), both of which required the use of mixed integer programming, and the multi-stage Stochastic Programming option (chapter 8) allowing users to simulate uncertain scenarios. A particularly challenging development was to enable the computation of general equilibria in a multi-regional setting, since doing so required a methodology beyond simple optimization (chapter 12).

Increasingly as TIMES benefitted from many enhancements and gained prominence in the community of modellers, and while some features found their way into the MARKAL model, in order to provide similar capabilities to the large existing MARKAL user base, ETSAP decided that there would be no further development of MARKAL though support would continue to be provided to the existing users. By the early 2010's, TIMES (and MARKAL) models were recognized as major contributors within the community of energy and climate change researchers, and the number of outreach projects increased tremendously. Today it is estimated that MARKAL/TIMES has been introduced to well over 300 institutions in more than 80 countries, and is generally considered the benchmark integrated energy system optimization platform available for use around the world.

13.2 ***A comparison of the TIMES and MARKAL models***

This section contains a point-by-point comparison of highlights of the TIMES and MARKAL models. It is of interest primarily to modelers already familiar with MARKAL, and to provide a sense of the advancements embodied in TIMES. The descriptions of the features given below are not detailed, since they are repeated elsewhere in the documentation of both models. Rather, the function of this section is to guide the reader, by mentioning the features that are present or improved in one model that are not found or only in a simplified form in the other.

13.2.1 Similarities

The TIMES and the MARKAL models share the same basic modeling paradigm. Both models are technology explicit, dynamic partial equilibrium models of energy markets[45]. In both cases the equilibrium is obtained by maximizing the total surplus of consumers and suppliers via Linear Programming, while minimizing total discounted energy system cost. Both models are by default clairvoyant, that is, they optimize over the entire modeling horizon, though partial look-ahead (or myopic) may also be employed. The two models also share the multi-regional feature, which allows the modeler to construct geographically integrated (even global) instances, though in MARKAL there are no inter-regional exchange process making the representation of trade (much) more cumbersome. These fundamental features were described in Chapter 3 of this documentation, and Section 1.3, PART I of the MARKAL documentation, and constitute the backbone of the common paradigm. However, there are also significant differences in the two models, which we now outline. These differences do not affect the basic paradigm common to the two models, but rather some of their technical features and properties.

13.2.2 TIMES features not in MARKAL

13.2.2.1 Variable length time periods

MARKAL has fixed length time periods, whereas TIMES allows the user to define period lengths in a completely flexible way. This is a major model difference, which indeed required a complete re-definition of the mathematics of most TIMES constraints and of the TIMES objective function. The variable period length feature is very useful in

[45] But recall that some extensions depart from the classical equilibrium properties, see chapters 8-12.

two instances: first if the user wishes to use a single year as initial period (quite useful for calibration purposes), and second when the user contemplates long horizons, where the first few periods may be described in some detail by relatively short periods (say 5 years), while the longer term may be regrouped into a few periods with long durations (perhaps 20 or more years).

13.2.2.2 Data decoupling

This somewhat misunderstood feature does not confer additional power to TIMES, but it greatly simplifies the maintenance of the model database and allows the user great flexibility in modifying the new definition of the planning horizon. In TIMES all input data are specified by the user independently from the definition of the time periods employed for a particular model run. All time-dependent input data are specified by the year in which the data applies. The model then takes care of matching the data with the periods, wherever required. If necessary the data is interpolated (or extrapolated) by the model preprocessor code to provide data points at those time periods required for the current model run. In addition, the user has control over the interpolation and extrapolation of each time series.

The general rule of data decoupling applies also to past data: whereas in MARKAL the user had to provide the residual capacity profiles for all existing technologies in the initial period, and over the periods in which the capacity remains available, in TIMES the user may provide technical and cost data at those past years when the investments actually took place, and the model takes care of calculating how much capacity remains in the various modeling periods. Thus, past and future data are treated essentially in the same manner in TIMES.

One instance when the data decoupling feature immensely simplifies model management is when the user wishes to change the initial period, and/or the lengths of the periods. In TIMES, there is essentially nothing to do, except declaring the dates of the new periods. In MARKAL, such a change represents a much larger effort requiring a substantive revision of the database.

13.2.2.3 Flexible time slices and storage processes

In MARKAL, only two commodities have time-slices: electricity and low temperature heat, with electricity having seasonal and day/night time-slices, and heat having seasonal time-slices. In TIMES, any commodity and process may have its own, user-chosen time-slices. These flexible time-slices are segregated into three groups, seasonal (or monthly), weekly (weekday vs. weekend), and daily (day/night or hourly), where any level may be expanded (contracted) or omitted.

The flexible nature of the TIMES time-slices supports storage processes that 'consume' commodities at one time-slice and release them at another. MARKAL only supports night-to-day (electricity) storage.

Note that many TIMES parameters may be time-slice dependent (such as availability factor (AF), basic efficiency (ACT_EFF), etc.

13.2.2.4 Process generality

In MARKAL processes in different RES sectors are endowed with different (data and mathematical) properties. For instance, end-use processes do not have activity variables (activity is then equated to capacity), and resource processes have no investment variables. In TIMES, all processes have the same basic features, which are activated or not solely via data specification, with some additional special features relevant to trade and storage processes.

13.2.2.5 Flexible processes

In MARKAL processes are by definition rigid, except for some specialized processes which permit flexible output (such as limit refineries or pass-out turbine CHPs), and thus outputs and inputs are in fixed proportions with one another. In TIMES, the situation is reversed, and each process starts by being entirely flexible, unless the user specifies certain attributes to rigidly link inputs to outputs. This feature permits better modeling of many real-life processes as a single technology, where MARKAL may require several technologies (as well as dummy commodities) to achieve the same result. A typical example is that of a boiler that accepts any of 3 fuels as input, but whose efficiency depends on the fuel used. In MARKAL, to model this situation requires four processes (one per possible fuel plus one that carries the investment cost and other parameters), plus one dummy fuel representing the output of the three "blending" process. In TIMES one process is sufficient, and no dummy fuel is required. Note also that TIMES has a number of parameters that can limit the input share of each fuel, whereas in MARKAL, imposing such limits requires that several user constraints be defined.[46]

---

[46]In the end the two models use equivalent mathematical expressions to represent a flexible process. However, TIMES reduces the user's effort to a minimum, while MARKAL requires the user to manually define the multiple processes, dummy fuels and user constraints.

13.2.2.6 Investment and dismantling lead-times and costs

New TIMES parameters allow the user to model the construction phase and dismantling of facilities that have reached their end-of-life. These are: lead times attached to the construction or to the dismantling of facilities, capital costs for dismantling, and surveillance costs during dismantling. Like in MARKAL, there is also the possibility to define flows of commodities consumed at construction time, or released at dismantling times, thus allowing the representation of life-cycle energy and emission accounting.

13.2.2.7 Vintaged processes and age-dependent parameters

The variables associated with user declared vintaged processes employ both the time period *p* and vintage period *v* (in which new investments are made and associated input data is obtained). The user indicates that a process is to be modeled as a vintaged process by using a special vintage parameter. Note that in MARKAL vintaging is possible only for end-use devices (for which there is no activity variable) and only applies to the device efficiency (and investment cost, which is always vintaged by definition for all technologies) or via the definition of several replicas of a process, each replica being a different vintage. In TIMES, the same process name is used for all vintages of the same process.[47]

In addition, some parameters may be specified to have different values according to the age of the process. In the current version of TIMES, these parameters include the availability factors, the in/out flow ratios (equivalent to efficiencies), and the fixed cost parameters only. Several other parameters could, in principle, be defined to be age-dependent, but such extensions have not been implemented yet.

13.2.2.8 Commodity related variables

MARKAL has very few commodity related variables, namely exports/imports, and emissions. TIMES has a large number of commodity-related variables such as: total production, total consumption, but also (and most importantly) specific variables representing the flows of commodities entering or exiting each process. These variables

---

[47]The representation of vintage as a separate index helps eliminate a common confusion that existed in MARKAL, namely the confusion of *vintage* with the *age* of a process. For instance, if the user defines in MARKAL an annual O&M cost for a car, equal to 10 in 2005 and only 8 in 2010, the decrease would not only apply to cars purchased in 2010, but also to cars purchased in 2005 and earlier when they reach the 2010 period.

provide the user with many “handles” to define bounds and costs on commodity flows, and foster easier setup of user constraints looking to impose shares across technology groups (e.g., renewable electricity generation targets, maximum share of demand that can be met by a (set of) devices).

13.2.2.9 More accurate and realistic depiction of investment cost payments

In MARKAL each investment is assumed to be paid in its entirety at the beginning of thetime period in which it becomes available. In TIMES the timing of investment payments is quite detailed. For large facilities (e.g. a nuclear plant), capital is progressively laid out in yearly increments over the facility’s construction time, and furthermore, the payment of each increment is made in installments spread over the economic life (which may differ from the technical lifetime) of a facility. For small processes (e.g. a car) the capacity expansion is assumed to occur regularly each year rather than in one large lump, and the payments are therefore also spread over time. Furthermore, when a time period is quite long (i.e. longer that the life of the investment), TIMES has an automatic mechanism to repeat the investment more than once over the period. These features allow for a much smoother (and more realistic) representation of the stream of capital outlays in TIMES than in MARKAL.

Moreover, in TIMES all discount rates can be defined to be time-dependent, whereas in MARKAL both the general and technology-specific discount rates are constant over time.

13.2.2.10 Stochastic Programming

Both MARKAL and TIMES support stochastic programming (SP, Chapter 8) as a means for examining uncertainty and formulating hedging strategies to deal with same. In MARKAL only 2-stage SP was implemented, and thus the resolution of the uncertainty could only occur at one particular time period, whereas in TIMES uncertainty may be resolved progressively at different successive periods (e.g., mitigation level at one period and demand level at another).

13.2.2.11 Climate module

TIMES possesses a set of variables and equations that endogenize the concentration of CO_2, CH4, and N2O, and also calculate the radiative forcing and global temperature changes resulting from GHG emissions and accumulation here. This new feature is described in Chapter 7.

14 Appendix B: Linear Programming complements

This section is not strictly needed for a basic understanding of the TIMES model and may be skipped a first reading. However, it provides additional insight into the microeconomics of the TIMES equilibrium. In particular, it contains a review of the theoretical foundation of Linear Programming and Duality Theory. This knowledge may help the user to better understand the central role shadow prices and reduced costs play in the economics of the TIMES model. More complete treatments of Linear Programming and Duality Theory may be found in several standard textbooks such as Chvátal (1983) or Hillier and Lieberman (1990 and subsequent editions). Samuelson and Nordhaus (1977) contains a treatment of micro-economics based on mathematical programming.

14.1 ***A brief primer on Linear Programming and Duality Theory***

14.1.1 Basic definitions

In this subsection, the superscript *t* following a vector or matrix represents the transpose of that vector or matrix. A Linear Program may always be represented as the following *Primal Problem* in canonical form:

	$Max\ c^t x$	(14-1)
s.t.	$Ax \leq b$	(14-2)
	$x \geq 0$	(14-3)

where x is a vector of *decision variables*, $c^t x$ is a linear function representing the *objective* to maximize, and $Ax \leq b$ is a set of inequality *constraints*. Assume that the LP has a finite optimal solution, x^*.

Then each decision variable, x^*_j falls into one of three categories. x^*_j may be:

- equal to its lower bound (as defined in a constraint), or
- equal to its upper bound, or
- strictly between the two bounds.

In the last case, the variable x^*_j is called *basic*. Otherwise it is *non-basic*.

For each primal problem, there corresponds a *Dual problem* derived as follows:

$$\text{Min } b^t y \quad (14\text{-}4)$$

$$\text{s.t.} \quad A^t y \geq c \quad (14\text{-}5)$$

$$y \geq 0 \quad (14\text{-}6)$$

Note that the number of dual variables equals the number of constraints in the primal problem. In fact, each dual variable y_i may be assigned to its corresponding primal constraint, which we represent as: $A_i x \leq b_i$, where A_i is the i^{th} row of matrix A.

14.1.2 Duality Theory

Duality theory consists mainly of three theorems[48]: weak duality, strong duality, and complementary slackness.

Weak Duality Theorem

If x is any feasible solution to the primal problem and y is any feasible solution to the dual, then the following inequality holds:

$$c^t x \leq b^t y \quad (14\text{-}7)$$

The weak duality theorem states that the value of a feasible dual objective is never smaller than the value of a feasible primal objective. The difference between the two is called the *duality gap* for the pair of feasible primal and dual solutions *(x,y)*.

Strong duality theorem

If the primal problem has a *finite, optimal* solution x^*, then so does the dual problem (y^*), and both problems have the same optimal objective value (their duality gap is zero):

$$c^t x^* = b^t y^* \quad (14\text{-}8)$$

Note that the optimal values of the dual variables are also called the *shadow prices* of the primal constraints.

[48] Their proofs may be found in the textbooks on Linear Programming already referenced.

Complementary Slackness theorem

At an optimal solution to an LP problem:

- If y^*_i is > 0 then the corresponding primal constraint is satisfied at equality (i.e. $A_i x^*=b_i$ and the i^{th} primal constraint is called *tight*. Conversely, if the i^{th} primal constraint is *slack* (not tight), then $y^*_i = 0,$
- If x^*_j is basic, then the corresponding dual constraint is satisfied at equality, (i.e. $A^t_j{}^*y = c_j$, where A^t_j is the j^{th} row of A^t, i.e. the j^{th} column of A. Conversely, if the j^{th} dual constraint is slack, then x^*_j is equal to one of its bounds.

Remark: Note however that a primal constraint may have zero slack and yet have a dual equal to 0. And, a primal variable may be non basic (i.e. be equal to one of its bounds), and yet the corresponding dual slack be still equal to 0. These situations are different cases of the so-called degeneracy of the LP. They often occur when constraints are over specified (a trivial case occurs if a constraint is repeated twice in the LP)

14.2 ***Sensitivity analysis and the economic interpretation of dual variables***

It may be shown that if the j^{th} RHS b_j of the primal is changed by an infinitesimal amount d, and if the primal LP is solved again, then its new optimal objective value is equal to the old optimal value plus the quantity $y_j^*\bullet d$, where y_j^* is the optimal dual variable value.

Loosely speaking[49], one may say that the partial derivative of the optimal primal objective function's value with respect to the RHS of the i^{th} primal constraint is equal to the optimal shadow price of that constraint.

14.2.1 Economic interpretation of the dual variables

If the primal problem consists of maximizing the surplus (objective function $c^t x$), by choosing an activity vector x, subject to upper limits on several resources (the b vector) then:

- Each a_{ij} coefficient of the dual problem matrix, $A,$ then represents the consumption of resource b_j by activity x_i;
- The optimal dual variable value y^*_j is the unit price of resource $j,$ and

[49] Strictly speaking, the partial derivative may not exist for some values of the RHS, and may then be replaced by a directional derivative (see Rockafellar 1970).

- The total optimal surplus derived from the optimal activity vector, x^*, is equal to the total value of all resources, b, priced at the optimal dual values y^* (strong duality theorem).

Furthermore, each dual constraint $A^t_j{}^*y \geq c_j$ has an important economic interpretation. Based on the Complementary Slackness theorem, if an LP solution x^* is optimal, then for each x^*_j that is not equal to its upper or lower bound (i.e. each basic variable x^*_j), there corresponds a *tight* dual constraint $y^*A'_j = c_j$, which means that the revenue coefficient c_j must be exactly equal to the cost of purchasing the resources needed to produce one unit of x_j. In economists' terms, *marginal cost equals marginal revenue, and both are equal to the market price of* $\boldsymbol{x^*_j}$. If a variable is not basic, then by definition it is equal to its lower bound or to its upper bound. In both cases, the unit revenue c_j need not be equal to the cost of the required resources. The technology is then either non-competitive (if it is at its lower bound) or it is super competitive and makes a surplus (if it is at its upper bound).

Example: The optimal dual value attached to the balance constraint of commodity c represents the change in objective function value resulting from one additional unit of the commodity. This is precisely the internal unit price of that commodity.

14.2.2 Reduced surplus and reduced cost

In a maximization problem, the difference $y^*A'_j - c_j$ is called the *reduced surplus* of technology j, and is available from the solution of a TIMES problem. It is a useful indicator of the competitiveness of a technology, as follows:

- If x^*_j is at its lower bound, its unit revenue c_j is *less* than the resource cost (i.e. its reduced surplus is positive). The technology is not competitive (and stays at its lower bound in the equilibrium);
- If x^*_j is at its upper bound, revenue c_j is *larger* than the cost of resources (i.e. its reduced surplus is negative). The technology is super competitive and produces a surplus; and
- If x^*_j is basic, its reduced surplus is equal to 0. The technology is competitive but does not produce a surplus.

We now restate the above summary in the case of a Linear Program that minimizes cost subject to constraints:

$$
\begin{aligned}
& Min\ c^t x \\
\text{s.t.}\quad & Ax \geq b \\
& x \geq 0
\end{aligned}
$$

In a minimization problem (such as the usual formulation of TIMES), the difference c_j - $y^*A'_j$ is called the *reduced cost* of technology *j.* The following holds:

- If x^*_j is at its lower bound, its unit cost c_j is *larger* than the value created (i.e. its reduced cost is positive). The technology is not competitive (and stays at its lower bound in the equilibrium);
- if x^*_j is at its upper bound, its cost c_j is *less* than the value created (i.e. its reduced cost is negative). The technology is super competitive and produces a profit; and
- if x^*_j is basic, its reduced cost is equal to 0. The technology is competitive but does not produce a profit

The reduced costs/surpluses may thus be used to rank all technologies, *including those that are not selected by the model.*

15 References

Altdorfer, F., (1981) , "Introduction of price elasticities on energy demand in MARKAL", *Memorandum No 345, KFA*, Julich, July 1981.

Barreto, L. (2001), “Technological Learning in Energy Optimization Models and the Deployment of Emerging Technologies”, *Ph.D. Thesis No 14151, Swiss Federal Institute of Technology Zurich (ETHZ)*. Zurich, Switzerland.

Berger, C., R. Dubois, A. Haurie, E. Lessard, R. Loulou, and J.-Ph. Waaub (1992), "Canadian MARKAL: an Advanced Linear Programming System for Energy and Environment Modelling", *INFOR*, Vol. 20, 114-125, 1992.

Birge, J. R., and Rosa, C. H. (1996), “Incorporating Investment Uncertainty into Greenhouse Policy Models,” *The Energy Journal*, 17/1, 79-90.

Brooke, A, D. Kendrick, A. Meeraus, and R. Raman, 1998, *The Solver Manual of the GAMS: A USER'S GUIDE*, December 1998.

Chen, K., M. Crucini (2012), "Comparing General and Partial Equilibrium Approaches to the Study of Real Business Cycles", *Vanderbilt University Research paper,* June 2012 (kan.chen@vanderbilt.edu, mario.crucini@vanderbilt.edu)

Chvátal, V. (1983), *Linear Programming*, Freeman and Co, New-York, 1983

Clarke, L., J. Edmonds , V. Krey, R.Richels, S. Rose, M. Tavoni (2009), “International climate policy architectures: Overview of the EMF 22 International Scenarios”, *Energy Economics* **31**, S64–S81

Dantzig, G.B. (1963), *Linear programming and Extensions*, Princeton University Press, Princeton, New-Jersey, 1963.

de Walque, G., F. Smets, and R. Wouters (2005), “An Estimated Two-Country DSGE Model for the Euro Area and the US Economy,” *Technical Report*, European Central Bank, 2005.

Dorfman, R., P.A. Samuelson, and R.M. Solow (1987 reprint), *Linear programming and Economic Analysis*, McGraw-Hill, New-York, 1958-1987.

Drouet L., Edwards N.R. and A. Haurie (2004), “Coupling Climate and Economic Models in a Cost-Benefit Framework: A Convex Optimization Approach”. *Environmental Modeling and Assessment*.

Edmonds, J.A., H.M. Pitcher, D. Barns, R. Baron, and M.A. Wise (1991), "Modelling future greenhouse gas emissions: The second generation model description", in *Modelling global change*, L.R. Klein and Fu-Chen Lo eds, United Nations University Press, Tokyo-New-York-Paris, 1991, 292-362.

European Commission, *The GEM-E3 Model*, *General Equilibrium Model for Economy, Energy and Environment,* https://ec.europa.eu/jrc/en/gem-e3/model.

Fankhauser, S. (1994), "The social cost of GHG emissions: an expected value approach", *Energy Journal 15/2*, 157-184.

Fishbone L G, Giesen G, Hymmen H A, Stocks M, Vos H, Wilde, D, Zoelcher R, Balzer C, and Abilock H. (1983), “Users Guide for MARKAL: A Multi-period, Linear Programming Model for Energy Systems Analysis”, *BNL Upton, NY, and KFA, Julich, Germany*, BNL 51701.

Fishbone, L.G., and H. Abilock (1981), "MARKAL, A Linear Programming Model for Energy Systems Analysis: Technical Description of the BNL Version", *International Journal of Energy Research*, Vol. 5, 353-375.

Fragnière, E. and Haurie, A. (1996), "MARKAL-Geneva: A Model to Assess Energy-Environment Choices for a Swiss Canton", in C. Carraro and A. Haurie (eds.), *Operations Research and Environmental Management*, Kluwer Academic Books.

Gale, D. (1989), *The Theory of Linear Economic Models*, McGraw-Hill, New-York, 1960, 11th edition Feb. 10 1989

Gerking. H., and Voss, A. (1986), “An Approach on How to Handle Incomplete Foresight in Linear Programming Models*,” International Conference on Models and Uncertainty in the Energy Sector*, Risö National Laboratory, Denmark, 11-12 February, 1986, p. 145.

G. Goldstein, G., 1991, “PC-MARKAL and the MARKAL User’s Support System (MUSS)”, BNL 46319, April 1991

Gorenstin, B. G. et al. (1993), “Power System Expansion Planning Under Uncertainty,” *IEEE Transactions on Power Systems*, **8**, 1, pp 129-136.

Grubb, M., (1993), "Policy Modelling for Climate Change: The Missing Models," *Energy Policy*, 21, 3, pp 203-208.

Hillier, F. S, and G. L. Lieberman (1990), *Introduction to Operations Research, Fifth Edition,* McGraw-Hill, New-York, 1990.

Hogan, W.W., (1975), "Energy Policy Models for Project Independence", *Computers and Operations Research,* 2, 251-271.

IPCC, 1996, *Climatic change (1995): The Science of Climate Change*. Intergovernmental Panel on Climate Change, Second Assessment Report, Working Group I. Cambridge : Cambridge University Press. pp.572.

IPCC, 2001, *Climatic change 2001: The Scientific Basis,* Intergovernmental Panel on Climate Change, Third Assessment Report, Working Group I.

IPCC, 2007, *Climate Change, Fourth Assessment Report, WG III: Mitigation*, B. Metz, O. Davidson, P. Bosch, R. Dave, L. Meyer, editors, Cambridge : Cambridge.

Jaccard, M., J. Nyboer, C. Bataille, and B. Sadownik (2003). "Modelling the Cost of Climate Policy: Distinguishing between alternative cost definitions and long-run cost dynamics*." Energy Journal* 24(1): 49-73.

Johansen, Leif (1960), *A Multi-Sectoral Study of Economic Growth,* North-Holland (2nd enlarged edition 1974).

Kanudia, A., R. Gerboni, R. Loulou, M. Gargiulo, M. Labriet, E. Lavagno, R. De Miglio, L.Schranz, G. C. Tosato, “Modelling EU-GCC energy systems and trade corridors: Long term sustainable, clean and secure scenarios*”, International Journal of Energy Management*, **7**, 2, 2013, 243 – 268.

Kanudia, A., and R. Loulou, “Robust Responses to Climate Change via Stochastic MARKAL: the case of Québec”, *European Journal of Operations Research*, vol. 106, pp. 15-30, 1998

Kunsch, P. L. and Teghem, J., Jr. (1987), "Nuclear Fuel Cycle Optimization Using Multi-Objective Stochastic Linear Programming," *European Journal of Operations Research 31/2*, 240-249.

Kypreos, S. (2006), *An Algorithm to Decompose the Global MARKAL-MACRO (GMM) and TIMES Models*, Internal document for ETSAP, Paul Scherrer Institut.

Kypreos, S., and A. Lehtila, 2013, "TIMES-Macro: Decomposition into Hard-Linked LP and NLP Problems", Technical note, http://www.etsap.org/documentation.asp.

Larsson, T. and Wene, C.-O. (1993), "Developing Strategies for Robust Energy Systems. I: Methodology," *International Journal of Energy Research 17*, 503-513.

Larsson, T. (1993), "Developing Strategies for Robust Energy Systems. II: Application to CO_2 Risk Management," *International Journal of Energy Research 17*, 505-535.

Loulou, R., and A. Kanudia (1999), "Minimax Regret Strategies for Greenhouse Gas Abatement: Methodology and Application", *Operations Research Letters,* 25, 219-230, 1999.

Loulou, R. and A. Kanudia (2000), "Using Advanced Technology-rich models for Regional and Global Economic Analysis of GHG Mitigation", in *Decision and Control: Essays in honor of Alain Haurie,* G. Zaccour ed., Kluwer Academic Publishers, Norwell, USA, pp.153-175, 2000. A condensed version of this article was published electronically in the *Proceedings of the International Energy Agency International conference on Climate Change Modelling and Analysis*, held in Washington DC, June15-17, 1999.

Loulou, R., Goldstein, G. & Noble, K. (2004), *Documentation for the MARKAL Family of Models: Part II – MARKAL-MACRO*. October 2004. http://www.etsap.org/documentation.asp

Loulou, R., and M. Labriet (2007), "ETSAP-TIAM: The TIMES Integrated Assessment Model --Part I: Model Structure", *Computational Management Science special issue on Energy and Environment*, Vol. 5, No 1-2, pp. 7-40.

Loulou, R., M. Labriet, A. Haurie, A. Kanudia, 2007, "OPEC Oil Pricing Strategies in a Climate Regime: a Two-Level Optimization Approach in an Integrated Assessment Model", special report for the ERMITAGE Project, European Commission.

Loulou, R., (2007), "ETSAP-TIAM: The TIMES Integrated Assessment Model --Part II: Mathematical Formulation", *Computational Management Science special issue on Energy and Environment* , Vol. 5, No 1-2, pp. 41-66

Loulou, R., and Lavigne, D. (1996), "MARKAL Model with Elastic Demands: Application to GHG Emission Control", in *Operations Research and Environmental*

Engineering, C. Carraro and A. Haurie eds., Kluwer Academic Publishers, Dordrecht, Boston, London, 1996, pp. 201-220.

Manne, A. S., (1977), "ETA-MACRO: A model of energy-economy interactions"*NASA STI/Recon Technical Report N 11/1977*; 78:26612, Stanford Univ., CA.

Manne, A.S., (1999),"Greenhouse gas abatement: toward Pareto-optimality in integrated assessment". In: Arrow KJ, Cottle RW, Eaves BC, Olkin I (eds) *Education in a research university*. Springer, Dordrecht

Manne, A., Mendelsohn, R. and Richels, R. (1995), "MERGE: a Model for Evaluating Regional and Global Effects of GHG Reduction Policies", *Energy Policy* 23 (1), 17-34.

Manne, A.S., and Wene, C-O. (1992), "MARKAL-MACRO: A Linked Model for Energy-Economy Analysis", *BNL-47161 report, Brookhaven National Laboratory*, Upton, New-York, February 1992.

Meade, D. (1996) "LIFT User's Guide", Technical report, INFORUM, IERF, College Park, Maryland (July 1996).

Messner, S. and Strubegger, M. (1995*), User's Guide for MESSAGE III*, WP-95-69, IIASA, Luxembourg, Austria.

Moomaw, W.R. and J.R. Moreira (2001)," Technological and Economic Potential of Greenhouse Greenhouse Gas Emissions Reduction". In *Climate Change 2001: Mitigation*, edited by Metz, B., Davidson, O., Swart, R. and J. Pan, Intergovernmental Panel on Climate Change (IPCC*), Third Assessment Report*, Working Group III, p.167-299. Cambridge : Cambridge University Press.

Nakicenovic, N. (ed.) (2000), Special Report on Emissions Scenarios. A Special Report of Working III of the Intergovernmental Panel on Climate Change.Cambridge : Cambridge University Press, p.599.

Negishi, T. (1960), "Welfare eoconomics and existence of an equilibrium for a competitive economy", *Metroeconomic*a, 12, 2-3, pp. 92–97, June 1960.

Nemhauser, G.L., A.H.G. Rinnooy Kan, and M.J. Todd editors (1989), *Handbooks in Operations Research and Management Science*, *Vol I: Optimization*, North-Holland, 1989.

Noble-Soft Systems, 2009, "ANSWERv6-TIMES User Manual", March 2009

Nordhaus, W. (1993), "Rolling the DICE: an optimal transition path for controlling GHG's", *Resources and Energy Economics 15/1*, 27-50.

Nordhaus, W. D. and J. Boyer (1999), Roll the DICE Again: Economic Models of Global Warming. Yale University, manuscript edition.

Papadimitriou, C.H., and K. Steiglitz (1982), *Combinatorial Optimization -- Algorithms and Complexity*, Prentice-Hall, New-Jersey, 1982.

Peck, S. C., and Teisberg,T. J. (1995), "Optimal CO2 Control Policy with Stochastic Losses from Temperature Rise," *Climatic change*, 31/1, 19-34.

Raiffa, H. (1968), *Decision Analysis*, Addison-Wesley, Reading, Mass., 1968

Remme, U. & Blesl, M. (2006), *Documentation of the TIMES-MACRO model*. Energy Technology Systems Analysis Programme (ETSAP), February 2006. http://www.etsap.org/documentation.asp

Rockafellar, R. T., (1970), *Convex Analysis*, Princeton University Press, Princeton, New-Jersey, 1970

Rutherford, Thomas F. (1992), "Sequential Joint Maximization", *Discussion Papers in Economics* – 92-08; Boulder, University of Colorado, September 1992.

Samuelson, P.A. (1952), "Spatial Price Equilibrium and Linear Programming", *American Economic Review*, 42, 283-303, 1952.

Samuelson, P.A., and W. Nordhaus (1977), *Economics (17th edition)*, John Wiley, 1977

Schrijver, A. (1986), *Theory of Linear and Integer Programming* , John Wiley and Sons.

Smets, F. and R. Wouters (2007), "Shocks and Frictions in US Business Cycles: a Bayesian DSGE Approach." *American Economic Review* **97**(3):586–606.

Stanton E. (2010), "Negishi welfare weights in integrated assessment models: the mathematics of global inequality", *Climatic Change*, published first online December 16, 2010.

Takayama, T., and Judge G.G. (1971), *Spatial and Temporal Price and Allocation Models*, North Holland, Amsterdam, 1971.

Tosato, G.C. (1980), "Extreme Scenarios in MARKAL LP Model: use of Demand Elasticity", presented at the *5th Italian-Polish Symposium on Applications of Systems Theory to Economics and Technology*, Torun, June 11-16 1980.

U.S. Energy Information Administration, 2000, http://pubsonline.informs.org/doi/pdf/10.1287/opre.49.1.14.11195

U.S. Department of Energy/Energy Information Administration, US Government (2000), "The National Energy Modeling System: An overview (2000)", *DOE/EIA-0581*, Washington. DC, March 2000.

U.S. Energy Information Administration (2002), *System for the Analysis of Global Energy Markets (SAGE)*, US Dept of Energy.

Van Regemorter, D., 1998, "Development of MARKAL Towards a Partial Equilibrium Model", ETSAP, December 1998

Voort, E. van der, Donni, E., Thonet, C., Bois d'Enghien, E., Dechamps, C. & Guilmot, J.F. (1984), *Energy Supply Modelling Package EFOM-12C Mark I, Mathematical description*. Louvain-la-Neuve, Cabay: Commission of the European Communities, EUR-8896.

Weistroffer, H. R., Smith, C. H., and Narula, S. C., 2005, "Multiple criteria decision support software", Ch 24 in: Figueira, J., Greco, S., and Ehrgott, M., eds, *Multiple Criteria Decision Analysis: State of the Art Surveys Series*, Springer: New York, 2005

Wets, R. J. B. (1989), 'Stochastic Programming'. In: Nemhauser George L., Rinnooy Kan, Alexander, H.G., Todd Michael J. (eds.) *Handbooks in OR and MS Vol. 1*, Elsevier Science Publishers, Amsterdam.

Wigley, T.M.L., Solomon, M. and S.C.B. Raper(1994), *Model for the Assessment of Greenhouse-Gas Induced Climate Change. Version 1.2.* Climate Research Unit, University of East Anglia, UK.

Energy Technology Systems Analysis Programme

http://www.iea-etsap.org/web/Documentation.asp

Documentation for the TIMES Model

PART II

July 2016

Authors:

Richard Loulou
Antti Lehtilä
Amit Kanudia
Uwe Remme
Gary Goldstein

General Introduction to the TIMES Documentation

This documentation is composed of five Parts.

Part I provides a general description of the TIMES paradigm, with emphasis on the model's general structure and its economic significance. Part I also includes a simplified mathematical formulation of TIMES, a chapter comparing it to the MARKAL model, pointing to similarities and differences, and chapters describing new model options.

Part II constitutes a comprehensive reference manual intended for the technically minded modeler or programmer looking for an in-depth understanding of the complete model details, in particular the relationship between the input data and the model mathematics, or contemplating making changes to the model's equations. Part II includes a full description of the sets, attributes, variables, and equations of the TIMES model.

Part III describes the organization of the TIMES modeling environment and the GAMS control statements required to run the TIMES model. GAMS is a modeling language that translates a TIMES database into the Linear Programming matrix, and then submits this LP to an optimizer and generates the result files. Part III describes how the routines comprising the TIMES source code guide the model through compilation, execution, solve, and reporting; the files produced by the run process and their use; and the various switches that control the execution of the TIMES code according to the model instance, formulation options, and run options selected by the user. It also includes a section on identifying and resolving errors that may occur during the run process.

Part IV provides a step-by-step introduction to building a TIMES model in the VEDA-Front End (VEDA-FE) model management software. It first offers an orientation to the basic features of VEDA-FE, including software layout, data files and tables, and model management features. It then describes in detail twelve Demo models (available for download from the ETSAP website) that progressively introduce VEDA-TIMES principles and modeling techniques.

Part V describes the VEDA Back-End (VEDA-BE) software, which is widely used for analyzing results from TIMES models. It provides a complete guide to using VEDA-BE, including how to get started, import model results, create and view tables, create and modify user sets, and step through results in the model Reference Energy System. It also describes advanced features and provides suggestions for best practices.

PART II: REFERENCE MANUAL

TABLE OF CONTENTS FOR PART II

1 Introduction

The purpose of the Reference Manual is to lay out the full details of the TIMES model, including data specification, internal data structures, and mathematical formulation of the model's Linear Program (LP) formulation, as well as the Mixed Integer Programming (MIP) formulations required by some of its options. As such, it provides the TIMES modeller/programmer with sufficiently detailed information to fully understand the nature and purpose of the data components, model equations and variables. A solid understanding of the material in this Manual is a necessary prerequisite for anyone considering making programming changes in the TIMES source code.

The Reference Manual is organized as follows:

Chapter 1	Basic notation and conventions: lays the groundwork for understanding the rest of the material in the Reference Manual;
Chapter 2	Sets: explains the meaning and role of various sets that identify how the model components are grouped according to their nature (e.g. demand devices, power plants, energy carriers, etc.) in a TIMES model;
Chapter 3	Parameters: elaborates the details related to the user-provided numerical data, as well as the internally constructed data structures, used by the model generator (and report writer) to derive the coefficients of the LP matrix (and prepare the results for analysis);
Chapter 4	Usage notes on special types of processes: Gives additional information about using input sets and parameters for the modelling of special types of processes: CHP, inter-regional exchange, and storage processes;
Chapter 5	Variables: defines each variable that may appear in the matrix, both explaining its nature and indicating how it fits into the matrix structure;
Chapter 6	Equations: states each equation in the model, both explaining its role and providing its explicit mathematical formulation. Includes user constraints that may be employed by modellers to formulate additional linear constraints, which are not part of the generic constraint set of TIMES.
Appendix A	The Climate Module;
Appendix B	The Damage Cost Functions, and
Appendix C	The Endogenous Technological Learning capability.

1.1 Basic notation and conventions

To assist the reader, the following conventions are employed consistently throughout this chapter:

- Sets, and their associated index names, are in lower and bold case, e.g., **com** is the set of all commodities;
- Literals, explicitly defined in the code, are in upper case within single quotes (note that in conformity with the GAMS syntax, single quotes must, in fact, be apostrophes), e.g., 'UP' for upper bound;
- Parameters, and scalars (constants, i.e., un-indexed parameters) are in upper case, e.g., NCAP_AF for the availability factor of a technology;
- Variables are in upper case with a prefix of VAR_, e.g., VAR_ACT corresponds to the activity level of a technology.

- Equations are in upper case with a prefix of EQ_ or EQ(l)_ with the placeholder (l) denoting the equation type (l=E for a strict equality, l=L for an inequality with the left hand side term being smaller than or equal to the right hand side term and l=G for an inequality with the left hand side term being greater than or equal to the right hand side term), e.g., EQ_PTRANS is the process transformation equation (strict equality), and EQG_COMBAL is the commodity balance constraint of type G (inequality).

1.2 GAMS modelling language and TIMES implementation

TIMES consists of generic variables and equations constructed from the specification of sets and parameter values depicting an energy system for each distinct region in a model. To construct a TIMES model, a preprocessor first translates all data defined by the modeller into special internal data structures representing the coefficients of the TIMES matrix applied to each variable of Chapter 5 for each equation of Chapter 6 in which the variable may appear. This step is called Matrix Generation. Once the model is solved (optimised) a Report Writer assembles the results of the run for analysis by the modeller. The matrix generation, report writer, and control files are written in GAMS[1] (the General Algebraic Modelling System), a powerful high-level language specifically designed to facilitate the process of building large-scale optimisation models. GAMS accomplishes this by relying heavily on the concepts of sets, compound indexed parameters, dynamic looping and conditional controls, variables and equations. Thus there is very a strong synergy between the philosophy of GAMS and the overall concept of the RES specification embodied in TIMES, making GAMS very well suited to the TIMES paradigm.

Furthermore, by nature of its underlying design philosophy, the GAMS code is very similar to the mathematical description of the equations provided in Chapter 6. Thus, the approach taken to implement a TIMES model is to “massage” the input data by means of a (rather complex) preprocessor that handles the necessary exceptions that need to be taken into consideration to properly construct the matrix coefficients in a form ready to be applied to the appropriate variables in the respective equations. GAMS also integrates seamlessly with a wide range of commercially available optimisers that are charged with the task of solving the actual TIMES linear (LP) or mixed integer (MIP) problems that represent the desired model. This step is called the Solve or Optimisation step. CPLEX or XPRESS are the optimisers most often employed to solve the TIMES LP and MIP formulations.

The standard TIMES formulation has optional features, such as lumpy investments and endogenous technology learning. The organization and layout of the TIMES code, along with how it is processed by GAMS during a model run, is discussed in detail in PART III. In addition, a modeller experienced in GAMS programming and the details of the TIMES implementation could define additional equation modules or report routine modules based on this organization, which allows the linkage of these modules to the standard TIMES code in a flexible way. However, any thoughts of modifying the core TIMES code should be discussed and coordinated with ETSAP.

To build, run, and analyse a TIMES model, several software tools have been developed in the past or are currently under development, so that the modeller does not need to provide the input information needed to build a TIMES model directly in GAMS. These tools are the model interfaces VEDA-FE and ANSWER-TIMES, as well as the reporting and analysing tool VEDA-BE.

[1] *GAMS A User’s Guide*, A. Brooke, D. Kendrick, A. Meeraus, R. Raman, GAMS Development Corporation, December 1998.

2 Sets

Sets are used in TIMES to group elements or combinations of elements with the purpose of specifying qualitative characteristics of the energy system. One can distinguish between one-dimensional and multi-dimensional sets. The former sets contain single elements, e.g. the set **prc** contains all processes of the model, while the elements of multi-dimensional sets are a combination of one-dimensional sets. An example for a multi-dimensional set is the set **top**, which specifies for a process the commodities entering and leaving that process.

Two types of sets are employed in the TIMES framework: user input sets and internal sets. User input sets are created by the user, and used to describe qualitative information and characteristics of the depicted energy system. One can distinguish the following functions associated with user input sets:

- definition of the elements or building blocks of the energy system model (i.e. regions, processes, commodities),
- definition of the time horizon and the sub-annual time resolution,
- definition of special characteristics of the elements of the energy system.

In addition to these user sets, TIMES also generates its own internal sets. Internal sets serve to both ensure proper exception handling (e.g., from what date is a technology available, or in which time-slices is a technology permitted to operate), as well as sometimes just to improve the performance or smooth the complexity of the actual model code.

In the following sections, the user input sets and the internal sets will be presented. A special type of set is a one-dimensional set, also called index, which is needed to build multi-dimensional sets or parameters. At the highest level of the one-dimensional sets are the master or "domain" sets that define the comprehensive list of elements (e.g., the main building blocks of the reference energy system such as the processes and commodities in all regions) permitted at all other levels, with which GAMS performs complete domain checking, helping to automatically ensure the correctness of set definition (for instance, if the process name used in a parameter is not spelled correctly, GAMS will issue a warning). Therefore, before elaborating on the various sets, the indexes used in TIMES are discussed.

2.1 Indexes (One-dimensional sets)

Indexes (also called one-dimensional sets) contain in most cases the different elements of the energy model. A list of all indexes used in TIMES is given in Table 2. Examples of indexes are the set **prc** containing all processes, the set **c** containing all commodities, or the set **all_reg** containing all regions of the model. Some of the one-dimensional sets are subsets of another one-dimensional set, e.g., the set **r** comprising the so-called internal model regions is a subset of the set **all_reg,** which in addition also contains the so-called external model regions[2]. To express that the set **r** depends on the set **all_reg**, the master set **all_reg** is put in brackets after the set name **r**: **r**(**all_reg**).

The set **cg** comprises all commodity groups[3]. Each commodity c is considered as a commodity group with only one element: the commodity itself. Thus the commodity set **c** is a subset of the commodity group set **cg**.

Apart from indexes that are under user control, some indexes have fixed elements to serve as indicators within sets and parameters, most of which should not be modified by the user (Table 1). Exceptions to this rule are the sets defined in the file MAPLISTS.DEF. For

[2] The meaning and the role of internal and external regions is discussed in Section 2.2.

[3] See Section 2.2.1 for a more in-depth treatment of commodity groups.

example, while the process groups (**prc_grp)** listed in Table 1 are used within the code and must not be deleted, other process groups may be added by the user.

Table 1: Sets with fixed elements

Set/Index name	Description
Sets defined in INITSYS.MOD (never to be changed by the user)	
bd(lim)	Index of bound type; subset of the set lim having the internally fixed elements 'LO', 'UP', 'FX'.
costagg	List of cost aggregation types available for user-defined cost constraints: INV investment costs INVTAX investment taxes INVSUB investment subsidies INVTAXSUB investment taxes and subsidies INVALL all investment costs, taxes and subsidies FOM fixed O&M costs FOMTAX fixed operating taxes FOMSUB fixed operating subsidies FOMTAXSUB fixed operation taxes and subsidies FOMALL all fixed operation costs, taxes and subsidies COMTAX commodity taxes COMSUB commodity subsidies COMTAXSUB commodity taxes and subsidies FLOTAX taxes FLOSUB subsidies FLOTAXSUB flow taxes and subsidies FIX total fixed costs (investment+fixed O&M costs) FIXTAX total fixed taxes FIXSUB total fixed subsidies FIXTAXSUB total fixed taxes and subsidies FIXALL all fixed costs, taxes and subsidies ALLTAX all taxes ALLSUB all subsidies ALLTAXSUB all taxes and subsidies
ie	Export/import exchange index; internally fixed to the two elements: 'IMP' standing for import and 'EXP' standing for export.
io	Input/Output index; internally fixed elements: 'IN', 'OUT'; used in combination with processes and commodities as indicator whether a commodity enters or leaves a process.
lim	Index of limit types; internally fixed to the elements 'LO', 'UP', 'FX', 'N'.
side	Index of constraint sides; internally fixed to the elements 'LHS', 'RHS'
tslvl	Index of timeslice levels; internally fixed to the elements 'ANNUAL', 'SEASON', 'WEEKLY', 'DAYNITE'.
uc_grptype	Index of internally fixed key types of variables: ACT, FLO, IRE, CAP, NCAP, COMPRD, COMNET, COMCON, UCN These are used in association with the user constraints.
uc_cost	Internally fixed list of cost types that can be used as modifier attributes in user constraints: COST, DELIV, TAX, SUB
uc_name	List of internally fixed indicators for attributes able to be referenced as coefficients in user constraints (e.g. the flow variable may be multiplied by the attribute FLO_COST in a user constraint if desired): COST, DELIV, TAX, SUB, EFF, NET, BUILDUP, CAPACT, CAPFLO, GROWTH, NEWFLO, ONLINE, PERIOD, PERDISC, INVCOST, INVTAX, INVSUB, CUMSUM, SYNC, YES See Section 6.4.6 for more detailed information.

Set/Index name	Description
Sets defined in MAPLIST.DEF (additional elements may be added by user)	
com_type	Indicator of commodity type; initialized to the following elements: DEM demand NRG energy MAT material ENV environment FIN financial The predefined elements should never be deleted.
dem_sect	List of demand sectors; internally established in MAPLIST.DEF as: AGR agriculture RES residential COM commercial and public services IND industry TRN transport NE non-energy OTH other The predefined elements should not be deleted.
env_type	List of emission types; internally established in MAPLIST.DEF as: GHG greenhouse gas PEM particulate matter emissions OEM other emissions into air or water OTHENV other environmental indicator
nrg_type	List of energy types; internally established in MAPLIST.DEF as: FOSSIL fossil fuel NUCLR nuclear SYNTH synthetic fuel FRERENEW free renewable LIMRENEW limited renewable (no commodity balance) ELC electricity HTHEAT high temperature heat LTHEAT low temperature heat CONSRV conservation The predefined elements should not be deleted.
prc_grp	List of process groups; internally established in MAPLIST.DEF as: CHP combined heat and power plant DISTR distribution process DMD demand device ELE electricity producing technology excluding CHP HPL heat plant IRE inter-regional exchange process MISC miscellaneous PRE energy technology not falling in other groups PRV technology with material output measured in volume units PRW technology with material output measured in weight units REF refinery process RENEW renewable energy technology XTRACT extraction process NST night (off-peak) storage process STG storage process (timeslice storage, unless also STK/NST) STK stockpiling process (inter-period storage) STS generalized timeslice storage The user may augment this list with any additional groups desired. The following predefined groups affect the data processing carried out by the model generator, and should not be deleted by the user: CHP, DISTR, DMD, ELE, HPL, IRE, PRE, PRV, PRW, REF, NST, STG, STK and STS.

Table 2: Indexes in TIMES

Index[4]	**Aliases**[5]	**Related Indexes**[6]	**Description**
age	life, jot		Index for age (number of years since installation) into a parameter shaping curve; default elements 1–200.
all_r	all_reg	r	All internal and external regions.
bd	bnd_type	lim	Index of bound type; subset of lim, having the internally fixed elements ‘LO’, ‘UP’, ‘FX’.
c(cg)	com, com1, com2, com3	cg	User defined7 list of all commodities in all regions; subset of cg.
cg	com_grp, cg1, cg2, cg3, cg4	c	User defined list of all commodities and commodity groups in all regions[8]; each commodity itself is considered a commodity group; initial elements are the members of com_type.
com_type			Indicator of commodity type; initialized to the predefined types DEM, NRG, MAT, ENV, FIN (see Table 1).
costagg			Indicator of cost aggregation type; initialized to list of predefined types (see Table 1).
cur	cur		User defined list of currency units.
datayear (year)		year	Years for which model input data are specified.
dem_sect			Indicator of demand sector; initialized to list of predefined sectors (see Table 1);
env_type			Indicator of environmental commodity type; initialized to list of predefined elements (see Table 1);
ie	impexp		Export/import exchange indicator; internally fixed = 'EXP' for exports and 'IMP' for imports.
io	inout		Input/Output indicator for defining whether a commodity flow enters or leaves a process; internally fixed = 'IN' for enters and 'OUT' for leaves.
j	jj		Indicator for elastic demand steps and sequence number of the shape/multi curves; default elements 1–999.
kp			Index for “kink” points in ETL formulation; currently limited to 1-6 {can be extended in <case>.run file by including SET KP / 1*n /; for n-kink points.
lim	lim_type, l	bd	Index of limit types; internally fixed = 'LO', 'UP', 'FX' and 'N'.
nrg_type			Indicator of energy commodity type; initialized to predefined types (see Table 1);
p	prc		User defined list of all processes in all regions[9].
pastyear	pyr	modlyear, year	Years for which past investments are specified; pastyears must be before the beginning of the first period.
prc_grp			List of process groups; internally established in MAPLIST.DEF (see Table 1).
r(all_r)	reg	all_r	Explicit regions within the area of study.

[4] This column contains the names of the indexes as used in this document.

[5] For programming reasons, alternative names (aliases) may exist for some indexes. This information is only relevant for those users who are interested in gaining an understanding of the underlying GAMS code.

[6] This column refers to possible related indexes, e.g. the index set **c** is a subset of the index set **cg**.

[7] VEDA/ANSWER compiles the complete list from the union of the commodities defined in each region.

[8] VEDA/ANSWER compiles the complete list from the union of the commodity groups defined in each region.

[9] VEDA/ANSWER compiles the complete list from the union of processes defined in each region.

Index[4]	Aliases[5]	Related Indexes[6]	Description
s	all_ts, ts, s2, sl		Timeslice divisions of a year, at any of the tslvl levels.
side			Side indicator for defining coefficients in user constraints; internally fixed = 'LHS', 'RHS'
t	milestonyr, tt	year	Representative years for the model periods.
teg		p	Technologies modelled with endogenous technology learning.
tslvl			Timeslice level indicator; internally fixed = 'ANNUAL', 'SEASON', 'WEEKLY', 'DAYNITE'.
u	units	units_com, units_cap, units_act	List of all units; maintained in the file UNITS.DEF.
uc_grptype			Fixed internal list of the key types of variables (see Table 1).
uc_n	ucn		User specified unique indicator for a user constraint.
uc_name			Fixed list of indicators associated with various attributes that can be referenced in user constraints to be applied when deriving a coefficient (see Table 1).
unit			List of capacity blocks that can be added in lumpy investment option; default elements 0–100; the element '0' describes the case when no capacity is added.
units_act		u	List of activity units; maintained in the file UNITS.DEF.
units_cap		u	List of capacity units; maintained in the file UNITS.DEF.
units_com		u	List of commodity units; maintained in the file UNITS.DEF.
v(year)	modlyear	pastyear, t	Union of the set pastyear and t corresponding to all modelling periods.
ww	allsow	sow, w	States of the world that can be used; default elements 1–64; under user control by the dollar control parameter $SET MAXSOW <n> in the <case>.RUN file
year	allyear, ll	datayear, pastyear, modlyear, milestonyr	Years that can be used in the model; default range 1850-2200; under user control by the dollar control parameters $SET BOTIME <y> and $SET EOTIME <y> in the <case>.RUN file.

2.2 User input sets

The user input sets contain the fundamental information regarding the structure and the characteristics of the underlying energy system model. The user input sets can be grouped according to the type of information related to them:

- One dimensional sets defining the components of the energy system: regions, commodities, processes;
- Sets defining the Reference Energy System (RES) within each region;
- Sets defining the inter-connections (trade) between regions;
- Sets defining the time structure of the model: periods, timeslices, timeslice hierarchy;
- Sets defining various properties of processes or commodities.

The formulation of user constraints also uses sets to specify the type and the features of a constraint. The structure and the input information required to construct a user constraint is covered in detail in Chapter 6, and therefore will not be presented here.

Most of the set specifications are handled for the user by the user shell through process and commodity characterization, and the user does not need to input these sets directly.

In the following subsections first the sets related to the definition of the RES will be described (subsection 2.2.1), then the sets related to the time horizon and the sub-annual representation of the energy system will be presented (subsection 2.2.2). The mechanism for defining trade between regions of a multi-regional model is discussed in subsection 2.2.3. Finally, an overview of all possible user input sets is given in subsection 2.2.4.

2.2.1 Definition of the Reference Energy System (RES)

A TIMES model is structured by regions (**all_r**). One can distinguish between external regions and internal regions. The internal regions (**r**) correspond to regions within the area of study, and for which a RES has been defined by the user. Each internal region may contain processes and commodities to depict an energy system, whereas external regions serve only as origins of commodities (e.g. for import of primary energy resources or for the import of energy carriers) or as destination for the export of commodities. A region is defined as an internal region by putting it in the internal region set (**r)**, which is a subset of the set of all regions **all_r**. An external region needs no explicit definition, all regions that are member of the set **all_r** but not member of **r** are external regions. A TIMES model must consist of at least one internal region, the number of external regions is arbitrary. The main building blocks of the RES are processes (**p**) and commodities (**c)**, which are connected by commodity flows to form a network. An example of a RES with one internal region (UTOPIA) and two external regions (IMPEXP, MINRNW) is given in Figure 1.

All components of the energy system, as well as nearly the entire input information, are identified by a region index. It is therefore possible to use the same process name in different regions with different numerical data (and description if desired), or even completely different commodities associated with the process.

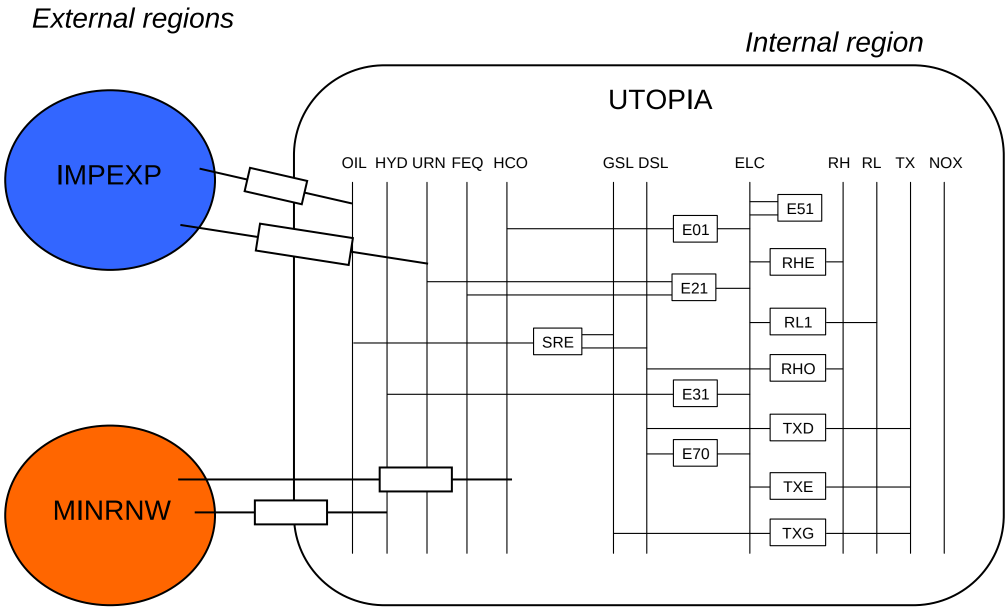


Figure 1: Example of internal and external regions in TIMES

2.2.1.1 Processes

A process may represent an individual plant, e.g. a specific existing nuclear power plant, or a generic technology, e.g. the coal-fired IGCC technology. TIMES distinguishes three main types of processes:

- Standard processes;
- Inter-regional exchange processes, and
- Storage processes.

2.2.1.1.1 Standard processes

The so-called standard processes can be used to model the majority of the energy technologies, e.g., condensing power plants, heat plants, CHP plants, demand devices such as boilers, coal extraction processes, etc. Standard processes can be classified into the following groups:

- PRE for generic energy processes;
- PRW for material processing technologies (by weight);
- PRV for material processing technologies (by volume);
- REF for refinery processes;
- ELE for electricity generation technologies;
- HPL for heat generation technologies;
- CHP for combined heat and power plants;
- DMD for demand devices;
- DISTR for distribution systems;
- MISC for miscellaneous processes.

The process classification is done via the set **prc_map(r,prc_grp,p)**. This grouping is mainly intended for reporting purposes, but in some cases it also affects the properties of the processes[10] and the constraint matrix. The set is maintained in the MAPLIST.DEF file, and may be adjusted by user with additional technology groups of interest, with some restrictions as noted in Table 1.

The topology of a standard process is specified by the set **top(r,p,c,io)** of all quadruples such that the process **p** in region **r** is consuming (**io** = 'IN') or producing (**io** = 'OUT') commodity **c**. Usually, for each entry of the topology set **top** a flow variable (see **VAR_FLO** in Chapter 5) will be created. When the so-called *reduction algorithm* is activated, some flow variables may be eliminated and replaced by other variables (see PART III, Section 3.7 for details).

The activity variable (**VAR_ACT**) of a standard process is in most cases equal to the sum of one or several commodity flows on either the input or the output side of a process. The activity of a process is limited by the available capacity, so that the activity variable establishes a link between the installed capacity of a process and the maximum possible commodity flows entering or leaving the process during a year or a subdivision of a year. The commodity flows that define the process activity are specified by the set **prc_actunt(r,p,cg,u)** where the commodity index **cg** may be a single commodity or a user-defined commodity group, and **u** is the activity unit. The commodity group defining the activity of a process is also called **P**rimary **C**ommodity **G**roup (PCG).

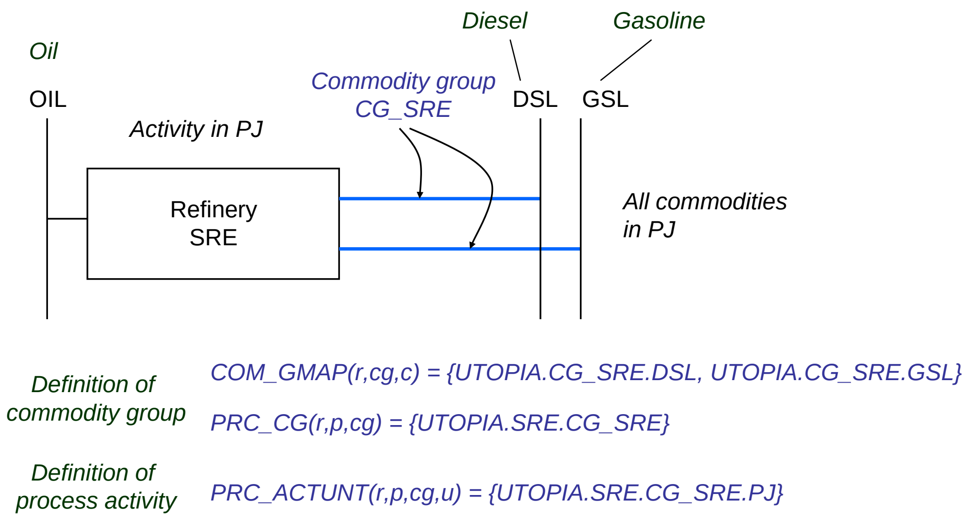


Figure 2: Example of the definition of a commodity group and the activity of a normal process.

User-defined commodity groups are specified by means of the set **com_gmap(r,cg,c)**, which indicates the commodities (c) belonging to the group (cg). Once a user-defined commodity group has been defined, one can use it for any processes for defining attributes that require a commodity group (not only for the definition of the process activity, but also for other purposes, e.g., in the transformation equation EQ_PTRANS), as long as the members of the group are valid for the particular process and the process characteristic to be defined.

An example for the definition of the activity of a process is shown in Figure 2. In order to define the activity of the process SRE as the sum of the two output flows of gasoline (GSL) and diesel (DSL), one has to define a commodity group called CG_SRE containing these two commodities. The name of the commodity group can be arbitrarily chosen by the modeller.

[10] Important cases are the process type CHP, which activates the CHP attributes, storage process indicators (STG, STS, STK, NST), and material conversion process types PRW and PRV, which may affect the creation of the internal set **prc_spg** (see Table 5).

In addition to the activity of a process, one has to define the capacity unit of the process. This is done by means of the set **prc_capunt(r,p,cg,u)**, where the index **cg** denotes the primary commodity group. In the example in Figure 3 the capacity of the refinery process is defined in mtoe/a (megatonne oil equivalent). Since the capacity and activity units are different (mtoe for the capacity and PJ for the activity), the user has to supply the conversion factor from the energy unit embedded in the capacity unit to the activity unit. This is done by specifying the parameter **prc_capact(r,p)**. In the example **prc_capact** has the value 41.868.

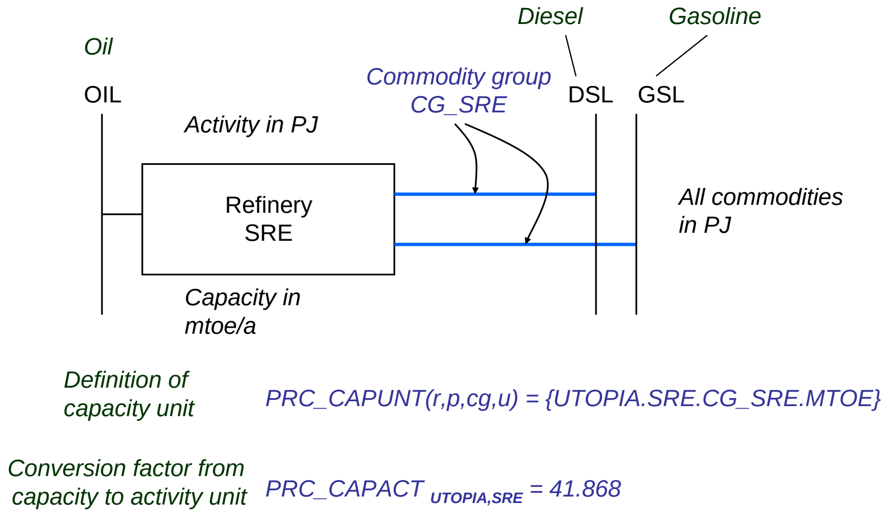


Figure 3: Example of the definition of the capacity unit

It might occur that the unit in which the commodity(ies) of the primary commodity group are measured, is different from the activity unit. An example is shown in Figure 4. The activity of the transport technology CAR is defined by commodity TX1, which is measured in passenger kilometres PKM. The activity of the process is, however, defined in vehicle kilometres VKM, while the capacity of the process CAR is defined as number of cars NOC.

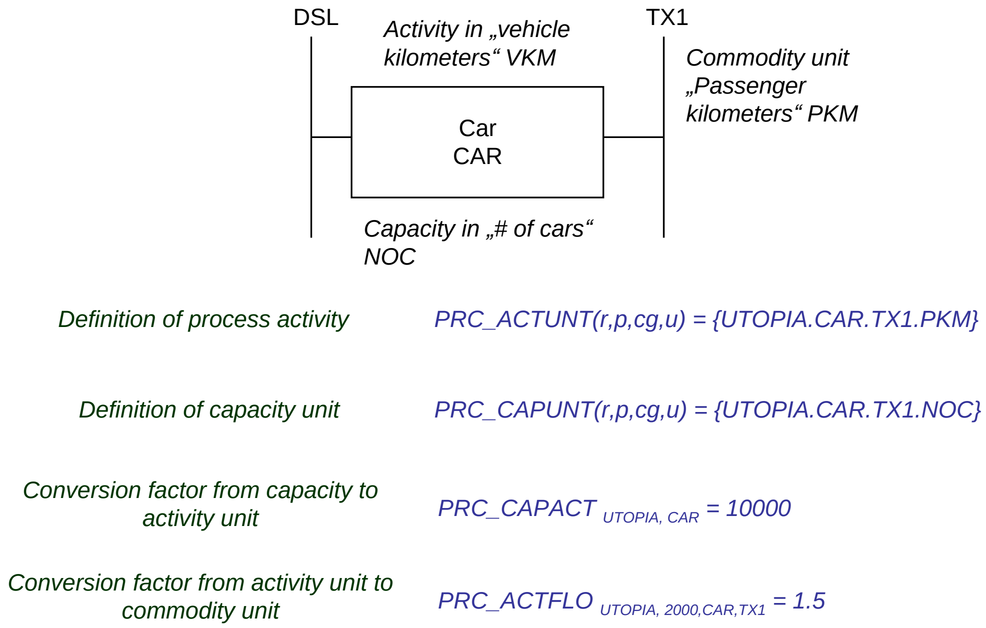


Figure 4: Example of different activity and commodity units

The conversion factor from capacity to activity unit **prc_capact** describes the average mileage of a car per year. The process parameter **prc_actflo(r,y,p,cg)** contains the conversion factor from the activity unit to the commodity unit of the primary commodity group. In the example this factor corresponds to the average number of persons per car (1.5).

2.2.1.1.2 Inter-regional exchange processes

Inter-regional exchange (IRE) processes are used for trading commodities between regions. They are needed for linking internal regions with external regions as well as for modelling trade between internal regions. A process is specified as an inter-regional exchange process by specifying it as a member of the set **prc_map(r,'IRE',p)**. If the exchange process is connecting internal regions, this set entry is required for each of the internal regions trading with region r. The topology of an inter-regional exchange process **p** is defined by the set **top_ire(all_reg,com,all_r,c,p)** stating that the commodity **com** in region **all_reg** is exported to the region **all_r** (the traded commodity may have a different name **c** in region **all_r** than in region **all_reg)**. For example the topology of the export of the commodity electricity (ELC_F) from France (FRA) to Germany (GER), where the commodity is called ELC_G via the exchange process (HV_GRID) is modelled by the **top_ire** entry:

top_ire('FRA', 'ELC_F', 'GER', 'ELC_G', 'HV_GRID')

The first pair of region and commodity ('FRA', 'ELC_F') denotes the origin and the name of the traded commodity, while the second pair ('GER', 'ELC_G') denotes the destination. The name of the traded commodity can be different in both regions, here 'ELC_F' in France and 'ELC_G' in Germany, depending on the chosen commodity names in both regions. As with standard processes, the activity definition set **prc_actunt(r,p,cg,u)** has to be specified for an exchange process belonging to each internal region. The special features related to inter-regional exchange processes are described in subsection 2.2.3.

2.2.1.1.3 Storage processes

Storage processes are used to store a commodity either between periods or between timeslices. A process (**p**) can be specified to be an inter-period storage (IPS) process for commodity (**c**) by defining the process to be of the type '**STK**' and **c** as its PCG (or, alternatively, including it as a member of the set **prc_stgips(r,p,c)**). In a similar way, a process is characterised as a timeslice storage by defining the process to be of the type '**STG**' and **c** as its PCG (alternatively, by inclusion in the set **prc_stgtss(r,p,c)**). A special case of timeslice storage is a so-called night-storage device (NST) where the commodity for charging and the one for discharging the storage are different. An example for a night storage device is an electric heating technology which is charged during the night using electricity and produces heat during the day. Including a process in the set **prc_nstts(r,p,s)** indicates that it is a night storage device which is charged in timeslice(s) **s**. More than one timeslice can be specified as charging timeslices, the non-specified timeslices are assumed to be discharging timeslices. The charging and discharging commodity of a night storage device are specified by the topology set (**top).** It should be noted that for inter-period storage and normal timeslice storage processes (non-NST) the commodity entering and leaving the storage (the charged and discharged commodity) should be a member of the PCG (and both should be, if they are different). Other auxiliary commodity flows are also permitted in combination with these two storage types, by including them in the topology (see Section 4.3.5).

As for standard processes, the flows that define the activity of a storage process are identified by providing the set **prc_actunt(r,p,c)** entry. In contrast to standard processes, the activity of a storage process is however interpreted as the amount of the commodity being stored in the storage process. Accordingly the capacity of a storage process describes the maximum commodity amount that can be kept in storage.

Internally, a **prc_map(r,'STG',p)** entry is always generated for all storage processes to put the process in the group of storage processes. A further **prc_map** entry is created to specify the type of storage ('STK' for inter-period storage, 'STS' for general time-slice storage and 'NST' for a night-storage device), unless already defined so by the user.

2.2.1.2 Commodities

As mentioned before, the set of commodities (**c**) is a subset of the commodity group set (**cg)**. A commodity in TIMES is characterised by its type, which may be an energy carrier ('NRG'), a material ('MAT'), an emission or environmental impact ('ENV'), a demand commodity ('DEM') or a financial resource ('FIN'). The commodity type is indicated by membership in the commodity type mapping set (**com_tmap(r,com_type,c)).** The commodity type affects the default sense of the commodity balance equation. For NRG, ENV and DEM the commodity production is normally greater than or equal to consumption, while for MAT and FIN the default commodity balance constraint is generated as an equality. The type of the commodity balance can be modified by the user for individual commodities by means of the commodity limit set (**com_lim(r,c,lim)**). The unit in which a commodity is measured is indicated by the commodity unit set (**com_unit(r,c,units_com)**). The user should note that within the GAMS code of TIMES no unit conversion, e.g., of import prices, takes place when the commodity unit is changed from one unit to another one. Therefore, the proper handling of the units is entirely the responsibility of the user (or the user interface).

2.2.2 Definition of the time structure

2.2.2.1 Time horizon

The time horizon for which the energy system is analysed may range from one year to many decades. The time horizon is usually split into several *periods* which are represented by so-called *milestone years* (**t(allyear)** or **milestonyr(allyear)**, see Figure 5). Each milestone year represents a point in time where decisions may be taken by the model, e.g. installation of new capacity or changes in the energy flows. The activity and flow variables used in TIMES may therefore be considered as average values over a period. The shortest possible duration of a period is one year. However, in order to keep the number of variables and equations at a manageable size, periods are usually comprised of several years. The durations of the periods do not have to be equal, so that it is possible that the first period, which usually represents the past and is used to calibrate the model to historic data, has a length of one year, while the following periods may have longer durations. Thus in TIMES both the number of periods and the duration of each period are fully under user control. The beginning year of a period **t**, **B(t)**, and its ending year, **E(t)**, have to be specified as input parameters by the user (see Table 13 in subsection 3.1.3).

To describe capacity installations that took place before the beginning of the model horizon, and still exist during the modeling horizon, TIMES uses additional years, the so-called *past years* (**pastyear(allyear)**), which identify the construction completion year of the already existing technologies. The amount of capacity that has been installed in a pastyear is specified by the parameter **NCAP_PASTI(r,allyear,p)**, also called *past investment*. For a process, an arbitrary number of past investments may be specified to reflect the age structure in the existing capacity stock. The union of the sets **milestonyr** and **pastyear** is called **modelyear** (or **v**). The years for which input data is provided by the user are called datayears (**datayear(allyear)**). The datayears do not have to coincide with modelyears, since the preprocessor will interpolate or extrapolate the data internally to the modelyears. All pastyears are by default included in datayears, but, as a general rule, any other years for which input data is provided should be explicitly included in the set **datayear** or that information will not be seen by the model. Apart from a few exceptions (see Table 3), all parameter values defined for years other than datayears (or pastyears) are ignored by the model generator. Due to the distinction between of modelyears and datayears, the definition of the model horizon, e.g., the duration and number of the periods, may be changed without having to adjust the input data to the new periods. The rules and options of the inter- and extrapolation routine are described in more detail in subection 3.1.1.

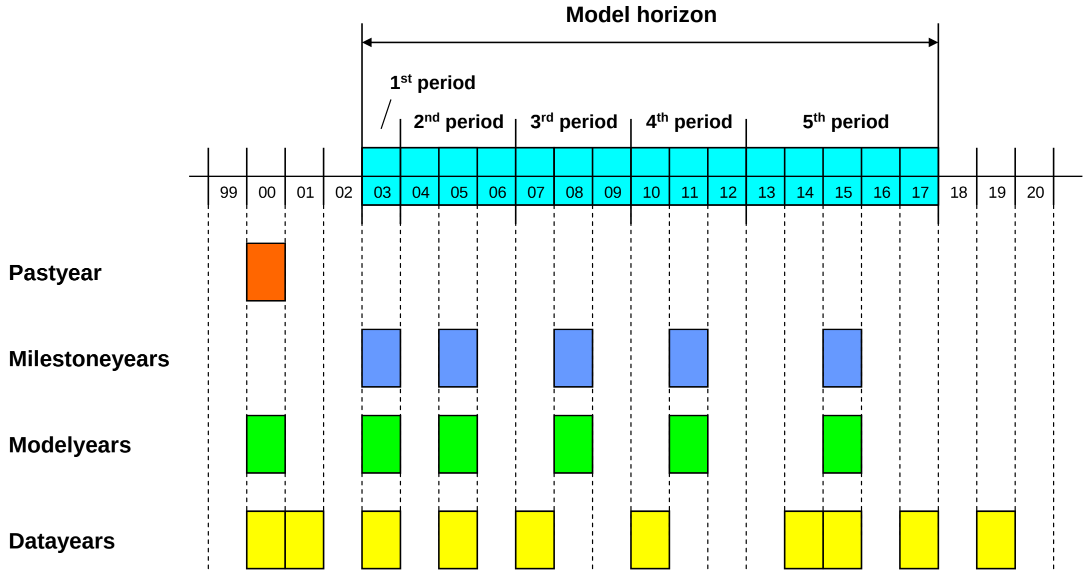


Figure 5: Definition of the time horizon and the different year types

One should note that it is possible to define past investments (**NCAP_PASTI**) not only for pastyears but also for any years within the model horizon, including the milestone years. Since the first period(s) of a model may cover historical data, it is useful to store the already known capacity installations made during this time-span as past investments and not as a bound on new investments in the model database. If one later changes the beginning of the model horizon to a more recent year, the capacity data of the first period(s) do not have to be changed, since they are already stored as past investments. This feature therefore supports the decoupling of the datayears, for which input information is provided, and the definition of the model horizon for which the model is run, making it relatively easy to change the definition of the modeling horizon. Defining past investments for years within the actual model horizon may also be useful for identifying already planned (although not yet constructed) capacity expansions in the near future[11].

[11] In this case the model may still decide to add additional new capacity, if this is economical and not inhibited by any investment bounds.

Table 3: Parameters that can have values defined for any year, irrespective of datayear[12]

Attribute name	Description
G_DRATE	General discount rate for currency in a particular year
MULTI	Parameter multiplier table with values by year
ACT_CUM	Cumulative limit on process activity
FLO_CUM	Cumulative limit on process flow
COM_CUMPRD	Cumulative limit on gross production of a commodity for a block of years
COM_CUMNET	Cumulative limit on net production of a commodity for a block of years
REG_CUMCST	Cumulative limit on regional costs, taxes or subsidies
UC_CUMACT	Coefficient for a cumulative amount of process activity in a user constraint
UC_CUMFLO	Coefficient for a cumulative amount of process flow in a user constraint
CM_EXOFORC	Radiative forcing from exogenous sources; included in the climate module extension (see Appendix A for a description of the climate module).
CM_HISTORY	Climate module calibration values; included in the climate module extension (see Appendix A for a description of the climate module).
CM_MAXC	Maximum level of climate variable; included in the climate module extension (see Appendix A for a description of the climate module).

2.2.2.2 Timeslices

The **milestone** years can be further divided in sub-annual timeslices in order to describe for the changing electricity load within a year, which may affect the required electricity generation capacity, or other commodity flows that need to be tracked at a finer than annual resolution. Timeslices may be organised into four hierarchy levels only: 'ANNUAL', 'SEASON', 'WEEKLY' and 'DAYNITE' defined by the internal set **tslvl**. The level ANNUAL consists of only one member, the predefined timeslice 'ANNUAL', while the other levels may include an arbitrary number of divisions. The desired timeslice levels are activated by the user providing entries in set **ts_group(r,tslvl,s)**, where also the individual user-provided timeslices **(s)** are assigned to each level. An additional user input set **ts_map(r,s1,s2)** is needed to determine the structure of a timeslice tree, where timeslice **s1** is defined as the parent node of **s2**. Figure 6 illustrates a timeslice tree, in which a year is divided into four seasons consisting of working days and weekends, and each day is further divided into day and night timeslices. The name of each timeslice has to be unique in order to be used later as an index in other sets and parameters. Not all timeslice levels have to be utilized when building a timeslice tree, for example one can skip the 'WEEKLY' level and directly connect the seasonal timeslices with the daynite timeslices. The duration of each timeslice is expressed as a fraction of the year by

[12] The purpose of this table is to list those parameters whose year values are independent of the input **datayear**s associated with most of the regular parameters, and therefore need not be included in the set **datayear**. For example, a value for MULTI(j,'2012') would not require including 2012 in **datayear**s if 2012 were not relevant to the other input parameters.

the parameter G_YRFR(r,s). The user is responsible for ensuring that each lower level group sums up properly to its parent timeslice, as this is not verified by the pre-processor. The definition of a timeslice tree is region-specific.[13] When different timeslice names and durations are used in two regions connected by exchange processes, the mapping parameters IRE_CCVT(r,c,reg,com) for commodities and IRE_TSCVT(r,s,reg,ts) for timeslices have to be provided by the user to map the different timeslice definitions. When the same timeslice definitions are used, these mapping tables do not need to be specified by the user.

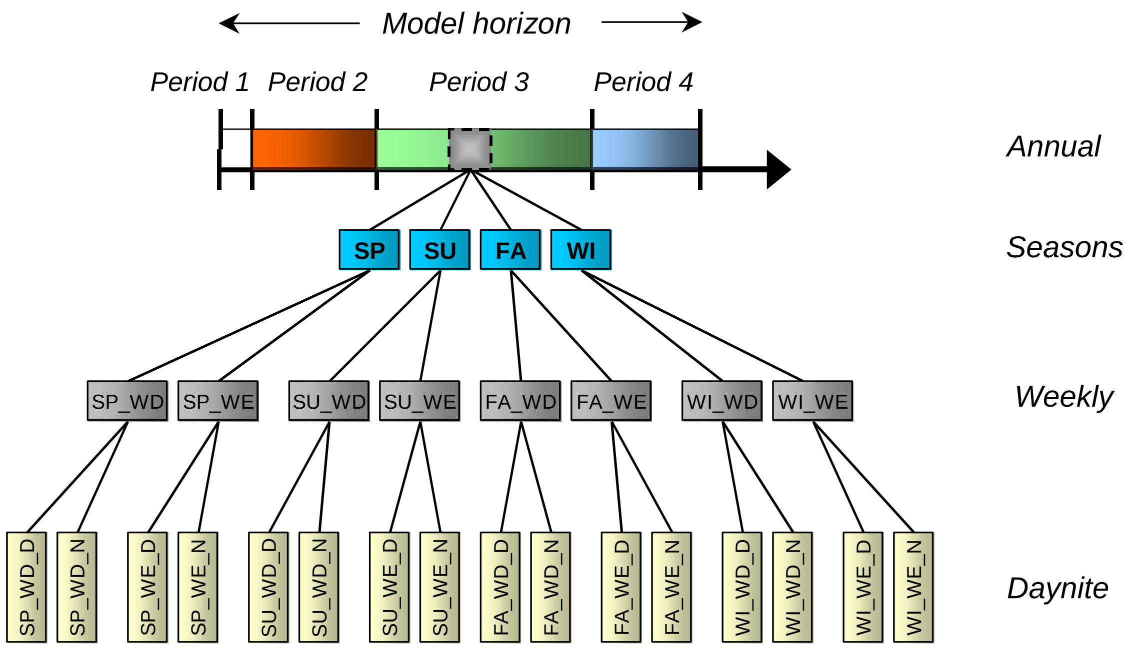


Figure 6: Example of a timeslice tree

Commodities may be tracked and process operation controlled at a particular timeslice level by using the sets **com_tsl(r,c,tslvl)** and **prc_tsl(r,p,tslvl)** respectively. Providing a commodity timeslice level determines for which timeslices the commodity balance will be generated, where the default is 'ANNUAL'. For processes, the set **prc_tsl** determines the timeslice level of the activity variable. Thus, for instance, condensing power plants may be forced to operate on a seasonal level, so that the activity during a season is uniform, while hydropower production may vary between days and nights, if the 'DAYNITE' level is specified for hydro power plants. Instead of specifying a timeslice level, the user can also identify individual timeslices for which a commodity or a process is available by the sets **com_ts(r,c,s)** and **prc_ts(r,p,s)** respectively. Note that when specifying individual timeslices for a specific commodity or process by means of **com_ts** or **prc_ts** they all have to be on the same timeslice level.

The timeslice level of the commodity flows entering and leaving a process are determined internally by the preprocessor. The timeslice level of a flow variable equals the timeslice level of the process when the flow variable is part of the primary commodity group (PCG) defining the activity of the process. Otherwise the timeslice level of a flow variable is set to whichever level is finer, that of the commodity or the process.

---

[13] By setting G_YRFR(r,s)=0 one can exclude any individual timeslices from specific regions, even if only a global timeslice tree is defined for all regions (as it is the case when using VEDA-FE). In this way each region can employ a different subset of the global tree.

2.2.3 Multi-regional models

If a TIMES model consists of several internal regions, it is called a multi-regional model. Each of the internal regions contains a unique RES to represent the particularities of the region. As already mentioned, the regions can be connected by inter-regional exchange processes to enable trade of commodities between the regions. Two types of trade activities can be depicted in TIMES: bi-lateral trade between two regions and multilateral trade between several supply and demand regions.

Bi-lateral trade takes place between specific pairs of regions. A pair of regions together with an exchange process and the direction of the commodity flow are first identified, where the model ensures that trade through the exchange process is balanced between these two regions (whatever amount is exported from region A to region B must be imported by region B from region A, possibly adjusted for transportation losses). The basic structure is shown in Figure 7. Bi-lateral trading may be fully described in TIMES by defining an inter-regional exchange process and by specifying the two pair-wise connections by indicating the regions and commodities be traded via the set **top_ire(r,c,reg,com,p)**. If trade should occur only in one direction then only that direction is provided in the set **top_ire** (export from region r into region reg). The process capacity and the process related costs (e.g. activity costs, investment costs) of the exchange process can be described individually for both regions by specifying the corresponding parameters in each regions. If for example the investment costs for an electricity line between two regions A and B are 1000 monetary units (MU) per MW and 60 % of these investment costs should be allocated to region A and the remaining 40 % to region B, the investment costs for the exchange process have to be set to 600 MU/MW in region A and to 400 MU/MW in region B.

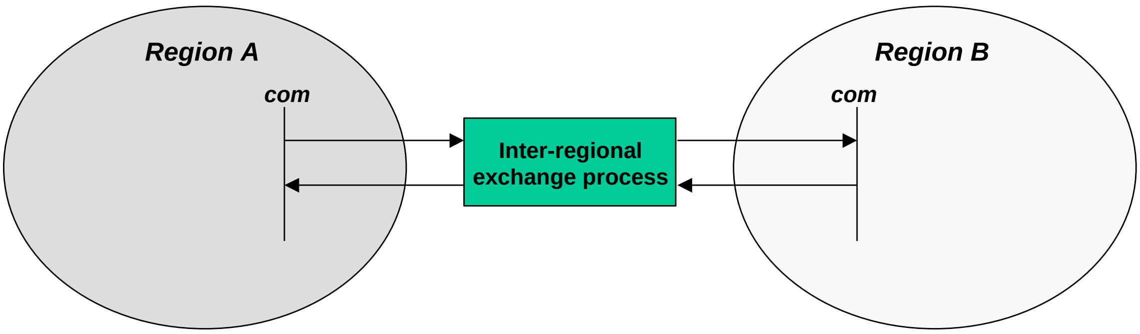


Figure 7: Bilateral trade in TIMES

Bi-lateral trade is the most detailed way to specify trade between regions. However, there are cases when it is not important to fully specify the pair of trading regions. In such cases, the so-called *multi-lateral trade* option decreases the size of the model while preserving enough flexibility. Multi-lateral trade is based on the idea that a common marketplace exists for a traded commodity with several supplying and several consuming regions for the commodity, e.g. for crude oil or GHG emission permits. To facilitate the modelling of this kind of trade scheme the concept of marketplace has been introduced in TIMES. To model a marketplace first the user has to identify one internal region that participates both in the production and consumption of the traded commodity. Then only one exchange process is used to link the supply and demand regions with the marketplace region using the set **top_ire**.[14]

The following example illustrates the modelling of a marketplace in TIMES. Assume that we want to set up a market-based trading where the commodity CRUD can be exported by regions A, B, C, and D, and that it can be imported by regions C, D, E and F (Figure 8).

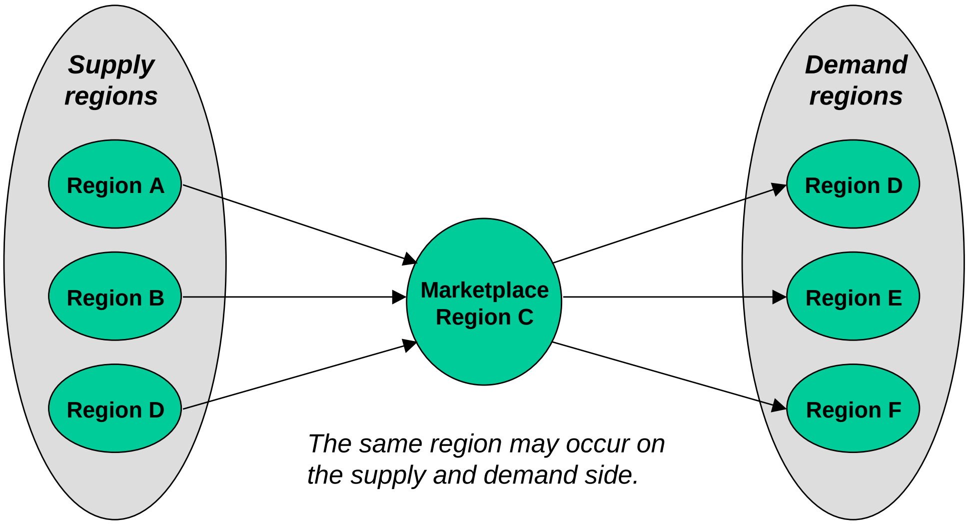


Figure 8: Example of multi-lateral trade in TIMES

First, the exchange process and marketplace should be defined. For example, we could choose the region C as the marketplace region. The exchange process has the name XP. The trade possibilities can then be defined simply by the following six **top_ire** entries:

```
SET PRC / XP /;
SET TOP_IRE /
  A .CRUD .C .CRUD .XP
  B .CRUD .C .CRUD .XP
  D .CRUD .C .CRUD .XP
  C .CRUD .D .CRUD .XP
  C .CRUD .E .CRUD .XP
  C .CRUD .F .CRUD .XP
/;
```

To complete the RES definition of the exchange process, only the set **prc_actunt(r,p,c,u)** is needed to define the units for the exchange process XP in all regions:

```
SET PRC_ACTUNT /
A .XP .CRUD .PJ
B .XP .CRUD .PJ
C .XP .CRUD .PJ
D .XP .CRUD .PJ
E .XP .CRUD .PJ
F .XP .CRUD .PJ
/;
```

[14] Note however that some flexibility is lost when using multilateral trade. For instance, it is not possible to express transportation costs in a fully accurate manner, if such cost depends upon the precise pair of trading regions in a specific way.

These definitions are sufficient for setting up of the market-based trade. Additionally, the user can of course specify various other data for the exchange processes, for example investment and distribution costs, and efficiencies.

2.2.4 Overview of all user input sets

All the input sets which are under user control in TIMES are listed in Table 4. For a few sets default settings exist that are applied if no user input information is given. Set names starting with the prefix 'com_' are associated with commodities, the prefix 'prc_' denotes process information and the prefix 'uc_' is reserved for sets related to user constraints. Column 3 of Table 4 is a description of each set. In some cases (especially for complex sets), two (equivalent) descriptions may be given, the first in general terms, followed by a more precise description within square brackets, given in terms of n-tuples of indices.

Remark

Sets are used in basically two ways:

- as the domain over which summations must be effected in some mathematical expression, or
- as the domain over which a particular expression or constraint must be enumerated (replicated).

In the case of n-dimensional sets, some indexes may be used for **enumeration and others for summation**. In each such situation, the distinction between the two uses of the indexes is made clear by the way each index is used in the expression.

An example will illustrate this important point: consider the 4-dimensional set **top,** having indexes r,p,c,io (see table 4 for its precise description). If some quantity **A(r,p,c,io)** must be enumerated for all values of the third index (c=commodity) and of the last index (io=orientation), but summed over all processes (p) and regions (r), this will be mathematically denoted:

$$EXPRESSION1_{c,io} = \sum_{r,p,c,io \in top} A(r,p,c,io)$$

It is thus understood from the indexes listed in the name of the expression (c,io), that these two indexes are being enumerated, and thus, by deduction, only r and p are being summed upon. Thus the expression calculates the total of A for each commodity c, in each direction io ('IN' and 'OUT'), summed over all processes and regions.

Another example illustrates the case of nested summations, where index r is enumerated in the inner summation, but is summed upon in the outer summation. Again here, the expression is made unambiguous by observing the positions of the different indexes (for instance, the outer summation is done on the r index)

$$EXPRESSION2_{c,io} = \sum_{r,p,c,io \in top} B(r) \sum_{p} A(r,p)$$

Table 4: User input sets in TIMES

Set ID/Indexes[15]	Alias[16]	Description
all_r	all_reg	Set of all regions, internal as well as external; a region is defined as internal by putting it in the internal region set (**r**), regions that are not member of the internal region set are per definition external.
c (cg)	com, com1, com2, com3	User defined list of all commodities in all regions; subset of **cg**.
cg	com_grp, cg1, cg2, cg3, cg4	User defined list of all commodities and commodity groups (see Figure 2) in all regions.
clu (p)		Set of cluster technologies in endogenous technology learning.
com_desc (r,c)		Commodities by region, only to facilitate different descriptions by region. The elements are pairs **{r,c}**, for which the description is specified according to the GAMS syntax.
com_gmap (r,cg,c)		Mapping of commodity **c** to user-defined commodity group **cg,** including itself [set of triplets {**r,cg,c**} such that commodity c in in group cg in region r].[17]
com_lim (r,c,lim)		Definition of commodity balance equation type [set of triplets {**r,c,lim**}such that commodity **c** has a balance of type lim (lim='UP','LO','FX', 'N') in region r]; Default: for commodities of type **NRG**, **DM** and **ENV** production is greater or equal consumption, while for **MAT** and **FIN** commodities the balance is a strict equality.
com_off (r,c,y1,y2)		Specifying that the commodity **c** in region **r** is not available between the years **y1** and **y2** [set of quadruplets {**r,c,y1,y2**} such that commodity c is unavailable from years y1 to y1 in region r] ; note that **y1** may be 'BOH' for the first year of the first period and **y2** may be 'EOH' for the last year of the last period.
com_peak (r,cg)		Set of pairs {**r,cg**} such that a peaking constraint is to be generated for commodity cg in region r; note that the peaking equation can be generated for a single commodity (**cg** also contains single commodities **c**) or for a group of commodities, e.g. electricity commodities differentiated by voltage level.
com_pkts (r,cg,s)		Set of triplets {**r,cg,s**} such that a peaking constraint for a single commodity or a group of commodities **cg** (e.g. if the model differentiates between three electricity commodities: electricity on high, middle and low voltage) is to be generated for the timeslice **s**; Default: all timeslices of **com_ts**; note that the peaking constraint will be binding only for the timeslice with the highest load.
com_tmap (r,com_type,c)		Mapping of commodities to the main commodity types (see **com_type** in Table 1); [set of triplets {**r,com_type,c**} such that commodity c has type com_type];

[15] The first row contains the set name. If the set is a one-dimensional subset of another set, the second row contains the parent set in brackets. If the set is a multi-dimensional set, the second row contains the index domain in brackets.

[16] For programming reasons, alternative names (aliases) may exist for some indexes. This information is only relevant for those users who are interested in gaining an understanding of the underlying GAMS code.

[17] For multidimensional sets such as this one, two definitions are sometimes given, one as an indicator function or mapping, the other (in square brackets) as a set of n-tuples.

Set ID/Indexes[15]	Alias[16]	Description
com_ts (r,c,s)		Set of triplets {**r,c,s**} such that commodity **c** is available in timeslice **s** in region **r**; commodity balances will be generated for the given timeslices; Default: all timeslices of timeslice level specified by **com_tsl**.
com_tsl (r,c,tslvl)		Set of triplets {**r,c,tslvl**} such that commodity **c** is modelled on the timeslice level **tslvl** in region r; Default: 'ANNUAL timeslice level.
com_unit (r,c,units_com)		Set of triplets {**r,c,units_com**} such that commodity **c** is expressed in unit **units_com** in region **r**.
cur		User defined list of currency units.
datayear (year)		Years for which model input data are to be taken; No default.
dem_smap (r,dem_sect,c)		Mapping of demands to main demand sectors (see **dem_sect** in Table 1); [set of triplets {**r,dem_sect,c**} such that commodity c belongs to sector dem_sect];
env_map (r,env_grp,c)		Mapping of environmental commodities to main types (see **env_grp** in Table 1); [set of triplets {**r,env_grp,c**} such that commodity c is of type env_grp].
nrg_tmap (r,nrg_type,c)		Mapping of energy commodities to main types (see **nrg_type** in Table 1); [set of triplets {**r,nrg_type,c**} such that commodity c is of type nrg_type].
p	prc	User defined list of all processes in all regions
pastyear (year)	pyr	Years for which past investments are specified; pastyears have to lie before the beginning of the first period; No default.
prc_actunt (r,p,cg,units_act)		Definition of activity [Set of quadruples such that the commodity group cg is used to define the activity of the process **p**, with units **units_act**, in region **r**].
prc_aoff (r,p,y1,y2)		Set of quadruples {**r,p,y1,y2**} such that process **p** cannot operate (activity is zero) between the years **y1** and **y2** in region r; note that y1 may be 'BOH' for first year of first period and **y2** may be 'EOH' for last year of last period.
prc_capunt (r,p,cg,units_cap)		Definition of capacity unit of process p [set of quadruples {**r,p,cg,units_cap**} such that process **p** uses commodity group **cg** and units **units_cap** to define its capacity in region **r**].
prc_desc (r,p)		Processes by region, only to facilitate different descriptions by region. The elements are pairs **{r,p}**, for which the process description is specified according to the GAMS syntax.
prc_dscncap (r,p)		Set of processes **p** to be modelled using the lumpy investment formulation in region **r**; Default: empty set. If **p** is not in this set, then any lumpy investment parameters provided for **p** are ignored.
prc_foff (r,p,c,s,y1,y2)		Set of sextuples specifying that the flow of commodity **c** at process **p** and timeslice **s** is not available between the years **y1** and **y2** in region r; note that **y1** may be 'BOH' for first year of first period and **y2** may be 'EOH' for last year of last period.
prc_grp		List of process groups, used mainly for reporting purposes; Predefined list of groups (defined in MAPLIST.DEF) is shown in section 2.2.1.
prc_map (r,prc_grp,p)		Grouping of processes into process groups (**prc_grp)** [set of triplets {**r,prc_grp,p**} such that process p belongs to group **prc_grp** in region **r**]. Note: used strictly for reporting purposes.
prc_noff (r,p,y1,y2)		Set of quadruples {**r,p,y1,y2**} such that new capacity of process **p** cannot be installed between the years **y1** and **y2** in region **r**; note that **y1** may be 'BOH' for first year of first period and **y2** may be 'EOH' for last year of last period.
prc_nstts (r,p,s)		Set of triplets {**r,p,s**} such that process **p** is a night storage device with charging timeslices **s** in region r; note that for night storage devices the commodity entering and the commodity leaving the storage may be different, as defined via the set **top**.

Set ID/Indexes[15]	Alias[16]	Description
prc_pkaf (all_r,p)		Set of pairs {**all_r,p**} such that the availability factor (**ncap_af)** is to be used as value for the fraction of capacity of process **p** that can contribute to the peaking constraints **(ncap_pkcnt**), in region **r**.
prc_pkno (all_r,p)		Set of pairs {**all_r,p**}such that process **p** cannot be used in the peaking constraints in region **r**.
prc_rcap (r,p)		Set of pairs {**r,p**}such that early retirements are activated for process **p** in region **r**.
prc_ts (all_r,p,s)	prc_ts2	Set of triplets {**all_r,p,s**} such that process **p** can operate at timeslice **s** in region r; Default: all timeslices on the timeslice level specified by **prc_tsl**.
prc_tsl (r,p,tslvl)		Set of triplets {**r,p,tslvl**} such that process **p** can operate at timeslice level **tslvl** in region **r**; Default: 'ANNUAL' timeslice level.
prc_vint (r,p)		Set of processes **p** that are vintaged technologies in region **r**, i.e. technical characteristics are tied to when the capacity was installed, not the current period; Default: process is not vintaged; note that vintaging increases the model size.
r (all_reg)	reg	Set of internal regions; Subset of **all_r**.
s	all_ts, ts, s2, sl	Set of all timeslices (define the sub-annual divisions of a period). Timeslices effectively defined for specific processes and technologies are subsets of this set.
t (year)	milestonyr, tt	Set of representative years (middle years) for the model periods within the modelling horizon.
teg (p)		Set of technologies selected for endogenous technology learning; Subset of set **p;** if **p** not in **teg**, then any ETL investment parameters provided are ignored**.**
top (r,p,c,io)		RES topology definition indicating that commodity **c** enters (**io**=’IN’) or leaves (**io**='OUT') the process **p** [set of quadruples {**r,p,c,io**} such that process **p** has a flow of commodity **c** with orientation **io** in region **r**].
top_ire (all_reg,com, all_r,c,p)		RES topology definition for trade between regions [Set of quintuples indicating that commodity **com** from region **all_reg** is traded (exported) via exchange process **p** (where it is imported) into region **all_r** as commodity **c]**; note: the name of the traded commodity may be different in the two regions. By using **all_reg**=**all_r**, one can also define bi-directional processes within a region, e.g. for modeling transmission lines.
ts_group (all_r,tslvl,s)		Set of triplets {**all_r,tslvl,s**} such that timeslice **s** belongs to the timeslice level **tslvl** in region **r**; needed for the definition of the timeslice tree; only default is that the 'ANNUAL' timeslice belongs to the 'ANNUAL' timeslice level.
ts_map (all_r,s,ts)		Set of triplets {**all_r,s,ts**} such that s is an intermediate node **s** of the timeslice tree (neither 'ANNUAL' nor the lowest level), and **ts** is a node directly under **s** in region **r**; the set is further extended by allowing **ts** = **s** (see figure 1).
uc_attr (r,uc_n,side, uc_grptype, uc_name)		Set of quintuples such that the UC modifier specified by the **uc_name** (e.g., cost, conversion factor, etc.) will be applied to the coefficient for the variable identified by **uc_grptype** in the user constraint **uc_n**, for the side **side** ('LHS' or 'RHS') in region **r**; if **uc_name**='GROWTH' the user constraint represents a growth constraint.
uc_n	ucn	List of user specified unique indicators of the user constraints.
uc_dynbnd (uc_n,bd)		List of user constraint names **uc_n** that will be handled as simplified process-wise dynamic bound constraints of type **bd**. Can be used together with UC_ACT, UC_CAP, and UC_NCAP for defining the growth/decay coefficients and RHS constants for the dynamic bounds. See EQ_UCRTP for information on usage.
uc_r_each (all_r,uc_n)		Set of pairs {**all_r,uc_n**} such that the user constraint **uc_n** is to be generated for each specified region **all_r**.

Set ID/Indexes[15]	Alias[16]	Description
uc_r_sum (all_r,uc_n)		Set of pairs {**all_r,uc_n**} indicating that the user constraint **uc_n** is summing over all specified regions **all_r** (that is these constraints do not have a region index). Note that depending on the specified regions in **ur_r_sum**, the summation may be done only over a subset of all model regions. For example if the model contains the regions FRA, GER, ESP and one wants to create a user constraint called GHG summing over the regions FRA and GER but not ESP, the set **uc_r_sum** contains has the two entries {'FRA', 'GHG'} and {'GER', 'GHG'}.
uc_t_each (r,uc_n,t)		Indicator that the user constraint **uc_n** is to be generated for each specified period **t**.
uc_t_succ (r,uc_n,t)		Indicator that the user constraint **uc_n** is to be generated between the two successive periods **t** and **t**+1.
uc_t_sum (r,uc_n,t)		Indicator that the user constraint **uc_n** is to be generated summing over the periods **t**.
uc_ts_each (r,uc_n,s)		Indicator that the user constraint **uc_n** will be generated for each specified timeslice **s**.
uc_ts_sum (r,uc_n,s)		Indicator that the user constraint **uc_n** is to be generated summing over the specified timeslice **s**.
uc_tsl (r,uc_n,side,tslvl)		Indicator of the target timeslice level **tslvl** of a timeslice-dynamic (or pseudo-dynamic) user constraint **uc_n**.
v	modlyear	Union of the sets **pastyear** and **t** corresponding to all the years (periods) of a model run (thus actually an internal set).

2.3 Definition of internal sets

The sets internally derived by the TIMES model generator are given in Table 5. The list of internal sets presented here concentrates on the ones frequently used in the model generator and the ones used in the description of the model equations in Chapter 6. Some internal sets are omitted from Table 5 as they are strictly auxiliary sets of the preprocessor whose main purpose is the reduction of the computation time for preprocessor operations.

Table 5: Internal sets in TIMES

Set ID[18] Indexes**[19]**	**Description**
afs (r,t,p,s,bd)	Indicator that the internal parameter COEF_AF, which is used as coefficient of the capacity (new investment variable VAR_NCAP plus past investments NCAP_PASTI) in the capacity utilization constraint EQ(l)_CAPACT, exists.
bohyear (*)[20]	Set **allyear** plus element '**BOH**' (Beginning Of Horizon).
dm_year (year)	Union of sets **datayear** and **modlyear**
eachyear (year)	Set of all years between scalars **MINYR** (first year needed for cost calculation in objective function) and **MIYR_VL + DUR_MAX** (estimation of last year possible cost terms may occur).
eohyear (*)	Set **allyear** plus element '**EOH**' (Ending OF Horizon)
eohyears (year)	Set of all years between scalars **MINYR** (first year needed for cost calculation in objective function) and **MIYR_VL** (last year of model horizon).
finest (r,s)	Set of finest timeslices **s** used in region **r**.
fs_emis (r,p,cg,c,com)	Indicator that the flow variable (VAR_FLO) associated with emission **com** can be replaced by the flow variable of **c** multiplied by the emission factor **FLO_SUM**, which is used in the transformation equation (EQ_PTRANS) between the commodity group **cg** and the commodity **com**; used in the reduction algorithm (see Part III).
g_rcur (r,cur)	Indicator of main currency **cur** by region **r**. For regions having several discounted currencies, the one having highest present value factors is selected; used for undiscounting the solution marginals.
invspred (year,jot,k,y)	Set of investment years **y** and commissioning years **k** belonging to the investment spread starting with **year** and having **jot** number of steps (used for investment and fixed cost accounting).
invstep (year,jot,y,jot)	Set of investment years **y** belonging to the investment spread starting with **year** and having **jot** number of steps (used for investment and fixed cost accounting).
miyr_1 (t)	First **milestonyr**.
no_act (r,p)	List of processes **p** in region **r** not requiring the activity variable; used in reduction algorithm
no_cap (r,p)	List of processes **p** in region **r** not having any capacity related input parameters; used in reduction algorithm.
no_rvp (r,v,p)	New investment in process **p** in region **r** is not possible in period **v** and previously installed capacity does not exist anymore.

[18] Name of the internal set as used in this documentation and the GAMS code.

[19] Index domain of the internal set is given in brackets (Note: the symbols **y**, **y1**, **y2**, **k**, and **ll** all refer to **year**).

[20] The asterisk denotes in the modeling system GAMS a wildcard, so that domain checking is disabled and any index may be used.

Set ID[18] Indexes[19]	**Description**
obj_1a (r,v,p)	Investment case small investment (NCAP_ILED/D(v) <= G_ILEDNO) and no repetition of investment (NCAP_TLIFE + NCAP_ILED >= D(v)) for process **p** in region **r** and vintage period **v**.
obj_1b (r,v,p)	Investment case small investment (NCAP_ILED/D(v) <= G_ILEDNO) and repetition of investment (NCAP_TLIFE + NCAP_ILED < D(v)) for process **p** in region **r** and vintage period **v**.
obj_2a (r,v,p)	Investment case large investment (NCAP_ILED/D(v) > G_ILEDNO) and no repetition of investment (NCAP_TLIFE + NCAP_ILED >= D(v)) for process **p** in region **r** and vintage period **v**.
obj_2b (r,v,p)	Investment case large investment (NCAP_ILED/D(v) > G_ILEDNO) and repetition of investment (NCAP_TLIFE + NCAP_ILED < D(v)) for process **p** in region **r** and vintage period **v**.
obj_idc (r,v,p,life,k,age)	Summation control for calculating the interest during constriction (IDC) for investment Cases 2.a and 2.b.
obj_sumii (r,v,p,life,y,jot)	Summation control for investment and capacity related taxes and subsidies of the in the annual objective function, with lifetime **life**, spread starting in commissioning year **y**, having **jot** number of steps in the spread, and vintage period **v**.
obj_sumiii (r,v,p,ll,k,y)	Summation control for decommissioning costs with for the running year index **y** of annual objective function, vintage period **v**, startup-year **ll**, and commissioning year **k** (e.g. for spreading decommissioning costs over decommissioning time).
obj_sumiv (r,v,p,life,y,jot)	Summation control for fixed costs in the annual objective function with lifetime **life**, spread starting in commissioning year **y**, having **jot** number of steps in the spread, and vintage period **v**.
obj_sumivs (r,v,p,k,y)	Summation control for decommissioning surveillance costs with running year index **y** of annual objective function, vintage period **v** and commissioning year **k**.
obj_sums (r,v,p)	Indicator that process **p** in region **r** with vintage period **v** has a salvage value for investments with a (technical) lifetime that extends past the model horizon.
obj_sums3 (r,v,p)	Indicator that process **p** in region **r** with vintage period **v** has a salvage value associated with the decommissioning or surveillance costs.
obj_sumsi (r,v,p,k)	Indicator that for commissioning years **k** process **p** in region **r** with vintage period **v** has a salvage value due to investment, decommissioning or surveillance costs arising from the technical lifetime extending past the model horizon.
periodyr (v,y)	Mapping of individual years **y** to the modlyear (milestonyr or pastyear; **v)** period they belong to; if **v** is a **pastyear**, only the pastyear itself belongs to the period; for the last period of the model horizon also the years until the very end of the model accounting horizon (MIYR_VL + DUR_MAX) are elements of **periodyr**.
prc_act (r,p)	Indicator that a process **p** in region **r** needs an activity variable (used in reduction algorithm).
prc_cap (r,p)	Indicator that a process **p** in region **r** needs a capacity variable (used in reduction algorithm).
prc_spg (r,p,cg)	Shadow primary group (SPG) of a process **p**; all commodities on the opposite process side of the primary commodity group (PCG) which have the same commodity type as the PCG, usually internally determined (though it may be specified by the user under special circumstances (e.g., when not all the commodities on the opposite side of the process, which should be in the SPG, are of the same commodity type **com_type**). If no commodity of the same type is found: if PCG is of type 'DEM' and process is a material processing process (PRV or PRW), then the SPG contains all material commodities on the SPG side; otherwise the SPG is selected as the first type among the commodity types on the SPG side, in the flowing order: When PCG type is DEM: (NRG, MAT, ENV) When PCG type is NRG: (MAT, DEM, ENV) When PCG type is MAT: (NRG, DEM, ENV) When PCG type is ENV: (NRG, MAT, DEM)

Set ID[18] Indexes[19]	**Description**
prc_stgips (r,p,c)	Set of triplets {**r,p,c**} such that process **p** is an inter-period storage for the commodity **c** in region **r**; the commodity **c** enters and/or leaves the storage according to the set **top**; the storage can only operate at the ANNUAL level.
prc_stgtss (r,p,c)	Set of triplets {**r,p,c**} such that process **p** is a storage process between timeslices (e.g., seasonal hydro reservoir, day/night pumped storage) for commodity **c** in region **r**; commodity **c** enters and/or leaves the storage according to set **top**; the storage operates at the timeslice level **prc_tsl**.
rc (r,c)	List of all commodities **c** found in region **r**.
rc_agp (r,c,lim)	Indicator of which commodities **c** are aggregated into other commodities by aggregation type **lim**.
rc_cumcom (r,com_var,y1,y2,c)	Indicator of a cumulative constraint of type **com_var** defined for commodity **c** between years **y1** and **y2**
rcj (r,c,j,bd)	Steps **j** used in direction **bd** for the elastic demand formulation of commodity **c**.
rcs_combal (r,t,c,s,bd)	Indicator of which timeslices (**s**) associate with commodity **c** in region **r** for time period **t** the commodity balance equation (EQ(l)_COMBAL) is to be generated, with a constraint type corresponding to **bd**.
rcs_comprd (r,t,c,s,bd)	Indicator of which timeslices (**s**) associate with commodity **c** in region **r** for time period **t** the commodity production equation (EQ(l)_COMBAL) is to be generated, with a constraint type according to **bd,** when a corresponding **rhs_comprd** indicator exists.
rcs_comts (r,c,s)	All timeslices **s** being at or above timeslice level (**com_tsl**) of commodity **c** in region **r**.
rdcur (r,cur)	List of currencies **cur** that are discounted (G_DRATE provided) in each region **r**.
rhs_combal (r,t,c,s)	Indicator that the commodity net variable (VAR_COMNET) is required in commodity balance (EQE_COMBAL), owing to bounds/costs imposed on the net amount.
rhs_comprd (r,t,c,s)	Indicator that the commodity production variable (VAR_COMPRD) is required in commodity balance (EQE_COMPRD), owing to a limit/costs imposed on the production.
rp (r,p)	List of processes (**p**) in each region (**r**).
rp_aire (r,p,ie)	List of exchange processes (**p**) in each region (**r**) with indicators (**ie**) corresponding to the activity being defined by imports/exports or both.
rp_flo (r,p)	List of all processes in region (**r)**, except inter-regional exchange processes (**ire**).
rp_inout (r,p,io)	Indicator as to whether a process (**p**) in a region (**r**) is input or output (io = 'IN'/'OUT') normalized with respect to its activity.
rp_ire (all_r,p)	List of inter-regional exchange processes (**p**) found in each region (**all_r**).
rp_pg (r,p,cg)	The primary commodity group (**cg)** of each process (**p)** in a region (**r**).
rp_pgtype (r,p,com_type)	The commodity type (**com_type**) of primary commodity group of a process (**p**) in a region (**r**).
rp_sgs (r,p)	List of those standard processes (**p)** in each region (**r)**, which have been defined to have a night storage (NST) capability.
rp_std (r,p)	List of standard processes (**p**) in each region (**r**).
rp_stg (r,p)	List of storage processes (**p**) in each region (**r**).
rp_sts (r,p)	List of generalized timeslice storage processes (**p**) in each region (**r**).
rp_upl (r,p,lim)	List of those processes (**p**) in each region (**r**) that have dispatching attributes ACT_MINLD/ACT_UPS defined, with qualifier **lim**.
rp_ups (r,p,tslvl,lim)	Timeslices (**s**) of a process (**p**) in a region (**r**) during which start-ups are permitted (used for processes in the set **rp_upl(r,p,'FX'**))
rpc (r,p,c)	List of commodities (**c**) associated with a process **p** in region **r** (by **top** or **top_ire**).

Set ID[18] Indexes[19]	**Description**
rpc_act (r,p,c)	Indicator that the primary commodity group of a process (**p**), except exchange processes (see **rpc_aire**) consists of only one commodity (**c**), enabling the corresponding flow variable to be replaced by the activity variable (used in reduction algorithm).
rpc_aire (r,p,c)	Indicator that the primary commodity group of an exchange process (**p**) consists of only one commodity (**c**), enabling the corresponding flow variable to be replaced by the activity variable (used in reduction algorithm).
rpc_capflo (r,v,p,c)	Indicator that a commodity flow **c** in region **r** is associated with the capacity of a process (**p**, due to NCAP_ICOM, NCAP_OCOM, or NCAP_COM being provided).
rpc_cumflo (r,p,c,y1,y2)	Indicator of a cumulative constraint defined for commodity flow **c** of process **p** between years **y1** and **y2**
rpc_noflo (r,p,c)	A subset of **rpc_capflo** indicating those processes (**p**) in a region (**r**) where a commodity (**c**) is only consumed or produced through capacity based flows, and thus has no flow variable for the commodity.
rpc_emis (r,p,cg)	Indicator that the flow variable of an emission commodity (**cg**) associated with process (**p**) in a region (**r**) can be replaced by the fuel flow causing the emission multiplied by the emission factor (used in reduction algorithm).
rpc_eqire (r,p,c)	Indicator of the commodities (**c**) associated with inter-regional exchange processes (**p**) in region (**r**) for which an inter-region exchange equation (EQ_IRE) is to be generated; the set does not contain the marketplace region (**rpc_market**).
rpcc_ffunc (r,p,c)	Flow variable of a commodity (**c**) associated with a process (**p**) that can be replaced by another flow variable of the process, due to a direct FLO_FUNC or FLO_SUM relationship.
rpc_ire (all_r,p,c,ie)	Commodities (**c**) imported or exported (**ie='IMP'/'EXP'**) via process **p** in a region (**all_r**).
rpc_market (all_r,p,c,ie)	List of market regions (subset of **all_r**) that trade a commodity (**c**) through a process (**p**) either by only multidirectional export links (**ie**='EXP') or by both import and export links (**ie**='IMP'). The market structure is user-defined through the set **top_ire**.
rpc_pg (r,p,cg,c)	Mapping of the commodities (**c**) in a region (**r**) that belong to the primary commodity group (**cg**) associated with process **p**.
rpc_spg (r,p,c)	The list of commodities (**c**) in a region (**r**) belonging to the shadow primary group of process (**p**).
rpc_stg (r,p,c)	List of stored (charged/discharged) commodities (**c**) of storage processes (**p**) in region (**r**).
rpc_stgn (r,p,c,io)	List of those stored (charged/discharged) commodities (**c**) of storage processes (**p**) in region (**r**), which are connected to the commodity balance on one side (**io**) only.
rpcg_ptran (r,p,c1,c2,cg1,cg2)	Indicator of the transformation equations (EQ_PTRANS) that can be eliminated by the reduction algorithm.
rpcs_var (r,p,c,s)	The list of valid timeslices for the flow variable (VAR_FLO) of commodity **c** associated with process **p** in region **r**; flow variables of commodities which are part of the primary commodity group have the timeslice resolution of the process (**prc_tsl**), while all other flow variables are created according to the **rps_s1** timeslices.
rps_prcts (r,p,s)	All (permitted) timeslices (**s**) at or above the process (**p**) timeslice level (**prc_tsl**) in a region (**r**).
rps_s1 (r,p,s)	All (permitted) timeslices (**s**) belonging to the finest timeslice level of the process (**p, prc_tsl**) and the commodity timeslice level (**com_tsl**) of the shadow primary commodity group.
rps_s2 (r,p,s)	For an ANNUAL level NST process, contains all permitted timeslices (**s**) at the level above the finest commodity timeslice levels (**com_tsl**) of the shadow primary group (spg). For all other processes, rps_s2 = rps_s1.
rps_stg (r,p,s)	Process level timeslices (**s**) of timeslice storage process (**p**) in a region (**r**).
rreg (all_reg,all_r)	Indicator that trade exists from region **all_reg** to region **all_r**.

Set ID[18] Indexes[19]	**Description**
rs_below (all_r,ts,s)	All timeslices (**s**) strictly below the higher timeslice (**ts**) in the timeslice tree.
rs_below1 (all_r,ts,s)	All timeslices (**s**) immediately (one level) below the higher timeslice (**ts**) in the timeslice tree.
rs_tree (all_r,ts,s)	For a timeslice (**ts**) all timeslices (**s**) that are on the same paths within the timeslice tree, e.g. if **ts**=SP_WD in Fig. 6, valid timeslices **s** are: ANNUAL, SP, SP_WD, SP_WD_D, SP_WD_N
rtc_cumnet (r,t,c)	Indicator that the commodity net variable (VAR_COMNET) for commodity **c** in region **r** for period **t** has a cumulative bound applied.
rtc_cumprd (r,t,c)	Indicator that the commodity production variable (VAR_COMPRD) for commodity **c** in region **r** for period **t** has a cumulative bound applied.
rtcs_sing (r,t,c,s,io)	Indicator that a commodity **c** is not available in a specific period **t** and timeslice **s**, since the all the processes producing (**io**='OUT') or consuming it (**io**='IN') are turned-off. In the case of **io**='OUT', the commodity is not available, meaning that processes having only this commodity as input cannot operate. Similar reasoning applies to the case **io**='IN'.
rtcs_varc (r,t,c,s)	For commodity (**c**) in region (**r**) indicator for the timeslices (**s**) and the periods (**t**) the commodity is available.
rtp = rvp (r,v,p)	Indication of the periods and pastyears for which process (**p**) in region (**r**) is available; all other RTP_* control sets are based on this set.
rtp_cptyr (r,v,t,p)	For each vintage period (**v**) an indication of the periods (**t**) for which newly installed capacity of process (**p**) in a region (**r**) is available, taking into account construction lead-time (**NCAP_ILED**) and technical lifetime (**NCAP_TLIFE**).
rtp_off (r,t,p)	Indication of the periods (**t**) in which no new investment is permitted for a process (**p**) in a region (**r**).
rtp_vara (r,t,p)	Indication of the periods (**t**) for which a process (**p**) in a region (**r**) is available.
rtp_varp (r,t,p)	Indicator that the capacity variable (**VAR_CAP**) will be generated for process (**p**) in a region (**r**) in period (**t**).
rtp_vintyr (r,v,t,p)	An indication of for which periods (**t**) a process (**p**) in a region (**r**) is available since it was first installed (**v**); for vintaged processes (**prc_vint**) identical to **rtp_cptyr**, for non-vintaged processes the **v** index in the **rtp_cptyr** entries is ignored by setting it to **t** (**v** = **t**).
rtpc (r,t,p,c)	For a process (**p**) in a region (**r**) the combination of the periods it is available (**rtp**) and commodities associated with it (**rpc**).
rtps_off (r,t,p,s)	An indication for process (**p**) of the timeslices (**s**) for which the process is turned-off (used in reduction algorithm).
rtpcs_varf (r,t,p,c,s)	The list of valid timeslices (**s**) and periods (**t**) for the flow variable (VAR_FLO) of process (**p**) and commodity (**c**); taking into account the availability of the activity, capacity and flow (**rtp_vara**, **rpcs_var** and **prc_foff**). The timeslice level of a flow variable equals the process timeslice level (**prc_tsl**) when the flow is part of the primary commodity group of the process. Otherwise the timeslice level of a flow variable is set to the finest level of the commodities in the shadow group (SPG) or the process level, whichever is finer.
uc_dyndir (r,uc_n,side)	If **side** = 'RHS', indicator for growth constraints to be generated between the periods **t−1** and **t**; if **side** = 'LHS', the set is ignored.
uc_gmap_c (r,uc_n,uc_grptype,c)	Indicator that a commodity variable (VAR_COMCON or VAR_COMPRD) for commodity (**c**) in a region (**r**) appears in a user constraint (**uc_n**).
uc_gmap_p (r,uc_n,uc_grptype,p)	Indicator that a variable (VAR_ACT, VAR_NCAP or VAR_CAP) associated with a process (**p**) in a region (**r**) appears in a user constraint (**uc_n**).
uc_gmap_u (r,uc_n,ucn)	Indicator that a variable (VAR_UCRT) associated with a user constraint (**ucn**) in a region (**r**) appears in another user constraint (**uc_n**).
uc_map_flo (uc_n,r,p,c)	Indicator that the flow variable (VAR_FLO) for region **r**, process **p** and commodity **c** is involved in user constraint **uc_n**.
uc_map_ire (uc_n,r,p,c)	Indicator that an import/export (according to **top_ire**) trade variable (VAR_IRE) for region **r**, process **p**, and commodity **c** is involved in a user constraint (**uc_n**).
v	Union of the input sets **pastyear** and **t**, corresponding to all the periods of a model run (=modlyear).

3 Parameters

While sets describe structural information of the energy system or qualitative characteristics of its entities (e.g. processes or commodities), parameters contain numerical information. Examples of parameters are the import price of an energy carrier or the investment cost of a technology. Most parameters are time-series where a value is provided (or interpolated) for each year (datayear). The TIMES model generator distinguishes between user input parameters and internal parameters. The former are provided by the modeller (usually by way of a data handling system or "shell" such a VEDA-FE or ANSWER-TIMES), while the latter are internally derived from the user input parameters, in combination with information given by sets, in order to calculate for example the cost coefficients in the objective function. This Chapter first covers the user input parameters in Section 3.1 and then describes the most important internal parameters as far as they are relevant for the basic understanding of the equations (Section 3.2). Section 3.3 presents the parameters used for reporting the results of a model run.

3.1 User input parameters

This section provides an overview of the user input parameters that are available in TIMES to describe the energy system. Before presenting the various parameters in detail in Section 3.1.3 two preprocessing algorithms applied to the user input data are presented, namely the inter-/extrapolation and the inheritance/aggregation routines. User input parameters that are time-dependent can be provided by the user for those years for which statistical information or future projections are available, and the inter-/extrapolation routine described in Section 3.1.1 used to adjust the input data to the years required for the model run. Timeslice dependent parameters do not have to be provided on the timelice level of a process, commodity or commodity flow. Instead the so-called inheritance/aggregation routine described in Section 3.1.2 assigns the input data from the user provided timeslice level to the appropriate timeslice level as necessary.

3.1.1 Inter- and extrapolation of user input parameters

Time-dependent user input parameters are specified for specific years, the so-called *datayears* (**datayear**). These datayears do not have to coincide with the modelyears (**v** or **modelyear)** needed for the current run. Reasons for differences between these two sets are for example that the period definition for the model has been altered after having provided the initial set of input data leading to different milestoneyears (**t** or **milestoneyr)** or that data are only available for certain years that do not match the modelyears. In order to avoid burdening the user with the cumbersome adjustment of the input data to the modelyears, an inter-/extrapolation (I/E) routine is embedded in the TIMES model generator. The inter-/extrapolation routine distinguishes between a default inter-/extrapolation that is automatically applied to the input data and an enhanced user-controlled inter-/extrapolation that allows the user to specify an inter-/extrapolation rule for each time-series explicitly. Independent of the default or user-controlled I/E options, TIMES inter-/extrapolates (using the standard algorithm) all cost parameters in the objective function to the individual years of the model as part of calculating the annual cost details (see section 3.1.1.3 below).

The possibility of controlling interpolation on a time-series basis improves the independence between the years found in the primary database and the data actually used in the individual runs of a TIMES model. In this way the model is made more flexible with respect to running scenarios with arbitrary model years and period lengths, while using basically the very same input database.

3.1.1.1 Inter/extrapolation options

The TIMES interpolation/extrapolation facility provides both a default I/E method for all time-series parameters, and options for the user to control the interpolation and extrapolation of each individual time series (Table 6). The option 0 does not change the default behavior. The specific options that correspond to the default methods are 3 (the standard default) and 10 (alternative default method for bounds and RHS parameters).

Non-default interpolation/extrapolation can be requested for any parameter by providing an additional instance of the parameter with an indicator in the YEAR index and a value corresponding to one of the integer-valued Option Codes (see Table 6 and example below). This control specification activates the interpolation/extrapolation rule for the time series, and is distinguished from actual time-series data by providing a special control label ('**0**') in the YEAR index. The particular interpolation rule to apply is a function of the Option Code assigned to the control record for the parameter. Note that for log-linear interpolation the Option Code indicates the year from which the interpolation is switched from standard to log-linear mode. TIMES user shell(s) will provide mechanisms for imbedding the control label and setting the Option Code through easily understandable selections from a user-friendly drop-down list, making the specification simple and transparent to the user.

The enhanced interpolation/extrapolation facility provides the user with the following options to control the interpolation and extrapolation of each individual time series:

- Interpolation and extrapolation of data in the default way as predefined in TIMES. This option does not require any explicit action from the user.
- No interpolation or extrapolation of data (only valid for non-cost parameters).
- Interpolation between data points but no extrapolation (useful for many bounds). See option codes 1 and 11 in Table 2 below.
- Interpolation between data points entered, and filling-in all points outside the interpolation window with the EPS (zero) value. This can useful for e.g. the RHS of equality-type user constraints, or bounds on future investment in a particular instance of a technology. See option codes 2 and 12 in Table 2 below.

Table 6: Option codes for the control of time series data interpolation

Option code	Action	Applies to
0 (or none)	Interpolation and extrapolation of data in the default way as predefined in TIMES (see below)	All
< 0	No interpolation or extrapolation of data (only valid for non-cost parameters).	All
1	Interpolation between data points but no extrapolation.	All
2	Interpolation between data points entered, and filling-in all points outside the interpolation window with the EPS value.	All
3	Forced interpolation and both forward and backward extrapolation throughout the time horizon.	All
4	Interpolation and backward extrapolation	All
5	Interpolation and forward extrapolation	All
10	Migrated interpolation/extrapolation within periods	Bounds, RHS
11	Interpolation migrated at end-points, no extrapolation	Bounds, RHS
12	Interpolation migrated at ends, extrapolation with EPS	Bounds, RHS
14	Interpolation migrated at end, backward extrapolation	Bounds, RHS
15	Interpolation migrated at start, forward extrapolation	Bounds, RHS
YEAR (≥ 1000)	Log-linear interpolation beyond the specified YEAR, and both forward and backward extrapolation outside the interpolation window.	All

- Forced interpolation and extrapolation throughout the time horizon. Can be useful for parameters that are by default not interpolated. See option codes 3, 4, and 5 as well as 14 and 15 in Table 2 below.
- Log-linear interpolation beyond a specified data year, and both forward and backward extrapolation outside the interpolation window. Log-linear interpolation is guided by relative coefficients of annual change instead of absolute data values.

Migration means that data points are interpolated and extrapolated within each period but not across periods. This method thus migrates any data point specified for other than milestoneyr year to the corresponding milestoneyr year within the period, so that it will be effective in that period.

Log-linear interpolation means that the values in the data series are interpreted as coefficients of annual change beyond a given YEAR. The YEAR can be any year, including modelyears. The user only has to take care that the data values in the data series correspond to the interpretation given to them when using the log-linear option. For simplicity, however, the first data point is always interpreted as an absolute value, because log-linear interpolation requires at least one absolute data point to start with.

3.1.1.2 Default inter/extrapolation

The standard default method of inter-/extrapolation corresponds to the option 3, which interpolates linearly between data points, while it extrapolates the first/last data point constantly backward/forward. This method, full interpolation and extrapolation, is by default applied to most TIMES time series parameters. However, the parameters listed in Table 7 are by default **NOT** inter/extrapolated in this way, but have a different default method.

3.1.1.3 Interpolation of cost parameters

As a general rule, all cost parameters in TIMES are densely interpolated and extrapolated. This means that the parameters will have a value for every single year within the range of years they apply, and the changes in costs over years will thus be accurately taken into account in the objective function. The user can use the interpolation options 1–5 for even cost parameters. Whenever an option is specified for a cost parameter, it will be first sparsely interpolated/extrapolated according to the user option over the union of modelyear and datayear, and any remaining empty data points are filled with the EPS value. The EPS values will ensure that despite the subsequent dense interpolation the effect of user option will be preserved to the extent possible. However, one should note that due to dense interpolation, the effects of the user options will inevitably be smoothed.

3.1.1.4 Examples of using I/E options

Example 1:

Assume that we have three normal data points in a FLO_SHAR data series:

```
FLO_SHAR('REG','1995','PRC1','COAL','IN_PRC1','ANNUAL','UP') = 0.25;
FLO_SHAR('REG','2010','PRC1','COAL','IN_PRC1','ANNUAL','UP') = 0.12;
FLO_SHAR('REG','2020','PRC1','COAL','IN_PRC1','ANNUAL','UP') = 0.05;
```

FLO_SHAR is by default NOT interpolated or extrapolated in TIMES. To force interpolation/extrapolation of the FLO_SHAR parameter the following control option for this data series should be added:

```
FLO_SHAR('REG','0','PRC1','COAL','IN_PRC1','ANNUAL','UP') = 3;
```

Table 7: Parameters not being fully inter/extrapolated by default

Parameter	Justification	Default I/E
ACT_BND	Bound may be intended at specific periods only.	10 (migration)
CAP_BND		
NCAP_BND		
NCAP_DISC		
FLO_FR		
FLO_SHAR		
STGIN_BND		
STGOUT_BND		
COM_BNDNET		
COM_BNDPRD		
COM_CUMNET		
COM_CUMPRD		
REG_BNDCST		
RCAP_BND		
IRE_BND		
IRE_XBND		
PRC_MARK	Constraint may be intended at specific periods only	11
PRC_RESID	Residual capacity usually intended to be only interpolated	1*
UC_RHST	User constraint may be intended for specific periods only	10 (migration)
UC_RHSRT		
UC_RHSRTS		
NCAP_AFM	Interpolation meaningless for these parameters (parameter value is a discrete number indicating which MULTI curve should be used).	10 (migration)
NCAP_FOMM		
NCAP_FSUBM		
NCAP_FTAXM		
COM_ELASTX	Interpolation meaningless for these parameters (parameter value is a discrete number indicating which SHAPE curve should be used).	10 (migration)
FLO_FUNCX		
NCAP_AFX		
NCAP_FOMX		
NCAP_FSUBX		
NCAP_FTAXX		
NCAP_PASTI	Parameter describes past investment for a single vintage year.	none
NCAP_PASTY	Parameter describes number of years over which to distribute past investments.	none
CM_MAXC	Bound may be intended at specific years only	none
PEAKDA_BL	Blending parameters at the moment not interpolated	none

* If only a single PRC_RESID value is specified, assumed to decay linearly over NCAP_TLIFE years

Example 2:

Assume that we define the following log-linear I/E option for a FLO_SHAR data series:

```
FLO_SHAR('REG','0','PRC1','COAL','IN_PRC1','ANNUAL','UP') = 2005;
```

This parameter specifies a log-linear control option with the value for the threshold YEAR of log-linear interpolation taken from 2005. The option specifies that all data points up to the year 2005 should be interpreted normally (as absolute data values), but all values beyond that year should be interpreted as coefficients of annual change. By using this interpretation, TIMES will then apply full interpolation and extrapolation to the whole data series. It is the responsibility of the user to ensure that the first data point and all data points up to (and including) the year 2005 represent absolute values of the parameter, and that all subsequent data points represent coefficients of annual change. Using the data of the example above, the first data point beyond 2005 is found for the year 2010, and it has the value of 0.12. The interpretation thus requires that the maximum flow share of COAL in the commodity group IN_PRC1 is actually meant to increase by as much as 12% per annum between the years 1995 and 2010, and by 5% per annum between 2010 and 2020.

3.1.1.5 Applicability

All the enhanced I/E options described above are available for all TIMES time-series parameters, excluding PRC_RESID and COM_BPRICE. PRC_RESID is always interpolated, as if option 1 were used, but is also extrapolated forwards over TLIFE when either I/E option 5 or 15 is specified. COM_BPRICE is not interpolated at all, as it is obtained from the Baseline solution. Moreover, the I/E options are not applicable to the integer-valued parameters related to the SHAPE and MULTI tables, which are listed in Table 8.

Table 8: Parameters which cannot be interpolated.

Parameter	Comment
NCAP_AFM	Parameter value is a discrete numbers indicating which MULTI curve should be used, and not a time series datum.
NCAP_FOMM	
NCAP_FSUBM	
NCAP_FTAXM	
COM_ELASTX	Parameter value is a discrete number indicating which SHAPE curve should be used, and not a time series datum.
FLO_FUNCX	
NCAP_AFX	
NCAP_FOMX	
NCAP_FSUBX	
NCAP_FTAXX	

Nonetheless, a few options are supported also for the extrapolation of the MULTI and SHAPE index parameters, as shown in Table 9. The extrapolation can be done either only inside the data points provided by the user, or both inside and outside those data points. When using the inside data points option, the index specified for any **datayear** is extrapolated to all modelyears **(v)** between that **datayear** and the following **datayear** for which the SHAPE index is specified. The extrapolation options are available for all of the SHAPE and MULTI parameters listed in Table 8.

Table 9: Option codes for the extrapolation of SHAPE/MULTI indexes.

Option code	Action
<= 0 (or none)	No extrapolation (default)
1	Extrapolation between data points only
2	Extrapolation between and outside data points
11	Extrapolation between data points only, migration at ends

Example:

The user has specified the following two SHAPE indexes and a control option for extrapolation:

```
NCAP_AFX('REG', '0', 'PRC1') = 1;
NCAP_AFX('REG', '1995', 'PRC1') = 12;
NCAP_AFX('REG', '2010', 'PRC1') = 13;
```

In this case, all modelyears **(v)** between 1995 and 2010 will get the shape index 12. No extrapolation is done for modelyears **(v)** beyond 2010 or before 1995.

3.1.2 Inheritance and aggregation of timesliced input parameters

As mentioned before, processes and commodities can be modelled in TIMES on different timeslice levels. Some of the input parameters that describe a process or a commodity are timeslice specific, i.e. they have to be provided by the user for specific timeslices, e.g. the availability factor NCAP_AF of a power plant operating on a 'DAYNITE' timeslice level. During the process of developing a model, the timeslice resolution of some processes or even the entire model may be refined. One could imagine for example the situation that a user starts developing a model on an 'ANNUAL' timeslice level and refines the model later by refining the timeslice definition of the processes and commodities. In order to avoid the need for all the timeslice related parameters to be re-entered again for the finer timeslices, TIMES supports inheritance and aggregation of parameter values along the timeslice tree.

Inheritance in this context means that input data being specified on a coarser timeslice level (higher up the tree) are inherited to a finer timeslice level (lower down the tree), whereas aggregation means that timeslice specific data are aggregated from a finer timeslice level (lower down the tree) to a coarser one (further up the tree). The inheritance feature may also be useful in some cases where the value of a parameter should be the same over all timeslices, since in this case it is sufficient to provide the parameter value for the 'ANNUAL' timeslice which is then inherited to the required finer target timeslices.[21]

The TIMES pre-processor supports different inheritance and aggregation rules, which depend on the type of attribute. The main characteristics of the different inheritance and

Table 10: Inheritance and aggregation rules

Inheritance rules	**Description**
Direct inheritance	A value on a coarser timeslice is inherited by target timeslices below (in the timeslice tree), without changing the numeric values.
Weighted inheritance	A value on a coarser timeslice is inherited by target timeslices below (in the timeslice tree) by weighting the input value with the ratio of the duration of the target timeslices to the duration of the coarser timeslice. Example: Parameter COM_FR.
No inheritance	Absolute bound parameters specified on a coarser timeslice level than the target timeslice level are not inherited. Instead a constraint summing over related variables on the finer timeslices is generated, e.g. an annual ACT_BND parameter specified for a process with a 'DAYNITE' process timeslice level (prc_tsl) leads to a constraint (EQ_ACTBND) with the summation over the activity variables on the 'DAYNITE' level as LHS term and with the bound as RHS term.
Aggregation rules	**Description**
Standard aggregation	The values specified on finer timeslices are aggregated to the target timeslice being a parent node in the timeslice tree by summing over the values on the finer timeslices.
Weighted aggregation	The values specified for finer timeslices are aggregated to the target timeslice being a parent node in the timeslice tree by summing over the weighted values on the finer timeslices. The ratios of the duration of the finer timeslices to the duration of the target timeslice serve as weighting factors.

[21] The term *target timeslice level* or *target timeslice* is used in the following as synonym for the timeslice level or timeslices which are required by the model generators depending on the process or commodity timeslice resolution (**prc_tsl** and **com_tsl** respectively).

aggregation rules are summarised in Table 10. The specific rules applied to each individual parameter are listed in the detailed reference Table 13 further below.

The different aggregation rules are illustrated by examples in Figure 9. It should be noted that if input data are specified on two timeslice levels different from the target level, then especially the weighted inheritance/aggregation method may lead to incorrect results. Therefore, at least for the parameters where weighted methods are applied, it is recommended to provide input data only for timeslices on one timeslice level. However, for parameters that are directly inherited, specifying values at multiple levels may sometimes be a convenient way to reduce the amount of values to be specified.[22]

Bound parameters are in most cases not levelized by inheritance, only by aggregation. Exceptions to this rule are the relative type bound parameters NCAP_AF and FLO_SHAR, which are inherited by the target timeslices. One should also notice that, due to levelization, fixed bounds that are either inherited or aggregated to the target timeslice level will always override any upper and lower bounds simultaneously specified.

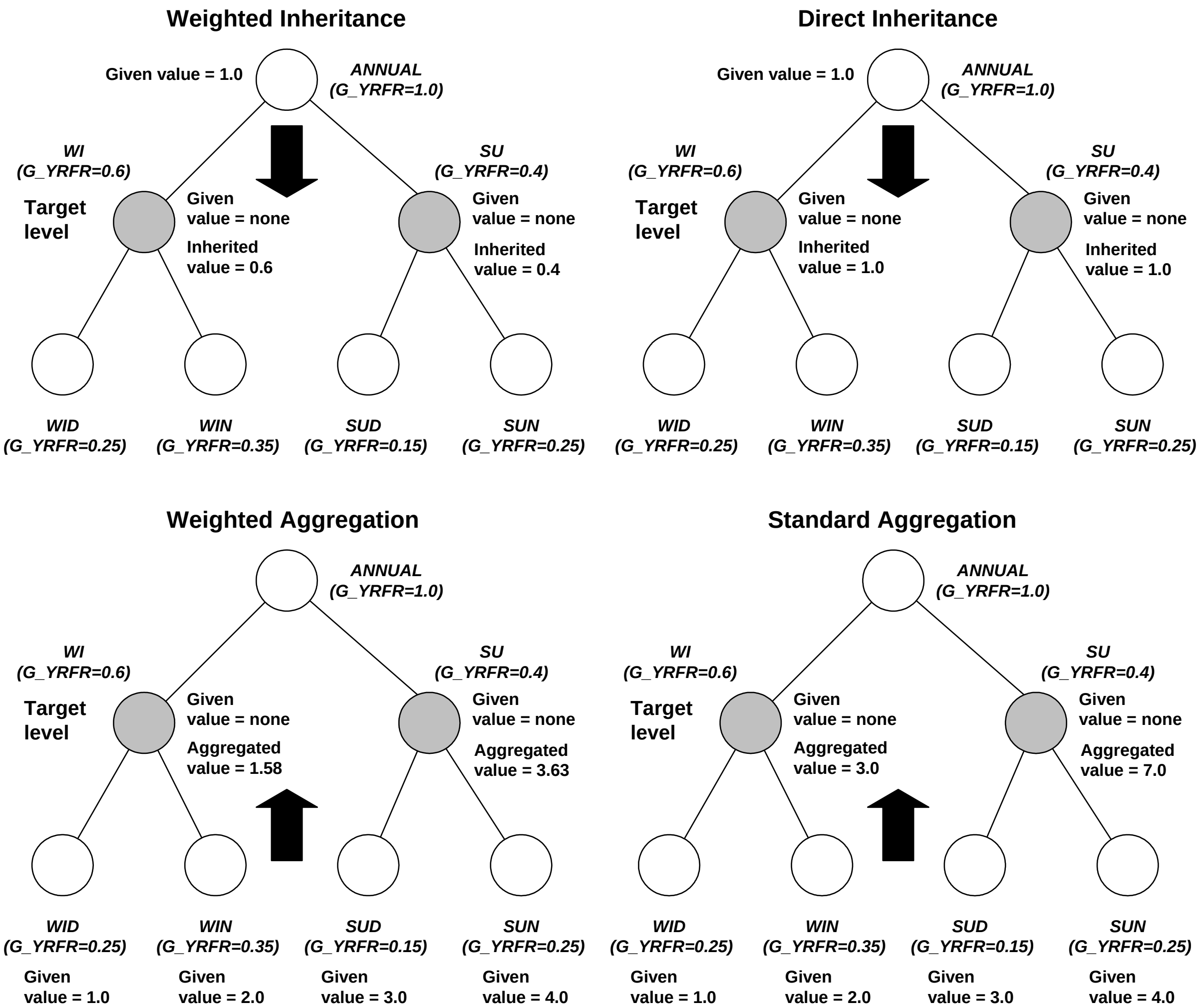


Figure 9: Inheritance and aggregation rules for timeslice specific parameters in TIMES

[22] Note that as an exception, for NCAP_AF direct inheritance and aggregation will be disabled if any values are specified at the process timeslice level. However, this may be circumvented by using NCAP_AFS for defining the values at process timeslices.

3.1.3 Overview of user input parameters

A list of all user input parameters is given in Table 13. In order to facilitate the recognition by the user of to which part of the model a parameter relates the following naming conventions apply to the prefixes of the parameters (Table 11).

Table 11: Naming conventions for user input parameters

Prefix	Related model component
G_	Global characteristic
ACT_	Activity of a process
CAP_	Capacity of a process
COM_	Commodity
FLO_	Process flow
IRE_	Inter-regional exchange
NCAP_	New capacity of a process
PRC_	Process
RCAP_	Retiring capacity of a process
REG_ / R_	Region-specific characteristic
STG_	Storage process
UC_	User constraint

For brevity, the default interpolation/extrapolation method for each parameter is given by using the abbreviations listed in Table 12.

Table 12: Abbreviations for default I/E method in Table 13.

Abbreviation	Description
STD	Standard full inter-/extrapolation (option 3)
MIG	Migration (option 10)
<number>	Option code for any other default method
none	No default inter-/extrapolation
N/A	Inter-/extrapolation not applicable

Table 13: User input parameters in TIMES

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
ACT_BND (r,datayear,p,s,bd)		Units of activity [0,∞); default value: none Default i/e[28]: MIG	Since inter-/extrapolation default is MIG, the bound must be explicitly specified for each period, unless an inter-/extrapolation option is set. If the bound is specified for a timeslice s above the process timeslice resolution (prc_tsl), the bound is applied to the sum of the activity variables according to the timeslice tree. Standard aggregation.	Bound on the overall activity a process.	Activity limit constraint (EQ(l)_ACTBND) when s is above prc_tsl. Direct bound on activity variable (VAR_ACT) when at the prc_tsl level.
ACT_COST (r,datayear,p,cur)	OBJ_ACOST, CST_ACTC, CST_PVP	Monetary unit per unit of activity [open]; default value: none Default i/e: STD		Variable costs associated with the activity of a process.	Applied to the activity variable (VAR_ACT) as a component of the objective function (EQ_OBJVAR). May appear in user constraints (EQ_UC*) if

[23] The first row contains the parameter name, the second row contains in brackets the index domain over which the parameter is defined.
[24] This column gives references to related input parameters (in upper case) or sets (in lower case) being used in the context of this parameter as well as internal parameters/sets or result parameters being derived from the input parameter.
[25] This column lists the unit of the parameter, the possible range of its numeric value [in square brackets] and the inter-/extrapolation rules that apply.
[26] An indication of circumstances for which the parameter is to be provided or omitted, as well as description of inheritance/aggregation rules applied to parameters having the timeslice (**s)** index.
[27] Equations or variables that are directly affected by the parameter.
[28] Abbreviation i/e = inter-/extrapolation

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
					specified in UC_NAME.
ACT_CSTPL (r,datayear,p,cur)	ACT_MINLD ACT_LOSPL	Monetary unit per unit of activity [0,∞); default value: none Default i/e: STD	Used as an alternative or supplement to using ACT_LOSPL(r,y,p,'FX'). When used as an alternative, the fuel increase at the minimum operating level that should be included in the cost penalty must be embedded in the ACT_CSTPL coefficient.	Partial load cost penalty, defined as an additional cost per activity at the minimum operating level, corresponding to the efficiency loss at that load level. Added as an extra term to variable costs in the objective and reporting.	Generates an additional term in EQ_OBJVAR for the increase in operating cost.
ACT_CSTUP (r,datayear,p,tslvl,cur)	ACT_MINLD ACT_UPS	Monetary unit per unit of capacity [0,∞); default value: none Default i/e: STD	The tslvl level refers to the timeslice cycle for which the start-up cost is defined. Only applicable when the min. stable operating level has been defined with ACT_MINLD.	Cost of process start-up per unit of started-up capacity. Added as an extra term to variable costs in the objective and reporting.	Activates generation of EQL_ACTUPS eqs. Generates an additional term in the variable operating costs included in EQ_OBJVAR.
ACT_CUM (r,p,y1,y2,bd)	FLO_CUM	Activity unit [0,∞); default value: none Default i/e: N/A	The years y1 and y2 may be any years of the set allyear; where y1 may also be 'BOH' for first year of first period and y2 may be 'EOH' for last year of last period.	Bound on the cumulative amount of annual process activity between the years y1 and y2, within a region.	Generates an instance of the cumulative constraint (EQ_CUMFLO)
ACT_EFF (r,datayear,p,cg,s)		Activity unit per flow unit [0,∞); default value: none Default i/e: STD	The group cg may be a single commodity, group, or commodity type on the shadow side, or a single commodity in the PCG; cg='ACT' refers to the default shadow group. If no group efficiency is defined,	Activity efficiency for process, i.e. amount of activity per unit of commodity flows in the group cg. For more information on usage, see Section 6.3 for details about	Generates instances of the activity efficiency constraint (EQE_ACTEFF)

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			shadow group is assumed to be the commodity type. Individual commodity efficiencies are multiplied with the shadow group efficiency (default=1). Direct inheritance. Weighted aggregation.	EQE_ACTEFF.	
ACT_FLO (r,datayear,p,cg,s)		Flow unit per activity unit [0,∞); default value: none Default i/e: STD	Inherited/aggregated to the timeslice levels of the process activity. Direct inheritance. Weighted aggregation.	Flow of commodities in cg in proportion to activity, in timeslice s.	Establishes a transformation relationship (EQ_PTRANS) between the flows in the PCG and one or more input (or output) commodities.
ACT_LOSPL (r,datayear,p,bd)	ACT_MINLD ACT_CSTPL	Decimal fraction [0,∞); default values: FX: none LO: default value is ACT_MINLD or 0.1 if that is not defined UP: 0.6 Default i/e: STD	Endogenous partial load modeling can only be used for processes that have their efficiency modelled by the ACT_EFF parameter. For other processes, the ACT_CSTPL parameter can be used for modeling a cost penalty at partial loads.	Partial load efficiency parameters. 1) (bd='FX'): Proportional increase in specific fuel consumption at minimum operating level 2) (bd='LO'): Minimum operating level of partial load operation 3) (bd='UP'): Fraction of feasible load range above the minimum operating level, below which the efficiency losses are assumed to occur	Generates instances of the partial load efficiency constraint EQ_ACTPL.
ACT_MINLD (r,datayear,p)	ACT_UPS ACT_CSTUP	Decimal fraction [0,∞);	Can only be used for standard processes (not	Minimum stable operating level of a	Generates instances of equations

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
	ACT_CSTPL ACT_LOSPL	default value: none Default i/e: STD	IRE or STG). Must be defined if ACT_CSTUP or ACT_TIME is specified.	dispatchable process.	EQ_CAPLOAD and EQE_ACTUPS.
ACT_TIME (r,datayear,p,lim)	ACT_MINLD ACT_CSTUP ACT_UPS	Hours [0,∞); default value: none Default i/e: STD	Requires that start-up costs have been modeled for the process, using both ACT_MINLD and ACT_CSTUP at the DAYNITE/WEEKLY level. The lim type 'FX' is not supported and is ignored.	1) Minimum online (UP) / offline (LO) hours of a process with start-up costs modeled (lim=LO/UP) 2) Maximum number of start-up cycles within the process timeslice cycles (lim=N).	Generates instances of EQL_ACTUPC
ACT_UPS (r,datayear,p,s,bd)	ACT_MINLD ACT_CSTUP ACT_CSTPL ACT_LOSPL	Decimal fraction [0,∞); default value: none Default i/e: STD	Inherited/aggregated to the timeslice levels of the process activity. Direct inheritance. Weighted aggregation. The ramp rates can only be specified with bd=LO/UP.	Maximum ramp-rate (down/up) of process activity as a fraction of nominal on-line capacity per hour.	Generates instances of equation EQ_ACTRAMP.
B (t)	M, D, E, COEF_CPT, rtp_vintyr			Beginning year of period t.	
CAP_BND (r,datayear,p,bd)	PAR_CAPLO, PAR_CAPUP	Capacity unit [0,∞); default value: none Default i/e: MIG	Since inter-/extrapolation is default is MIG, a bound must be specified for each period desired, if no explicit inter-/extrapolation option is given.	Bound on investment in new capacity.	Imposes an indirect limit on the capacity transfer equation (EQ_CPT) by means of a direct bound on the capacity variable (VAR_CAP).
CM_CONST (item)		Constant specific unit [open]; default value: See Appendix	See Appendix on Climate Module for details.	Various climate module constants, e.g. phi and sigma values between	EQ_CLITOT EQ_CLICONC EQ_CLITEMP EQ_CLIBEOH

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		Default i/e: N/A		reservoirs.	
CM_EXOFORC (year)		Forcing unit [open]; default value: none Default i/e: STD	Default values are provided. See Appendix on Climate Module for details.	Radiative forcing from exogenous sources	EQ_CLITOT
CM_GHGMAP (r,c,cm_var)		Units of climate module emissions per units of regional emissions [0, ∞); default value: none	The global emissions in the climate module (cm_var) are 'CO2-GtC' (GtC), 'CH4-Mt' (Mt) and 'N2O-Mt' (Mt). See Appendix on Climate Module for details.	Mapping and conversion of regional GHG emissions to global emissions in the climate module	EQ_CLITOT
CM_HISTORY (year,item)		Climate variable unit [0, ∞); default value: none Default i/e: STD	Default values are provided until 2010. See Appendix on Climate Module for details.	Calibration values for CO2 and forcing	EQ_CLITOT EQ_CLICONC EQ_CLITEMP EQ_CLIBEOH
CM_LINFOR (datayear,item,lim)		Forcing unit per concentration unit [open]; default value: none Default i/e: STD	With lim types LO/UP, CO2 forcing function can be automatically linearized between the concentration levels given. For CH4 and N2O, lim types FX/N must be used (N=concentration multiplier, FX=constant term). See Appendix on Climate Module for details.	Parameters of linearized forcing functions	EQ_CLITOT
CM_MAXC (datayear,item)		Climate variable unit [0, ∞); default value: none Default i/e: none	Since no default inter-/extrapolation, bounds must be explicitly specified for each desired year, unless an explicit inter-/extrapolation option is set. See Appendix on Climate Module for details.	Maximum level of climate variable	EQ_CLIMAX

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
COM_AGG (r,dayayear,c1,c2)		Commodity units [open]; default value: none Default i/e: STD	When commodity type is LO, VAR_COMNET of c1 is aggregated to c2; When commodity type is FX/N, VAR_COMPRD of c1 is aggregated to c2.	Aggregation of commodity NET/PRD production to the production side of the balance of another commodity.	Adds a term in EQ(l)_COMBAL and EQ(l)_COMPRD.
COM_BNDNET (r,datayear,c,s,bd)	rhs_combal, rcs_combal	Commodity unit [open]; default value: none Default i/e: MIG	Since inter-/extrapolation default is MIG, a bound must be specified for each period desired, if no explicit inter-/extrapolation option is given. If the bound is specified for a timeslice s above the commodity timeslice resolution (com_tsl), the bound is applied to the sum of the net commodity variables (VAR_COMNET) below it, according to the timeslice tree. Standard aggregation.	Limit on the net amount of a commodity within a region for a particular timeslice.	The balance constraint is set to an equality (EQE_COMBAL). Either the finer timeslice variables are summed (EQ(l)_BNDNET) or the bound applied direct to the commodity net variable(VAR_COMNET) when at the commodity level (com_tsl).
COM_BNDPRD (r,datayear,c,s,bd)	rhs_comprd, rcs_comprd	Commodity unit [0,∞); default value: none Default i/e: MIG	Since inter-/extrapolation default is MIG, a bound must be specified for each period desired, if no explicit inter-/extrapolation option is given. If the bound is specified for a timeslice s being above the commodity timeslice resolution (com_tsl), the bound is applied to the sum of the commodity production variables	Limit on the amount of a commodity produced within a region for a particular timeslice.	The balance constraint is set to an equality (EQE_COMBAL). Finer timeslice variables summed (EQ(l)_BNDPRD). or the bound is applied direct to the commodity production variable (VAR_COMPRD) when at the commodity level (com_tsl).

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			(VAR_COMPRD) below it, according to the timeslice tree. Standard aggregation.		
COM_BPRICE (r,t,c,s,cur)	COM_ELAST, COM_STEP, COM_VOC	Monetary unit per commodity unit [open]; default value: none Default i/e: none	The control parameter $SET TIMESED 'YES' to activate elastic demands must be set.	Base price of a demand commodity for the elastic demand formulation.	Controls the inclusion of the elastic demand variable (VAR_ELAST) in the commodity balance equation(EQ(l)_COMBAL) Applied to the elastic demand variable (VAR_ELAST) in the objective function (EQ_OBJELS).
COM_CSTNET (r,datayear,c,s,cur)	OBJ_COMNT, CST_COMC, CST_PVC, rhs_combal, rcs_combal	Monetary unit per commodity unit [open]; default value: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Cost on the net amount of a commodity within a region for a particular timeslice.	Forces the net commodity variable (VAR_COMNET) to be included in the equality balance constraint (EQE_COMBAL). Applied to said variable in the cost component of the objective function (EQ_OBJVAR).
COM_CSTPRD (r,datayear,c,s,cur)	OBJ_COMPD, CST_COMC, CST_PVC, rhs_comprd, rcs_comprd	Monetary unit per commodity unit [open]; default value: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Cost on the production of a commodity, within a region for a particular timeslice.	Forces the commodity production variable (VAR_COMPRD) to be included in the equality balance constraint (EQE_COMBAL). Applied to said variable in the cost component of the objective

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
					function (EQ_OBJVAR).
COM_CUMNET (r,y1,y2,bd)	bohyear, eohyear, rhs_combal, rcs_combal, rtc_cumnet	Commodity unit [0,∞); default value: none Default i/e: not possible	The years y1 and y2 may be any years of the set allyear; where y1 may also be 'BOH' for first year of first period and y2 may be 'EOH' for last year of last period.	Bound on the cumulative net amount of a commodity between the years y1 and y2, within a region for a particular timeslice.	Forces the net commodity variable (VAR_COMNET) to be included in the equality balance constraint (EQE_COMBAL). Generates the cumulative commodity constraint (EQ(l)_CUMNET).
COM_CUMPRD (r,y1,y2,bd)	bohyear, eohyear, rhs_comprd, rcs_comprd, rtc_cumprd	Commodity unit [0,∞); default value: none Default i/e: not possible	The years y1 and y2 may be any years of the set allyear; where y1 may also be 'BOH' for first year of first period and y2 may be 'EOH' for last year of last period.	Bound on the cumulative production of a commodity between the years y1 and y2 within a region for a particular timeslice.	Forces the net commodity variable (VAR_COMPRD) to be included in the balance equation (EQE_COMBAL). The cumulative constraint is generated (EQ(l)_CUMPRD).
COM_ELAST (r,datayear,c,s,bd)	COM_BPRICE, COM_STEP, COM_VOC	Dimensionless [open]; default value: none Default i/e: STD	The control parameter $SET TIMESED 'YES' to activate elastic demands must be set. An elasticity is required for each direction the demand is permitted to move. The index bd = 'LO' corresponds to the direction of decreasing the demand, while bd = 'UP' denotes the direction for demand increase. A different value may be provided for each direction,	Elasticity of demand indicating how much the demand rises/falls in response to a unit change in the marginal cost of meeting a demand that is elastic.	Controls the inclusion of the elastic demand variable (VAR_ELAST) in the commodity balance equation(EQ(l)_COMBAL) Applied to the elastic demand variable (VAR_ELAST) in the objective function costs (EQ_OBJELS).

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			thus curves may be asymmetric.		
COM_ELASTX (r,datayear,c,bd)	COM_ELAST	Integer scalar [1,999]; default value: none Default extrapolation: MIG	Provided when shaping of elasticity based upon demand level is desired. Note: Shape index 1 is reserved for constant 1.	Shape index for the elasticity of demand	Affects the demand elasticities applied in EQ_OBJELS
COM_FR (r,datayear,c,s)	COM_PROJ, com_ts, com_tsl, RTCS_TSFR	Decimal fraction [0,1]; default value: timeslice duration (G_YRFR) Default i/e: STD	Only applicable to demand commodities (com_type = 'DEM'). Affects timeslice resolution at which a commodity is tracked (RTCS_TSFR), and thereby may affect when a process cannot operate (rtps_off). Weighted inheritance. Weighted aggregation.	Fraction of the annual demand (COM_PROJ) occurring in timeslice s; describes the shape of the load curve.	Applied to the annual demand (COM_PROJ) as the RHS of the balance equation (EQ(l)_COMBAL). Enters the peaking equation (EQ_PEAK), if a peaking commodity. Applied when setting the upper bound of an elastic demand step (VAR_ELAST).
COM_IE (r,datayear,c,s)		Decimal fraction (0,∞); default value: 1 Default i/e: STD	Direct inheritance. Weighted aggregation.	Infrastructure or transmission efficiency of a commodity.	Overall efficiency applied to the total production of a commodity in the commodity balance equation (EQ(l)_COMBAL).
COM_PKFLX (r,datayear,c,s)	com_peak, com_pkts, COM_PKRSV, FLO_PKCOI	Scalar [open]; default value: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Difference between the average demand and the peak demand in timeslice s, expressed as fraction of the average demand.	Applied to the total consumption of a commodity to raise the capacity needed to satisfy the peaking constraint (EQ_PEAK).
COM_PKRSV (r,datayear,c)	com_peak, com_pkts,	Scalar [0,∞);		Peak reserve margin as fraction of peak	Applied to the total consumption of a

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
	COM_PKFLX, FLO_PKCOI	default value: none Default i/e: STD		demand, e.g. if COM_PKRSV = 0.2, the total installed capacity must exceed the peak load by 20%.	commodity to raise the capacity needed to satisfy the peaking constraint (EQ_PEAK).
COM_PROJ (r,datayear,c)	COM_FR	Commodity unit [0,∞); default value: none Default i/e: STD	Only applicable to demand commodities (com_type = 'DEM').	Projected annual demand for a commodity.	Serves as the RHS (after COM_FR applied) of the commodity balance constraint (EQ(l)_COMBAL). Enters the peaking equation (EQ_PEAK), if a peaking commodity. Applied when setting the upper bound of an elastic demand step (VAR_ELAST).
COM_STEP (r,c,bd)	COM_BPRICE, COM_ELAST, COM_VOC, rcj	Integer number [1,∞); default value: none	The control parameter $SET TIMESED 'YES' to activate elastic demands must be set. The number of steps is required for each direction the demand is permitted to move. The index bd = LO corresponds to the direction of decreasing the demand, while bd = UP denotes the direction for demand increase. A different value may be provided for each direction, thus curves may be asymmetric.	Number of steps to use for the approximation of change of producer/consumer surplus when using the elastic demand formulation.	Controls the instance of the elastic demand variable (VAR_ELAST) in: the commodity balance equation (EQ(l)_COMBAL); setting of the step limit for the elastic demand variable (VAR_ELAST); enters the objective function costs (EQ_OBJELS).

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
COM_SUBNET (r,datayear,c,s,cur)	OBJ_COMNT, CST_COMX, CST_PVC, rhs_combal, rcs_combal	Monetary unit per commodity unit [0,∞); default value: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Subsidy on the net amount of a commodity within a region for a particular timeslice.	Forces the net commodity variable (VAR_COMNET) to be included in the equality balance constraint (EQE_COMBAL). Applied (-) to said variable in the cost component of the objective function (EQ_OBJVAR).
COM_SUBPRD (r,datayear,c,s,cur)	OBJ_COMPD, CST_COMX, CST_PVC, rhs_comprd, rcs_comprd	Monetary unit per commodity unit [0,∞); default value: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Subsidy on the production of a commodity within a region for a particular timeslice.	Forces the commodity production variable (VAR_COMPRD) to be included in the equality balance constraint (EQE_COMBAL). Applied (-) to said variable in the cost component of the objective function (EQ_OBJVAR).
COM_TAXNET (r,datayear,c,s,cur)	OBJ_COMNT, CST_COMX, CST_PVC, rhs_combal, rcs_combal	Monetary unit per commodity unit [0,∞); default value: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Tax on the net amount of a commodity within a region for a particular timeslice.	Forces the net commodity variable (VAR_COMNET) to be included in the equality balance constraint (EQE_COMBAL). Applied to said variable in the cost component of the objective function (EQ_OBJVAR).
COM_TAXPRD (r,datayear,c,s,cur)	OBJ_COMPD, CST_COMX, CST_PVC,	Monetary unit per commodity unit [0,∞);	Direct inheritance. Weighted aggregation.	Tax on the production of a commodity within a region for a	Forces the commodity production variable (VAR_COMPRD) to be

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
	rhs_comprd, rcs_comprd	default value: none Default i/e: STD		particular timeslice.	included in the equality balance constraint (EQE_COMBAL). Applied to said variable in the cost component of the objective function (EQ_OBJVAR).
COM_VOC (r,datayear,c,bd)	COM_BPRICE, COM_STEP, COM_ELAST	Dimensionless [0,∞); default: none Default i/e: STD	The control parameter $SET TIMESED 'YES' to activate elastic demands must be set. A number is required for each direction the demand is permitted to move. The index bd = LO corresponds to the direction of decreasing the demand, while bd = UP denotes the direction for demand increase. A different value may be provided for each direction, thus curves may be asymmetric.	Possible variation of demand in both directions when using the elastic demand formulation.	Applied when setting the bound of an elastic demand step (VAR_ELAST). Applied to the elasticity variable in the objective function costs (EQ_OBJELS).
DAM_BQTY (r,c)	DAM_COST	Commodity unit [0,∞); default value: none Default i/e: N/A	Only effective when DAM_COST has been defined for commodity c.	Base quantity of emissions for damage cost accounting	EQ_DAMAGE EQ_OBJDAM
DAM_COST (r,datayear,c,cur)	DAM_BQTY	Monetary unit per commodity unit [0,∞); default value: none Default i/e: STD	Damage costs are by default endogenous (included in the objective). To set them exogenous, use $SET DAMAGE NO	Marginal damage cost of emissions at Base quantity.	EQ_DAMAGE EQ_OBJDAM
DAM_ELAST (r,c,lim)	DAM_COST DAM_BQTY	Dimensionless [0,∞);	Only effective when DAM_COST has been	Elasticity of damage cost in the lower or	EQ_OBJDAM

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		default value: none Default i/e: N/A	defined for commodity c.	upper direction from Base quantity.	
DAM_STEP (r,c,lim)	DAM_COST DAM_BQTY	Integer number [1,∞); default value: none Default i/e: N/A	Only effective when DAM_COST has been defined for commodity c.	Number of steps for linearizing damage costs in the lower or upper direction from Base quantity.	EQ_DAMAGE EQ_OBJDAM
DAM_VOC (r,c,lim)	DAM_COST DAM_BQTY	Decimal fraction LO: [0,1]; UP: [0,∞); default value: none Default i/e: N/A	Only effective when DAM_COST has been defined for commodity c.	Variance of emissions in the lower or upper direction from Base quantity as a fraction of Base quantity.	EQ_OBJDAM
E (t)	B, D, M, COEF_CPT, rtp_vintyr		For each modelyear period	End year of period t, used in determining the length of each period	The amount of new investment (VAR_NCAP) carried over in the capacity transfer constraint (EQ(l)_CPT). Amount of investments (VAR_NCAP) remaining past the modelling horizon that needs to be credited back to the objective function (EQ_OBJINV).
FLO_BND (r,datayear,p,cg,s,bd)		Commodity unit [0,∞); default: none Default i/e: MIG	If the bound is specified for a timeslice s being above the flow timeslice resolution (rtpcs_varf), the bound is applied to the sum of the flow variables (VAR_FLO) according to the timeslice tree, otherwise directly to the flow variable.	Bound on the flow of a commodity or the sum of flows within a commodity group.	Flow activity limit constraint (EQ(l)_FLOBND) when s is above rtpcs_varf Direct bound on activity variable (VAR_FLO) when at the rtpcs_varf level.

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			No aggregation.[29]		
FLO_COST (r,datayear,p,c,s,cur)	OBJ_FCOST, CST_FLOC, CST_PVP	Monetary unit per commodity unit [open]; default: none Default i/e: STD	Direct inheritance Weighted aggregation	Variable cost of a process associated with the production/ consumption of a commodity.	Applied to the flow variable (VAR_FLO) when entering the objective function (EQ_OBJVAR). May appear in user constraints (EQ_UC*) if specified in UC_NAME.
FLO_CUM (r,p,c,y1,y2,bd)	ACT_CUM	Flow unit [0,∞); default value: none Default i/e: N/A	The years y1 and y2 may be any years of the set allyear; where y1 may also be 'BOH' for first year of first period and y2 may be 'EOH' for last year of last period.	Bound on the cumulative amount of annual process activity between the years y1 and y2, within a region.	Generates an instance of the cumulative constraint (EQ_CUMFLO)
FLO_DELIV (r,datayear,p,c,s,cur)	OBJ_FDELV, CST_FLOC, CST_PVP	Monetary unit per commodity unit [open]; default: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Cost of a delivering (consuming) a commodity to a process.	Applied to the flow variable (VAR_FLO) when entering the objective function (EQ_OBJVAR). May appear in user constraints (EQ_UC*) if specified in UC_NAME.
FLO_EFF (r,datayear,p,cg,c,s)		Commodity unit of c / commodity unit of cg [open]; default value: none Default i/e: STD	Inherited/aggregated to the timeslice levels of the flow variables of the commodities in group cg. All parameters with the same process (p) and target commodity (c) are combined in the same	Defines the amount of commodity flow of commodity (c) per unit of other process flow(s) or activity (cg).	Generates process transformation equation (EQ_PTRANS) between one or more input (or output) commodities and one output (or input) commodities.

[29] Standard aggregation not implemented for FLO_BND.

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			transformation equation.		
FLO_EMIS (r,datayear,p,cg,com,s)	FLO_EFF (alias)	Commodity unit of c / commodity unit of cg [open]; default value: none Default i/e: STD	See FLO_EFF. If com is of type ENV and is not in the process topology, it is added to it as an output flow.	Defines the amount of emissions (c) per unit of process flow(s) or activity (cg).	See FLO_EFF.
FLO_FR (r,datayear,p,c,s,bd)		Decimal fraction [0,1] / [0,∞); default value: none Default i/e: MIG	FLO_FR may be specified as lower, upper or fixed bounds, in contrast to COM_FR. Can be specified for any flow variable having a subannual timeslice resolution. Weighted aggregation. Direct inheritance, if defined at the ANNUAL level.	1) Bounds the flow of commodity (c) entering or leaving process (p) in a timeslice, in proportion to annual flow. 2) If specified also at the ANNUAL level, bounds the flow **level** in proportion to the average level under the parent timeslice	A share equation (EQ(l)_FLOFR) limiting the amount of commodity (c) is generated according to the bound type (bd = l indicator).
FLO_FUNC (r,datayear,p,cg1,cg2,s)	FLO_SUM, FLO_FUNCX, COEF_PTRAN, rpc_ffunc, rpcg_ptran	Commodity unit of cg2/commodity unit of cg1 [open]; default value: see next column Default i/e: STD	If for the same indexes the parameter FLO_SUM is specified but no FLO_FUNC, the FLO_FUNC is set to 1. Important factor in determining the level at which a process operates in that the derived transformation parameter (COEF_PTRAN) is inherited/aggregated to the timeslice levels of the flow variables associated with the commodities in the group cg1.	A key parameter describing the basic operation of or within a process. Sets the ratio between the sum of flows in commodity group cg2 to the sum of flows in commodity group cg1, thereby defining the efficiency of producing cg2 from cg1 (subject to any FLO_SUM). cg1 and cg2 may be also single commodities.	Establishes the basic transformation relationship (EQ_PTRANS) between one or more input (or output) commodities and one or more output (or input) commodities. Establishes the relationship between storage charging / discharging and a related commodity flow (VAR_FLO) in the auxiliary storage flow equation (EQ_STGAUX).
FLO_FUNCX	FLO_FUNC,	Integer scalar	Provided when shaping	Age-based shaping	Applied to the flow

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
(r,datayear,p,cg1,cg2)	FLO_SUM, COEF_PTRAN	[1,999]; default value: none Default extrapolation: MIG	based upon age is desired. Vintaged processes only. Note: Shape index 1 is reserved for constant 1. ACT_EFF(cg): cg1=cg, cg2='ACT' ACT_FLO(cg): cg1='ACT', cg2=cg FLO_EMIS(cg,c): cg1=cg2=c FLO_EFF(cg,c): cg1=cg2=c FLO_FUNC(cg1,cg2): cgN=cgN	curve (SHAPE) to be applied to the flow parameters (ACT_EFF/ACT_FLO/ FLO_FUNC/FLO_SUM/ FLO_EMIS/FLO_EFF)	variable (VAR_FLO) in a transformation equation (EQ_PTRANS / EQE_ACTEFF) to account for changes in the transformation efficiency according to the age of each process vintage.
FLO_MARK (r,datayear,p,c,bd)	PRC_MARK	Decimal fraction [0,1]; default value: none Default i/e: STD	The same given fraction is applied to all timeslices of the commodity (this could be generalized to allow time-slice-specific fractions, if deemed useful). If an ANNUAL level market-share is desired for a timesliced commodity, PRC_MARK can be used instead.	Process-wise market share in total commodity production.	The individual process flow variables (VAR_FLO, VAR_IN, VAR_STGIN/OUT) are constrained (EQ(l)_FLMRK) to a fraction of the total production of a commodity (VAR_COMPRD). Forces the commodity production variable (VAR_COMPRD) to be included in the equality balance constraint (EQE_COMBAL).
FLO_PKCOI (r,datayear,p,c,s)	COM_PKRSV, COM_PKFLX, com_peak, com_pkts	Scalar [open]; default value: 1 Default i/e: STD	FLO_PKCOI is specified for individual processes p consuming the peak commodity c. Direct inheritance. Weighted aggregation. Used when the timeslices are not necessarily fine	Factor that permits attributing more (or less) demand to the peaking equation (EQ_PEAK) than the average demand calculated by the model, to handle the situation where peak	Applied to the flow variable (VAR_FLO) to adjust the amount of a commodity consumed when considering the average demand contributing to the peaking constraint (EQ_PEAK).

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			enough to pick up the actual peak within the peak timeslices.	usage is typically higher (or lower) due to coincidental (or non-coincidental) loads at the time of the peak demand.	
FLO_SHAR (r,datayear,p,c,cg,s,bd)		Decimal fraction [0,1]; default value: none Default i/e: MIG over milestoneyears, STD over pastyears	Direct inheritance. Weighted aggregation. A common example of using FLO_SHAR is to specify the power-to-heat ratio of CHP plants in the backpressure point. For example, for a heat output of a CHP technology, the FLO_SHAR parameter would have the value CHPR/(1+CHPR), with CHPR being the heat-to-power ratio.	Share of flow commodity c based upon the sum of individual flows defined by the commodity group cg belonging to process p.	When the commodity is an input an EQ(l)_INSHR equation is generated. When the commodity is an output an EQ(l)_OUTSHR equation is generated.
FLO_SUB (r,datayear,p,c,s,cur)	OBJ_FSUB, CST_FLOX, CST_PVP	Monetary unit per commodity unit [0,∞); default value: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Subsidy on a process flow.	Applied with a minus sign to the flow variable (VAR_FLO) when entering the objective function (EQ_OBJVAR). May appear in user constraints (EQ_UC*) if specified in UC_NAME.
FLO_SUM (r,datayear,p,cg1,c,cg2, s)	FLO_FUNC FLO_FUNCX COEF_PTRANS, fs_emis, rpc_emis, rpc_ffunc, rpcg_ptran	Commodity unit of cg2/commodity unit of c [open]; default value: see next column Default i/e: STD	If a FLO_SUM is specified and no corresponding FLO_FUNC, the FLO_FUNC is set to 1. If FLO_FUNC is specified for a true commodity group cg1, and no FLO_SUM is	Multiplier applied for commodity c of group cg1 corresponding to the flow rate based upon the sum of individual flows defined by the	The FLO_SUM multiplier is applied along with FLO_FUNC parameter in the transformation coefficient (COEF_PTRANS), which is applied to the flow

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			specified for the commodities in cg1, these FLO_SUM are set to 1. The derived parameter COEF_PTRANS is inherited/aggregated to the timeslice level of the flow variable of the commodity c.	commodity group cg2 of process p. Most often used to define the emission rate, or to adjust the overall efficiency of a technology based upon fuel consumed.	variable (VAR_FLO) in the transformation equation (EQ_PTRANS).
FLO_TAX (r,datayear,p,c,s,cur)	OBJ_FTAX, CST_FLOX, CST_PVP	Monetary unit per commodity unit [0,∞); default: none Default i/e: STD	Direct inheritance. Weighted aggregation.	Tax on a process flow.	Applied to the flow variable (VAR_FLO) when entering the objective function (EQ_OBJVAR). May appear in user constraints (EQ_UC*) if specified in UC_NAME.
G_CUREX (cur1,cur2)	R_CUREX	Scalar (0,∞) Default value: none	The target currency cur2 must have a discount rate defined with G_DRATE.	Conversion factor from currency cur1 to currency cur2, with cur2 to be used in the objective function.	Affects cost coefficients in EQ_OBJ
G_DRATE (r,allyear,cur)	OBJ_DISC, OBJ_DCEOH, NCAP_DRATE, COR_SALVI, COR_SALVD, COEF_PVT VDA_DISC	Decimal fraction (0,1]; default value = none Default i/e: STD	A value must be provided for each region. Interpolation is dense (all individual years included).	System-wide discount rate in region r for each time-period.	The discount rate is taken into consideration when constructing the objective function discounting multiplier (OBJ_DISC), which is applied in each components of the objective function (EQ_OBJVAR, EQ_OBJINV, EQ_OBJFIX, EQ_OBJSALV,

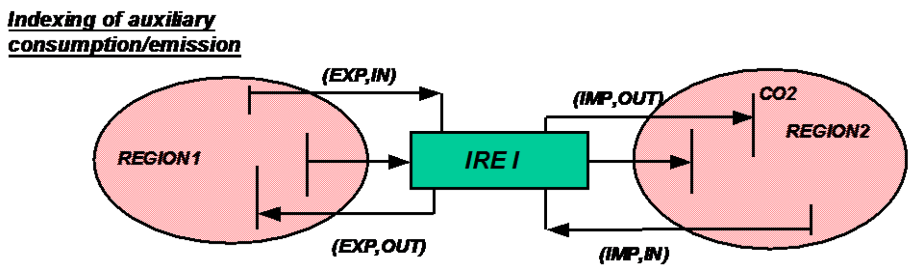
Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
					EQ_OBJELS).
G_DYEAR	OBJ_DISC COEF_PVT	Year [BOTIME,EOTIME]; default value = M(MIYR_1), i.e. the first milestone year		Base year for discounting.	The year to which all costs are to be discounted is taken into consideration when constructing the objective function discounting multiplier (OBJ_DISC), which is applied in each of the components of the objective function (EQ_OBJVAR, EQ_OBJINV, EQ_OBJFIX, EQ_OBJSALV, EQ_OBJELS).
G_ILEDNO	NCAP_ILED	Decimal fraction [0,1]; default value: 0.1	Only provided when the costs associated with the lead-time for new capacity (NCAP_ILED) are not to be included in the objective function. Not taken into account if the OBLONG switch or any alternative objective formulation is used.	If the ratio of lead-time (NCAP_ILED) to the period duration (D) is below this threshold then the lead-time consideration will be ignored in the objective function costs.	Prevents the investment costs associated with investment lead-times from energy the investment component of the objective function (EQ_OBJINV).
G_NOINTERP	All parameters that are normally subjected to interpolation / extrapolation	Binary indicator [0 or 1]; default value = 0	Only provide when interpolation / extrapolation is to be turned off for all parameters. Interpolation of cost parameters is always done.	Switch for generally turning-on (= 0) and turning-off (= 1) sparse inter- / extrapolation.	
G_OFFTHD	PRC_NOFF	Scalar	Setting G_OFFTHD=1 will	Threshold for	Affects availability of

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
(datayear)	PRC_AOFF PRC_FOFF COM_OFF	[0,1] Default value: 0 Default i/e: 5	make the *_OFF attributes effective only for periods fully included in the OFF range specified.	considering an *_OFF attribute disabling a process/commodity variable in period.	VAR_NCAP, VAR_ACT, VAR_FLO, VAR_COMNET/PRD
G_OVERLAP		Scalar [0,100] Default value: TIMESTEP/2	Used only when time-stepped solution is activated with the TIMESTEP control variable.	Overlap of stepped solutions (in years).	–
G_TLIFE	NCAP_TLIFE	Scalar [1,∞); default value = 10		Default value for the technical lifetime of a process if not provided by the user.	
G_YRFR (all_r,s)	RTCS_TSFR, RS_STGPRD	Fraction [0,1]; default value: none; only for the ANNUAL timeslice a value of 1 is predefined	Must be provided for each region and timeslice.	Duration of timeslice s as fraction of a year. Used for shaping the load curve and lining up timeslice duration for inter-regional exchanges.	Applied to various variables (VAR_NCAP+PASTI, VAR_COMX, VAR_IRE, VAR_FLO, VAR_SIN/OUT) in the commodity balance equation (EQ(l)_COMBAL).
IRE_BND (r,datayear,c,s,all_r,ie, bd)	top_ire	Commodity unit [0,∞); default value: none Default i/e: MIG	Only applicable for inter-regional exchange processes (IRE). If the bound is specified for a timeslice (s) being above the commodity (c) timeslice resolution, the bound is applied to the sum of the imports/exports according to the timeslice tree. Standard aggregation.	Bound on the total import (export) of commodity (c) from (to) region all_r in (out of) region r.	Controls the instances for which the trade bound constraint (EQ(l)_IREBND) is generated, and the RHS.
IRE_CCVT (r1,c1,r2,c2))	IRE_TSCVT, top_ire	Scalar (0,∞)	Required for mapping commodities involved in	Conversion factor between commodity	The conversion factor is applied to the flow

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter- /extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		Default value: 1 if commodity names are the same in both regions I/e: N/A	inter-regional exchanges between two regions whenever commodities traded are in different units in the regions.	units in region r1 and region r2. Expresses the amount of commodity c2 in region r2 equivalent to 1 unit of commodity c1 in region r1.	variable (VAR_IRE) in the inter-regional balance constraint (EQ_IRE). Similarly, applied to the flow variable (VAR_IRE) when an inter-regional exchange is bounded in the limit constraint (EQ(l)_IREBND). Similarly, applied to the flow variable (VAR_IRE) when an exchange with an external region is bounded (EQ(l)_XBND).
IRE_FLO (r1,datayear,p,c1,r2,c2, s2)	top_ire	Commodity unit c2/commodity unit c1 [0,∞); default value: 1 Default i/e: STD	Only applicable for inter-regional exchange processes (IRE) between two internal regions. Note that for each direction of trade a separate IRE_FLO needs to be specified. Similar to FLO_FUNC for standard processes. Direct inheritance. Weighted aggregation.	Efficiency of exchange process from commodity c1 in region r1 to commodity c2 in the region2 in timeslice s2; the timeslice s2 refers to the r2 region.	Applied to the exchange flow variable (VAR_IRE) in the inter-regional trade equation (EQ_IRE). Applied to the exchange flow variable (VAR_IRE) when a bound on inter-regional trade is to be applied (EQ(l)_IREBND).
IRE_FLOSUM (r,datayear,p,c1,s,ie,c2, io)	top_ire	Commodity unit c2/commodity unit c1 [open]; default value: none Default i/e: STD	Only applicable for inter-regional exchange processes (IRE). Since the efficiency IRE_FLO can only be used for exchange between internal regions, IRE_FLOSUM may be used	Auxiliary consumption (io = IN, owing to the commodity entering the process) or production/ emission (io = OUT, owing to the commodity leaving the process)	The multiplier is applied to the flow variable (VAR_IRE) associated with an inter-reginal exchange in the commodity balance constraint (EQ(l)_COMBAL).

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			to define an efficiency for an import/export with an external region by specifying the same commodity for c1 and c2 and the value 1-efficiency as auxiliary consumption. Direct inheritance. Weighted aggregation.	of commodity c2 due to the IMPort / EXPort (index ie) of the commodity c1 in region r[30]	If a flow share (FLO_SHAR) is provided for an inter-regional exchange process then the multiplier is applied to the flow variable (VAR_IRE) in the share constraint (EQ(l)_IN/OUTSHR). If a cost is provided for the flow (FLO_COST or FLO_DELIV) then the factor is applied to the flow variable (VAR_IRE) in the variable component of the objective function (EQ_OBJVAR).
IRE_PRICE (r,datayear,p,c,s,all_r,ie,cur)	OBJ_IPRIC, CST_COMC, CST_PVP, top_ire	Monetary unit / commodity unit [0,∞); default value: none Default i/e: STD	Only applicable for inter-regional exchange processes (IRE). Ignored if all_r is an internal region.	IMPort/EXPort price (index ie) for to/from an internal region of a commodity (c) originating	The price of the exchange commodity is applied to the trade flow variable (VAR_IRE) in the variable costs

[30] The indexing of auxiliary consumption flows or emissions of inter-regional exchange processes is illustrated in the figure below.



Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			Direct inheritance. Weighted aggregation.	from/heading to an external region all_r.	component of the objective function (EQ_OBJVAR).
IRE_TSCVT (r1,s1,r2,s2)	IRE_CCVT, top_ire	Scalar (0,∞); default value: 1 if timeslice tree and names are the same in both regions I/e: N/A	Used for mapping timeslices in different regions. Required if timeslice definitions are different in the regions.	Matrix for mapping timeslices; the value for (r1,s1,r2,s2) gives the fraction of timeslice s2 in region r2 that falls in timeslice s1 in region r1.	The conversion factor is applied to the flow variable (VAR_IRE) in the inter-regional balance constraint (EQ_IRE). Similarly, applied to the flow variable (VAR_IRE) when an inter-regional exchange is bounded in the limit constraint (EQ(l)_IREBND). Similarly, applied to the flow variable (VAR_IRE) when an exchange with an external region is bounded (EQ(l)_XBND).
IRE_XBND (all_r,datayear,c,s ie,bd)	top_ire	Commodity unit [0,∞); default value: none Default i/e: MIG	Only applicable for inter-regional exchange processes (IRE). Provide whenever a trade flow is to be constrained. Note that the limit is either imposed by summing lower or splitting higher flow variables (VAR_IRE) when specified at other than the actual flow level (as determined by the commodity and process levels (COM_TSL/ PRC_TSL	Bound on the total IMPort (EXPort) (index ie) of commodity c in region all_r with all sources (destinations).	The trade limit equation EQ(l)_XBND generated either sums lower flow variables (VAR_IRE) or splits (according to the timeslice tree) coarser variables.

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			).		
MULTI (j,allyear)	NCAP_AFM, NCAP_FOMM, NCAP_FSUBM, NCAP_FTAXM	Scalar [open]; default value: none I/e: Full dense interpolation and extrapolation	Only provided when the related shaping parameters are to be used.	Multiplier table used for any shaping parameters (*_*M) to adjust the corresponding technical data as function of the year; the table contains different multiplier curves identified by the index j.	*{See Related Parameters}*
NCAP_AF (r,datayear,p,s,bd)	NCAP_AFA, NCAP_AFS, NCAP_AFM, NCAP_AFX, COEF_AF	Decimal fraction [0,1]; default value: 1 Default i/e: STD Remark: In special cases values >1 can also be used (when PRC_CAPACT does not represent the max. technical level of activity per unit of capacity).	NCAP_AF, NCAP_AFA and NCAP_AFS can be applied simultaneously. Direct inheritance. Weighted aggregation. (Important remark: No inheritance/aggregation if any value is specified at process timeslices.)	Availability factor relating a unit of production (process activity) in timeslice s to the current installed capacity.	The corresponding capacity-activity constraint (EQ(l)_CAPACT) will be generated for any timeslice s. If the process timeslice level (PRC_TSL) is below said level, the activity variables will be summed.
NCAP_AFA (r,datayear,p,bd)	NCAP_AFA, NCAP_AFS, NCAP_AFM, NCAP_AFX, COEF_AF	Decimal fraction [0,1]; default value: none Default i/e: STD Remark: In special cases values >1 can also be used (when PRC_CAPACT has been chosen not to represent the max.	Provided when 'ANNUAL' level process operation is to be controlled. NCAP_AF, NCAP_AFA and NCAP_AFS can be applied simultaneously. NCAP_AFA is always assumed to be non-vintage dependent, even if the process is defined as a vintaged one; for vintage-	Annual availability factor relating the annual activity of a process to the installed capacity.	The corresponding capacity-activity constraint (EQ(l)_CAPACT) will be generated for the 'ANNUAL' timeslice. If the process timeslice level (PRC_TSL) is below said level, the activity variables will be summed.

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		technical level of activity per unit of capacity).	dependent annual availability NCAP_AFS with s='ANNUAL' can be used.		
NCAP_AFC (r,datayear,p,cg,tsl)	NCAP_AFCS	Decimal fraction [0,∞); default value: none Default i/e: STD	If the commodities are in the PCG, constraint is applied to the flows in the PCG as a whole (linear combination of flows). Independent equations are generated for commodities not in the PCG, or when NCAP_AFC(r,'0',p,'ACT',tsl) =−1 is also specified.	Commodity-specific availability of capacity for commodity group cg, at given timeslice level.	Generates instances of EQ(l)_CAFLAC (thereby disabling EQ(l)_CAPACT generation), or EQL_CAPFLO.
NCAP_AFCS (r,datayear,p,cg,ts)	NCAP_AFC	Decimal fraction [0,∞); default value: none Default i/e: STD	See NCAP_AFC. NCAP_AFCS is similar to NCAP_AFC but is defined on individual timeslices. Overrides NCAP_AFC.	Commodity-specific availability of capacity for commodity group cg, timeslice-specific.	See NCAP_AFC.
NCAP_AFM (r,datayear,p)	NCAP_AF, NCAP_AFA, NCAP_AFS, MULTI, COEF_AF	Integer number Default value: 0 (no multiplier applied) Default extrapolation: MIG	Provided when multiplication of NCAP_AF / NCAP_AFS based upon year is desired. Note: Multiplier index 1 is reserved for constant 1.	Period sensitive multiplier curve (MULTI) to be applied to the availability factor parameters (NCAP_AF/AFA/AFS) of a process.	*{See Related Parameters}*
NCAP_AFS (r,datayear,p,s,bd)		Decimal fraction [0,1]; default value: none Default i/e: STD Remark: In special cases values >1 can also be used (in cases where PRC_CAPACT has been chosen not	NCAP_AF, NCAP_AFA and NCAP_AFS can be applied simultaneously. NCAP_AFS being specified for timeslices s being below the process timeslice level are ignored. No inheritance. No aggregation.	Availability factor relating the activity of a process in a timeslice s being at or above the process timeslice level (prc_tsl) to the installed capacity. If for example the process timeslice	The corresponding capacity-activity constraint (EQ(l)_CAPACT) will be generated for a timeslice s being at or above the process timeslice level (prc_tsl). If the process timeslice level is below said level,

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		to represent the maximum technical level of activity per unit of capacity).	Can be used also on the process timeslices, and will then override the levelized NCAP_AF availability factors.	level is 'DAYNITE' and NCAP_AFS is specified for timeslices on the 'SEASONAL' level, the sum of the 'DAYNITE' activities within a season are restricted, but not the 'DAYNITE' activities directly.	the activity variables will be summed.
NCAP_AFX (r,datayear,p)	NCAP_AF, NCAP_AFA, NCAP_AFS, SHAPE, COEF_AF	Integer number Default value: 0 (no shape curve applied) Default extrapolation: MIG	Provided when shaping based upon age is desired. NCAP_AFX is applied to NCAP_AF and NCAP_AFS, but not the annual availability NCAP_AFA. For non-vintaged process, the SHAPE parameter is only applied to NCAP_AF, i.e. availabilities at process timeslices will be vintaged. Note: Shape index 1 is reserved for constant 1.	Age-based shaping curve (SHAPE) to be applied to the availability factor parameters (NCAP_AF/AFA/AFS) of a process.	*{See Related Parameters}*
NCAP_BND (r,datayear,p,bd)		Capacity unit [0,∞); default value: none Default i/e: MIG	Provided for each process to have its overall installed capacity (VAR_NCAP) limited in a period. Since inter-/extrapolation default is MIG, a bound must be specified for each period desired, if no explicit inter-/extrapolation option is given, e.g. NCAP_BND(R,'0',P) =2.	Bound on the permitted level on investment in new capacity	Imposes an indirect limit on the capacity transfer equation (EQ_CPT) by means of a direct bound on the new investments capacity variable (VAR_NCAP).
NCAP_BPME (r,datayear,p)	NCAP_CDME	Decimal fraction [0,∞);	The parameter is only taken into account when	Back pressure mode efficiency (or total	Process transformation equation, either

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		default value: none Default i/e: STD	the process is of type CHP, and NCAP_CDME has been also defined.	efficiency in full CHP mode).	EQE_ACTEFF or EQ_PTRANS
NCAP_CDME (r,datayear,p)	NCAP_BPME	Decimal fraction [0,∞); default value: none Default i/e: STD	The parameter can only be used for standard processes having electricity output in the PCG. The efficiency is applied between the default shadow group and the electricity. If the process is also defined as a CHP, heat efficiency is also included.	Condensing mode efficiency	Process transformation equation, either EQE_ACTEFF or EQ_PTRANS
NCAP_CEH (r,datayear,p)	NCAP_CHPR ACT_EFF	Decimal fraction [−1,∞]; default value: none Default i/e: STD	The parameter is only taken into account when the process is defined to be of type CHP. According to the CEH value, the process activity will be defined as: CEH ≤ 0: Max. electricity output according to CHPR 0 < CEH ≤1: Condensing mode electricity output CEH ≥ 1: Total energy output in full CHP mode.	Coefficient of electricity to heat along the iso-fuel line in a pass-out CHP technology.	Process transformation equation, either EQE_ACTEFF or EQ_PTRANS
NCAP_CHPR (r,datayear,p,bd)	FLO_SHAR	Decimal fraction [0,∞); default value: none Default i/e: STD	The parameter is only taken into account when the process is defined to be of type CHP.	Heat-to-power ratio of a CHP technology (fixed / minimum / maximum ratio).	The ratios are implemented with EQ(l)_OUTSHR
NCAP_CLAG (r,datayear,p,c,io)	NCAP_CLED NCAP_COM	Years [open]; default value: none Default i/e: STD	Provided when there is a delay in commodity output after commissioning new capacity. So, if the process is available in the year K, the commodity is produced	Lagtime of a commodity after new capacity is installed.	Applied to the investment variable (VAR_NCAP) in the commodity balance (EQ(l)_COMBAL) of the investment period or

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			during the years [K+CLAG, K+NCAP_TLIFE−1].		previous periods.
NCAP_CLED (r,datayear,p,c)	NCAP_ICOM COEF_ICOM	Years [open]; default value: = NCAP_ILED Default i/e: STD	Provided when a commodity must be available prior to availability of a process. So, if the process is available in the year B(v) +NCAP_ILED−1, the commodity is produced during the time span [B(v)+ILED−CLED, B(v) +NCAP_ILED−1]. Usually used when modelling the need for fabrication of reactor fuel the period before a reactor goes online.	Lead time requirement for a commodity during construction (NCAP_ICOM), prior to the initial availability of the capacity.	Applied to the investment variable (VAR_NCAP) in the commodity balance (EQ(l)_COMBAL) of the investment period or previous periods.
NCAP_COM (r,datayear,p,c,io)	rpc_capflo, rpc_conly	Commodity unit per capacity unit [open]; default value: none Default i/e: STD	Provided when the consumption or production of a commodity is tied to the level of the installed capacity.	Emission (or land-use) of commodity c associated with the capacity of a process for each year said capacity exists.	Applied to the capacity variable (VAR_CAP) in the commodity balance (EQ_COMBAL).
NCAP_COST (r,datayear,p)	OBJ_ICOST, OBJSCC, CST_INVC, CST_PVP	Monetary unit per capacity unit [0,∞); default value: none Default i/e: STD	Provided whenever there is a cost associated with putting new capacity in place.	Investment costs of new installed capacity according to the installation year.	Applied to the investment variable (VAR_NCAP) when entering the objective function (EQ_OBJNV). May appear in user constraints (EQ_UC*) if specified in UC_NAME.
NCAP_DCOST (r,datayear,p,cur)	NCAP_DLAG, COR_SALVD, OBJ_DCOST,	Monetary unit per capacity unit [0,∞); default value:	Provided when there are decommissioning costs associated with a process.	Cost of dismantling a facility after the end of its lifetime.	Applied to the current capacity subject to decommissioning

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
	CST_DECC, CST_PVP	none Default i/e: STD	Decommissioning of a process and the payment of decommissioning costs may be delayed by a lag time (NCAP_DLAG).		(VAR_NCAP+NCAP_PASTI) when entering the objective function (EQ_OBJNV).
NCAP_DELIF (r,datayear,p)	NCAP_DLIFE, COR_SALVD, DUR_MAX, OBJ_CRFD, SALV_DEC	Years (0,∞); default value: NCAP_DLIFE Default i/e: STD	Provided when the timeframe for paying for decommission is different from that of the actual decommissioning.	Economic lifetime of the decommissioning activity.	Applied to the investment variable (VAR_NCAP) when entering the salvage portion of the objective function (EQ_OBJSALV).
NCAP_DISC (r,datayear,p,unit)	rp_dscncap	Capacity unit [0,∞); default value: none Default i/e: MIG	Used for lumpy investments. Requires MIP. Since inter-/extrapolation default is MIG, a value must be specified for each period desired, if no explicit inter-/extrapolation option is given.	Size of capacity units that can be added.	Applied to the lumpy investment integer variable (VAR_DNCAP) in the discrete investment equation (EQ_DSCNCAP) to set the corresponding standard investment variable level (VAR_NCAP).
NCAP_DLAG (r,datayear,p)	COEF_OCOM, DUR_MAX, OBJ_DLAGC	Years [0,∞); default value: none Default i/e: STD	Provided when there is a lag in the decommissioning of a process (e.g., to allow the nuclear core to reduce its radiation).	Number of years delay before decommissioning can begin after the lifetime of a technology has ended.	Delay applied to a decommissioning flow (VAR_FLO) in the balance equation (EQ(l)_COMBAL) as production. Delay applied to the current capacity subject to decommissioning (VAR_NCAP+NCAP_PASTI) when entering the objective function components

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
					(EQ_OBJINV, EQ_OBJFIX, EQ_OBJSALV).
NCAP_DLAGC (r,datayear,p,cur)	NCAP_DLAG, OBJ_DLAGC, CST_DECC, CST_PVP	Monetary unit per capacity unit [0,∞); default value: none Default i/e: STD	Provided when there is a cost during any lag in the decommissioning (e.g., security).	Cost occurring during the lag time after the technical lifetime of a process has ended and before its decommissioning starts.	Cost during delay applied to the current capacity subject to decommissioning (VAR_NCAP+NCAP_PASTI) when entering the objective function components (EQ_OBJFIX, EQ_OBJSALV).
NCAP_DLIFE (r,datayear,p)	DUR_MAX	Years (0,∞); default value: none Default i/e: STD	Provided when a process has a decommissioning phase.	Technical time for dismantling a facility after the end its technical lifetime, plus any lag time (NCAP_DLAG).	Decommissioning time impacting (VAR_NCAP+NCAP_PASTI) when entering the objective function components (EQ_OBJINV, EQ_OBJSALV).
NCAP_DRATE (r,datayear,p)	G_DRATE, COR_SALVI, COR_SALVD	Percent (0,∞); default value: G_DRATE Default i/e: STD	Provided if the cost of borrowing for a process is different from the standard discount rate.	Technology specific discount rate.	Discount rate applied to investments (VAR_NCAP+NCAP_PASTI) when entering the objective function components (EQ_OBJINV, EQ_OBJSALV).
NCAP_ELIFE (r,datayear,p)	NCAP_TLIFE, COR_SALVI, OBJ_CRF	years (0,∞); default value: NCAP_TLIFE Default i/e: STD	Provided only when the economic lifetime differs from the technical lifetime (NCAP_TLIFE).	Economic lifetime of a process.	Economic lifetime of a process when costing investment (VAR_NCAP+NCAP_PASTI) or capacity in the objective function

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
					components (EQ_OBJINV, EQ_OBJSALV, EQ_OBJFIX).
NCAP_FOM (r,datayear,p,cur)	OBJ_FOM, CST_FIXC, CST_PVP	Monetary unit per capacity unit [0,∞); default value: none Default i/e: STD	Provided when there is a fixed cost associated with the installed capacity.	Fixed operating and maintenance cost per unit of capacity according to the installation year.	Fixed operating and maintenance costs associated with total installed capacity (VAR_NCAP+NCAP_PASTI) when entering the objective function components (EQ_OBJFIX).
NCAP_FOMM (r,datayear,p)	NCAP_FOM, MULTI	Integer number Default value: 0 (no multiplier curve applied) Default i/e: MIG	Provided when shaping based upon the period is desired. Note: Multiplier index 1 is reserved for constant 1.	Period sensitive multiplier curve (MULTI) applied to the fixed operating and maintenance costs (NCAP_FOM).	*{See Related Parameters}*
NCAP_FOMX (r,datayear,p)	NCAP_FOM, SHAPE	Integer number Default value: 0 (no shape curve applied) Default i/e: MIG	Provided when shaping based upon age is desired. Note: Shape index 1 is reserved for constant 1.	Age-based shaping curve (SHAPE) to be applied to the fixed operating and maintenance cost.	*{See Related Parameters}*
NCAP_FSUB (r,datayear,p,cur)	OBJ_FSB, CST_FIXX, CST_PVP	Monetary unit per capacity unit [0,∞); default value: none Default i/e: STD	Provided when there is a subsidy for associated with the level of installed capacity.	Subsidy per unit of installed capacity.	Fixed subsidy associated with total installed capacity (VAR_NCAP+NCAP_PASTI) when entering the objective function component (EQ_OBJFIX) with a minus sign.
NCAP_FSUBM (r,datayear,p)	NCAP_FSUB, MULTI	Integer number Default value: 0 (no	Provided when shaping based upon the period is	Period sensitive multiplier curve	*{See Related Parameters}*

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		multiplier curve applied) Default i/e: MIG	desired. Note: Multiplier index 1 is reserved for constant 1.	(MULTI) applied to the subsidy (NCAP_FSUB).	
NCAP_FSUBX (r,datayear,p)	NCAP_FSUB, SHAPE	Integer number Default value: 0 (no shape curve applied) Default i/e: MIG	Provided when shaping based upon age is desired. Note: Shape index 1 is reserved for constant 1.	Age-based shaping curve (SHAPE) to be applied to the fixed subsidy (NCAP_FSUB).	*{See Related Parameters}*
NCAP_FTAX (r,datayear,p,cur)	OBJ_FTX, CST_FIXX, CST_PVP	monetary unit per capacity unit [open]; default value: none Default i/e: STD	Provided when there is a fixed tax based upon the level of the installed capacity.	Tax per unit of installed capacity.	Fixed subsidy associated with total installed capacity (VAR_NCAP+NCAP_PASTI) when entering the objective function components (EQ_OBJFIX).
NCAP_FTAXM (r,datayear,p)	NCAP_FTAX, MULTI	Integer number Default value: 0 (no multiplier curve applied) Default i/e: MIG	Provided when shaping based upon the period is desired. Note: Multiplier index 1 is reserved for constant 1.	Period sensitive multiplier curve (MULTI) applied to the tax (NCAP_FTAX).	*{See Related Parameters}*
NCAP_FTAXX (r,datayear,p)	NCAP_FTAX, SHAPE	Integer number Default value: 0 (no shape curve applied) Default i/e: MIG	Provided when shaping based upon age is desired. Note: Shape index 1 is reserved for constant 1.	Age-based shaping curve (SHAPE) to be applied to the fixed tax (NCAP_FTAX).	*{See Related Parameters}*
NCAP_ICOM (r,datayear,p,c)	NCAP_CLED, rpc_capflo, rpc_conly	Commodity unit per capacity unit [open]; default value: none Default i/e: STD	Provided when a commodity is needed in the period in which the new capacity is to be available, or before NCAP_CLED. If NCAP_CLED is provided, the commodity is required during the years [B(v)+NCAP_CLED,B(v)+NCAP_ILED-NCAP_CLED]. If	Amount of commodity (c) required for the construction of new capacity.	Applied to the investment variable (VAR_NCAP) in the appropriate commodity constraints (EQ(l)_COMBAL) as part of consumption.

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			this time spans more than one period, the commodity flow is split up proportion-ally between the periods. For the commodity balance the commodity requirement in a period is converted to an average annual commodity flow for the entire period, although the construction may take place only for a few years of the period. Negative value describes production (e.g. emissions) at the time of a new investment.		
NCAP_ILED (r,t,p)	NCAP_ICOM, NCAP_COST, COEF_CPT, COEF_ICOM, DUR_MAX	Years [open]; default value: none Default i/e: STD	Provided when there is a delay between when the investment decision occurs and when the capacity (new capacity or past investment) is initially available. If NCAP_ILED>0, the investment decision is assumed to occur at B(v) and the capacity becomes available at B(v)+NCAP-ILED. If NCAP_ILED<0, the investment decision is assumed to occur at B(v)-NCAP_ILED and the capacity becomes available at B(v). Causes an IDC overhead in the investment	Lead time between investment decision and actual availability of new capacity (= construction time).	Applied to the investment variable (VAR_NCAP) balance constraints (EQ(l)_COMBAL) as part of consumption, if there is an associated flow (NCAP_ICOM). Used as to distinguish between small and large investments (VAR_NCAP) and thus influences the way the investment and fixed costs are treated in the objective function (EQ_OBJINV, EQ_OBJFIX,

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			costs accounting.		EQ_OBJSALV).
NCAP_ISUB (r,datayear,p,cur)	OBJ_ISUB, OBJSCC, CST_INVX, CST_SALV, CST_PVP	monetary unit per capacity unit [0,∞); default value: none Default i/e: STD	Provided when there is a subsidy for new investments in a period.	Subsidy per unit of new installed capacity.	Applied to the investment variable (VAR_NCAP) when entering the objective function (EQ_OBJNV) with a minus sign. May appear in user constraints (EQ_UC*) if specified in UC_NAME.
NCAP_ITAX (r,datayear,p,cur)	OBJ_ITAX, OBJSCC, CST_INVX, CST_SALV, CST_PVP	monetary unit per capacity unit [0,∞); default value: none Default i/e: STD	Provided when there is a tax associated with new investments in a period.	Tax per unit of new installed capacity	Applied to the investment variable (VAR_NCAP) when entering the objective function (EQ_OBJNV). May appear in user constraints (EQ_UC*) if specified in UC_NAME.
NCAP_OCOM (r,datayear,p,c)	NCAP_VALU, rpc_capflo, rpc_conly	Commodity unit per capacity unit [open]; default value: none Default i/e: STD	Provided when there is a commodity release associated with the decommissioning. The year index of the parameter corresponds to the vintage year. If the decommissioning time (NCAP_DLIFE) falls in more than one period, is split up proportionally among the periods. For the commodity balance the commodity release in a period is converted to an average annual commodity flow for the entire period,	Amount of commodity c per unit of capacity released during the dismantling of a process.	Applied to the investment variable (VAR_NCAP) in the appropriate commodity constraints (EQ(l)_COMBAL) as part of production in the appropriate period.

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			although the dismantling may take place only for a few years of the period.		
NCAP_OLIFE (r,datayear,p)	NCAP_TLIFE	Years (0,∞); default value: none Default i/e: STD	Requires that early retirements are enabled and the process is vintaged.	Maximum operating lifetime of a process, in terms of full-load years.	EQL_SCAP
NCAP_PASTI (r,pastyear,p)	NCAP_PASTY, OBJ_PASTI, PAR_PASTI, PRC_RESID	capacity unit [0,∞); default value: none No i/e	Past investment can also be specified for milestone years, e.g. if the milestone year is a historic year, so that capacity additions are known or if planned future investments are already known.	Investment in new capacity made before the beginning of the model horizon (in the year specified by pastyear).	EQ(l)_COMBAL EQ_CPT EQ_OBJINV, EQ_OBJSALV, EQ_OBJFIX
NCAP_PASTY (r,pastyear,p)	NCAP_PASTI	Years [1,999]; default value: none No i/e	Provided to spread a single past investment (NCAP_PASTI) back over several years (e.g., cars in the period before the 1st milestoneyr were bought over the previous 15 years). If overlaps with other past investments, the capacity values are added.	Number of years to go back to calculate a linear build-up of past investments	*{See NCAP_PASTI}*
NCAP_PKCNT (r,datayear,p,s)	com_peak, com_pkts, prc_pkaf, prc_pkno	Decimal fraction [0,1]; default value: 1 Default i/e: STD	If the indicator PRC_PKAF is specified, the NCAP_PKCNT is set equal to the availabilities NCAP_AF. Direct inheritance. Weighted aggregation.	Fraction of capacity that can contribute to peaking equations.	Applied to investments in capacity (VAR_NCAP, NCAP_PASTI) in the peaking constraint (EQ_PEAK).
NCAP_SEMI (r,datayear,p)	NCAP_DISC	Capacity unit (0,∞);	Upper bound for the capacity must be defined	Semi-continuous new capacity, lower	Applied to the semi-continuous investment

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		default value: none Default i/e: MIG	by NCAP_BND; if not defined, assumed to be equal to the lower bound. Requires MIP.	bound. (See Section 5.9)	variable VAR_SNCAP in the discrete investment equation EQ_DSCNCAP
NCAP_START (r,p)	PRC_NOFF	Year [1000,∞); default value: none	NCAP_START(r,p)=y is equivalent to PRC_NOFF(r,p,BOH,y−1).	Start year for new investments	Affects the availability of investment variable (VAR_NCAP)
NCAP_TLIFE (r,datayear,p)	NCAP_ELIFE, COEF_CPT, COEF_RPTI, DUR_MAX	Years (0,∞); default value: G_TLIFE Default i/e: STD	Expected for all technologies that have investment costs. Values below 0.5 cannot be well accounted in the objective function, and should thus be avoided (they are automatically resetted to 1).	Technical lifetime of a process.	Impacts all calculations that are dependent upon the availability of investments (VAR_NCAP) including capacity transfer (EQ_CPT), commodity flow (EQ(l)_COMBAL), costs (EQ_OBJINV, EQ_OBJFIX, EQ_OBJVAR, EQ_OBJSALV).
NCAP_VALU (r,datayear,p,c,cur)	NCAP_OCOM	Monetary unit / commodity unit [0,∞); default value: none Default i/e: STD	Provided when a released commodity has a value.	Value of a commodity released at decommissioning (NCAP_OCOM).	Applied to the investment related (VAR_NCAP, NCAP_PASTI) release flow at decommissioning in the objective function (EQ_OBJSALV).
PRC_ACTFLO (r,datayear,p,cg)	PRC_CAPACT, prc_actunt, prc_spg, rpc_aire	Commodity unit / activity unit (0,∞); default value: 1 Default i/e: STD	Only (rarely) provided when either the activity and flow variables of a process are in different units, or if there is a conversion efficiency between the activity and the flow(s) in the PCG.	1) Conversion factor from units of activity to units of those flow variables that define the activity (primary commodity group), or, 2) Conversion	Applied to the primary commodity (prc_pcg) flow variables (VAR_FLO, VAR_IRE) to relate overall activity (VAR_ACT in EQ_ACTFLO). When the Reduction

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			The group (cg) can be the whole PCG or any individual commodity in the PCG, or 'ACT' (=PCG).	multiplier representing the amount of flow(s) in the cg per 1 unit of activity.	algorithm activated it is applied to the activity variable (VAR_ACT) in those cases where the flow variable (VAR_FLO) can be replaced by the activity variable (e.g. the activity is defined by one commodity flow).
PRC_CAPACT (r,p)	PRC_ACTFLO, PRC_ACTUNT	Activity unit / capacity unit (0,∞); default value: 1 Default i/e: none		Conversion factor from capacity unit to activity unit assuming that the capacity is used for one year.	Applied along with the availability factor (NCAP_AF) to the investment (VAR_NCAP + NCAP_PASTI) in the utilization equations (EQ(l)_CAPACT, EQ(l)_CAFLAC). Applied to the investment (VAR_NCAP + NCAP_PASTI) in the peak constraint (EQ_PEAK). Applied to the investment (VAR_NCAP + NCAP_PASTI) in the capacity utilization constraint for CHP plants (ECT_AFCHP) and peak constraint in the IER extension (see Part III).
PRC_MARK (r,datayear,p,item,c,bd)	FLO_MARK	Decimal fraction [open]; default value: none	Combined limit on commodity production is derived as the sum of the	Process group-wise market share, which defines a constraint	EQ(l)_FLOMRK VAR_COMPRD

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		Default i/e: 11	process-specific productions multiplied by the inverse values of PRC_MARK. The constraint is applied to the annual production of commodity.	for the combined market share of multiple processes in the total commodity production.	
PRC_RESID (r,datayear,p)	NCAP_PASTI	Capacity unit [0,∞); default value: none Default i/e: 1 (options 5/15 may be used for extrapolation over TLIFE)	If only a single data point is specified, linear decay of the specified residual capacity over technical lifetime is assumed. Used as an alternative to NCAP_PASTI, not to use both for the same process.	Residual existing capacity stock of process (p) still available in the year specified (datayear). PRC_RESID is most useful for describing the stock of capacity with mixed vintages, while NCAP_PASTI is suited for capacities of a certain vintages, such as an individual power plants.	EQ(l)_CAPACT EQ(l)_CAFLAC EQL_CAPFLO EQ(l)_CPT VAR_CAP
R_CUREX (r,cur1,cur2)	G_CUREX	Scalar (0,∞) Default value: none Default i/e: N/A	The target currency cur2 must have a discount rate defined with G_DRATE.	Conversion factor from currency cur1 to currency cur2 in region r, in order to use cur2 in the objective function.	Affects cost coefficients in EQ_OBJ
RCAP_BLK (r,datayear,p)	PRC_RCAP RCAP_BND	Capacity unit [0,∞); default value: none Default i/e: STD	Only effective when lumpy early capacity retirements are active (RETIRE=MIP). Requires MIP.	Retirement block size.	EQ_DSCRET VAR_DRCAP VAR_SCAP
RCAP_BND (r,datayear,p,bd)	PRC_RCAP RCAP_BLK	Capacity unit [0,∞); default value: none Default i/e: STD	Unless the control variable DSCAUTO=YES, requires that PRC_RCAP is defined for process p.	Bound on the retired amount of capacity in a period (same bound for all vintages).	VAR_RCAP VAR_SCAP
REG_BNDCST	REG_CUMCST	Monetary unit	The cost aggregations	Bound on regional	EQ_BNDCST

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
(r,datayear,agg,cur,bd)		[0,∞); default value: none Default i/e: MIG	(agg) supported are listed in the set COSTAGG (see Table 1).	costs by type of cost aggregation.	VAR_CUMCST
REG_CUMCST (r,y1,y2,agg,cur,bd)	REG_BNDCST	Monetary unit [0,∞); default value: none Default i/e: N/A	The cost aggregations (agg) supported are listed in the set COSTAGG (see Table 1).	Cumulative bound on regional costs by type of cost aggregation.	EQ_BNDCST VAR_CUMCST
REG_FIXT (all_r)		Year [1000,∞); default value: none	Only taken into account when the first periods are fixed by using the FIXBOH control variable.	Year up to which periods are fixed by period	–
RPT_OPT (item,j)		Integer value [open]; default value: none	See Part III, Table 15 for a list and descriptions of available options.	Miscellaneous reporting options	–
SHAPE (j,age)	FLO_FUNC, FLO_SUM, NCAP_AFX, NCAP_FOMX, NCAP_FSUBX, NCAP_FTAXX	Scalar [open]; default value: none I/e: Full dense interpolation and extrapolation	Provided for each age dependent shaping curve that is to be applied.	Multiplier table used for any shaping parameters (*_*X) to adjust the corresponding technical data as function of the age; the table can contain different multiplier curves that are identified by the index j.	*{See Related Parameters}*
STG_CHRG (r,datayear,p,s)	prc_nstts, prc_stgips, prc_stgtss	Scalar [0,∞); default value: none Default i/e: STD	Only applicable to storage processes (STG): timeslice storage, inter-period storage or night storage devices.	Annual exogenous charging of a storage technology in a particular timeslice s.	Exogenous charging of storage enters storage equations (EQ_STGTSS, EQ_STGIPS) as right-hand side constant.
STG_EFF (r,datayear,p)	prc_nstts, prc_stgips, prc_stgtss	Decimal fraction [0,∞); default value: 1 Default i/e: STD	Only applicable to storage processes (STG): timeslice storage, inter-period storage or night storage	Efficiency of storage process.	Applied to the storage output flow (VAR_SOUT) in the commodity balance

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			devices.		(EQ(l)_COMBAL) for the stored commodity.
STG_LOSS (r,datayear,p,s)	prc_nstts, prc_stgips, prc_stgtss	Scalar [open]; default value: none Default i/e: STD	Only applicable to storage processes (STG): timeslice storage, inter-period storage or night storage devices. STG_LOSS>0 defines the loss in proportion to the initial storage level during one year's storage time. STG_LOSS<0 defines an equilibrium loss, i.e. how much the annual losses would be if the storage level is kept constant.	Annual loss of a storage process per unit of average energy stored.	Timeslice storage process (EQ_STGTSS): applied to the average storage level (VAR_ACT) between two consecutive timeslices. Inter-period storage process (EQ_STGIPS): applied to the average storage level from the pre-period (VAR_ACT) and the net inflow (VAR_SIN-VAR_SOUT) of the current period.
STGIN_BND (r,datayear,p,c,s,bd)	prc_nstts, prc_stgips, prc_stgtss	Commodity unit [0,∞); default value: none Default i/e: MIG	Only applicable to storage processes (STG): timeslice storage, inter-period storage or night storage devices.	Bound on the input flow of a storage process in a timeslice s.	Storage input bound constraint (EQ(l)_STGIN) when s is above prc_tsl of the storage process. Direct bound on storage input flow (VAR_SIN) when at the prc_tsl level.
STGOUT_BND (r,datayear,p,c,s,bd)	prc_nstts, prc_stgips, prc_stgtss	Commodity unit [0,∞); default value: none Default i/e: MIG	Only applicable to storage processes (STG): timeslice storage, inter-period storage or night storage devices.	Bound on the output flow of a storage process in a timeslice s.	Storage output bound constraint (EQ(l)_STGIN) when s is above prc_tsl of the storage process. Direct bound on storage output flow variable (VAR_SOUT) when at the prc_tsl level.

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
TL_CCAP0 (r,teg)	(Alias: CCAP0) PAT, CCOST0	Capacity unit [open]; default value: none	Requires using ETL. For learning technologies teg when ETL is used.	Initial cumulative capacity of a learning technology.	Cumulative investment constraint (EQ_CUINV) and cumulative capacity variable (VAR_CCAP) in endogenous technological learning formulation.
TL_CCAPM (r,teg)	(Alias: CCAPM) CCOSTM	Capacity unit [open]; default value: none	Requires using ETL. For learning technologies teg when ETL is used.	Maximum cumulative capacity.	Core ETL equations.
TL_CLUSTER (r,teg,prc)	(Alias: CLUSTER) TL_MRCLUST	Decimal fraction. [0-1]; default value: none	Requires using ETL (MIP). • Provided to model clustered endogenous technology learning. • Each of the learning parameters must also be specified for the key learning technology.	Indicator that a technology (teg) is a learning component that is part of another technology (prc) in region r; teg is also called key component.	EQ_CLU
TL_MRCLUST (r,teg,reg,p)	TL_CLUSTER	Decimal fraction. [0-1]; default value: none	Requires using ETL (MIP). • Provided to model clustered endogenous technology learning. • Each of the learning parameters must also be specified for the key learning technology.	Mapping for multi-region clustering between learning key components (teg) and processes (p) that utilize the key component.	EQ_MRCLU
TL_PRAT (r,teg)	(Alias: PRAT) ALPH BETA CCAPK CCOST0 PAT PBT	Scalar [0,1]; default value none	Requires using ETL. Provided for learning technologies (teg) when ETL is used.	Progress ratio indicating the drop in the investment cost each time there is a doubling of the installed capacity.	Fundamental factor to describe the learning curve and thus effects nearly all equations and variables related to endogenous technology learning (ETL).
TL_SC0 (r,teg)	(Alias: SC0)	Monetary unit / capacity unit	Requires using ETL. For learning technologies	Initial specific investment costs.	Defines together with CCAP0 initial point of

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
		[open]; default value: none	teg when ETL is used.		learning curve and affects thus the core equations and variables of endogenous technological learning (ETL).
TL_SEG (r,teg)	(Alias: SEG)	Integer [open];	Requires using ETL. For learning technologies teg when ETL is used. Currently limited to six segments by set kp.	Number of segments.	Influences the piecewise linear approximation of the cumulative cost curve (EQ_COS, EQ_LA1, EQ_LA2).
UC_ACT (uc_n,side,r,datayear,p,s)	uc_n, uc_gmap_p	None [open]; default value: none Default: i/e: STD	Used in user constraints. Direct inheritance. Weighted aggregation.	Coefficient of the activity variable VAR_ACT in a user constraint.	EQ(l)_UCXXX
UC_CAP (uc_n,side,r,datayear,p)	uc_n, uc_gmap_p	None [open]; default value: none Default: i/e: STD	Used in user constraints.	Coefficient of the activity variable VAR_CAP in a user constraint.	EQ(l)_UCXXX
UC_CLI (uc_n,side,r,datayear,item)		Dimensionless [open]; default value: none Default i/e: STD	Used in user constraints. Climate variable can be at least any of CO2-GTC, CO2-ATM, CO2-UP, CO2-LO, FORCING, DELTA-ATM, DELTA-LO (for carbon). See Appendix on Climate Module for details.	Multiplier of climate variable in user constraint	EQ(l)_UCXXX
UC_COMCON (uc_n,side,r,datayear,c,s)	uc_n, uc_gmap_c	None [open]; default value: none Default: i/e: STD	Used in user constraints. No inheritance/aggregation (might be changed in the future).	Coefficient of the commodity consumption variable VAR_COMCON in a user constraint.	EQ(l)_UCXXX
UC_COMPRD (uc_n,side,r,datayear,c,	uc_n, uc_gmap_c	None [open];	Used in user constraints. No inheritance/aggregation	Coefficient of the net commodity	EQ(l)_UCXXX

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
s)		default value: none Default: i/e: STD	(might be changed in the future).	production variable VAR_COMPRD in a user constraint.	
UC_CUMACT (uc_n,r,p,y1,y2)	ACT_CUM	Dimensionless [open]; default value: none I/e: N/A	Used in cumulative user constraints only.	Multiplier of cumulative process activity variable in user constraint.	EQ(l)_UC EQ(l)_UCR VAR_CUMFLO
UC_CUMCOM (uc_n,r,type,c,y1,y2)	COM_CUMNET COM_CUMPRD	Dimensionless [open]; default value: none I/e: N/A	Used in cumulative user constraints only. Type=NET/PRD determines the variable referred to (CUMNET/ CUMPRD).	Multiplier of cumulative commodity variable in user constraint.	EQ(l)_UC EQ(l)_UCR VAR_CUMCOM
UC_CUMFLO (uc_n,r,p,c,y1,y2)	FLO_CUM	Dimensionless [open]; default value: none I/e: N/A	Used in cumulative user constraints only.	Multiplier of cumulative process flow variable in user constraint.	EQ(l)_UC EQ(l)_UCR VAR_CUMFLO
UC_FLO (uc_n,side,r,datayear,p,c,s)	uc_n	None [open]; default value: none Default: i/e: STD	Used in user constraints. Direct inheritance. Weighted aggregation.	Coefficient of the flow VAR_FLO variable in a user constraint.	EQ(l)_UCXXX
UC_IRE (uc_n,side,r,datayear,p,c,s)	uc_n	None [open]; default value: none Default: i/e: STD	Used in user constraints. Direct inheritance. Weighted aggregation.	Coefficient of the trade variable VAR_IRE in a user constraint.	EQ(l)_UCXXX
UC_NCAP (uc_n,side,r,datayear,p)	uc_n, uc_gmap_p	None [open]; default value: none Default: i/e: STD	Used in user constraints.	Coefficient of the activity variable VAR_NCAP in a user constraint.	EQ(l)_UCXXX
UC_RHS (uc_n,lim)	uc_n, uc_r_sum, uc_t_sum, uc_ts_sum	None [open]; default value: none Default i/e: none	Used in user constraints. Binding user constraints are defined using bound types lim=UP/LO/FX. Non-binding (free) user constraints can be defined	RHS constant with bound type of bd of a user constraint.	RHS (right-hand side) constant of a user constraint, which is summing over regions (uc_r_sum), periods (uc_t_sum) and

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			using the lim type lim=N.		timeslices (uc_ts_sum) (EQ(l)_UC).
UC_RHSR (r,uc_n,lim)	uc_n, uc_r_each, uc_t_sum, uc_ts_sum	None [open]; default value: none Default i/e: none	Used in user constraints. Binding user constraints are defined using bound types lim=UP/LO/FX. Non-binding (free) user constraints can be defined using the lim type lim=N.	RHS constant with bound type of bd of a user constraint.	RHS constant of user constraints, which are generated for each specified region (uc_r_each) and are summing over periods (uc_t_sum) and timeslices (uc_ts_sum) (EQ(l)_UCR).
UC_RHSRT (r,uc_n,datayear,lim)	uc_n, uc_r_each, uc_t_each, uc_t_succ, uc_ts_sum	None [open]; default value: none Default i/e: MIG	Used in user constraints. Binding user constraints are defined using bound types lim=UP/LO/FX. Non-binding (free) user constraints can be defined using the lim type lim=N.	RHS constant with bound type of bd of a user constraint.	RHS constant of user constraints, which are generated for each specified region (uc_r_each) and period (uc_t_each) and are summing over timeslices (uc_ts_sum) (EQ(l)_UCRT). If uc_t_succ instead of uc_t_each is specified the constraints will be generated as dynamic constraint between the two successive periods (EQ(l)_UCRSU).
UC_RHSRTS (r,uc_n,datayear,s,lim)	uc_n, uc_r_each, uc_t_each, uc_t_succ, uc_ts_each	None [open]; default value: none Default i/e: MIG	Used in user constraints. No inheritance / aggregation, unless the target timeslice level is specified by UC_TSL. Direct inheritance, if the target timeslice level is specified by UC_TSL.	RHS constant with bound type of bd of a user constraint.	RHS constant of user constraints, which are generated for each specified region (uc_r_each), period (uc_t_each) and timeslice (uc_ts_each) (EQ(l)_UCRTS).

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
			Binding user constraints are defined using bound types lim=UP/LO/FX. Non-binding (free) user constraints can be defined using the lim type lim=N.		If uc_t_succ instead of uc_t_each is specified the constraints will be generated as dynamic constraint between the two successive periods (EQ(l)_UCRSUS).
UC_RHST (uc_n,datayear,lim)	uc_n, uc_r_sum, uc_t_each, uc_t_succ, uc_ts_sum	None [open]; default value: none Default i/e: MIG	Used in user constraints. Binding user constraints are defined using bound types lim=UP/LO/FX. Non-binding (free) user constraints can be defined using the lim type lim=N.	RHS constant with bound type of bd of a user constraint.	RHS constant of user constraints, which are generated for each specified period (uc_t_each) and are summing over regions (uc_r_sum) and timeslices (uc_ts_sum) (EQ(l)_UCT). If uc_t_succ instead of uc_t_each is specified the constraints will be generated as dynamic constraint between the two successive periods (EQ(l)_UCSU).
UC_RHSTS (uc_n,datayear,s,lim)	uc_n, uc_r_sum, uc_t_each, uc_t_succ, uc_ts_each	None [open]; default value: none Default i/e: MIG	Used in user constraints. No inheritance/aggregation. Binding user constraints are defined using bound types lim=UP/LO/FX. Non-binding (free) user constraints can be defined using the lim type lim=N.	RHS constant with bound type of bd of a user constraint.	RHS constant of user constraints, which are generated for each specified period (uc_t_each) and timeslice (uc_ts_each) and are summing over regions (uc_r_sum) (EQ(l)_UCTS). If uc_t_succ instead of uc_t_each is specified the constraints will be

Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
					generated as dynamic constraint between the two successive periods (EQ(l)_UCSUS).
UC_TIME (uc_n,r,datayear)		Dimensionless [open]; default value: none Default i/e: STD	Used in user constraints. Adds a time constant to the RHS side.	Multiplier for the number of years in model periods (static UCs), or between milestone years (dynamic UCs)	EQ(l)_UCXXX
UC_UCN (uc_n,side,r,datayear, ucn)	UC_RHSRT	Dimensionless [open]; default value: none Default i/e: STD	Only taken into account if the user constraint is by region & period, and summing over timeslices and the RHS side is activated (EQ(l)_UCRSU).	Multiplier of user constraint variable in another user constraint.	EQ(l)_UCRSU VAR_UCRT
VDA_EMCB (r,datayear,c,com)	FLO_EMIS FLO_EFF	Emission units per flow units default value: none Default i/e: STD	Available in the VEDA shell. Any process-specific FLO_EMIS / FLO_EFF with the commodities c and com will override VDA_EMCB.	Emissions (com) from the combustion of commodity (c) in region (r).	EQ_PTRANS

3.2 Internal parameters

Table 14 gives an overview of internal parameters generated by the TIMES preprocessor. Similar to the description of the internal sets, not all internal parameters used within TIMES are discussed. The list given in Table 14 focuses mainly on the parameters used in the preparation and creation of the equations in Chapter 6. In addition to the internal parameters listed here, the TIMES preprocessor computes additional internal parameters which are either used only as auxiliary parameters being valid only in a short section of the code or which are introduced to improve the performance of the code regarding computational time.

Table 14: Internal parameters in TIMES

Internal parameter[31] (Indexes)	**Instances (Required / Omit / Special conditions)**	**Description**
ALPH (r,kp,teg)	For learning technologies teg when ETL is used.	Axis intercept on cumulative cost axis for description of linear equation valid for segment kp.
BETA (r,kp,teg)	For learning technologies teg when ETL is used.	Slope of cumulative cost curve in segment kp (= specific investment cost).
CCAPK (r,kp,teg)	For learning technologies teg when ETL is used.	Cumulative capacity at kinkpoint kp.
CCOST0(r,teg)	For learning technologies teg when ETL is used.	Initial cumulative cost of learning technology teg.
CCOSTK (r,kp,teg)	For learning technologies teg when ETL is used.	Cumulative investment cost at kinkpoint kp.
CCOSTM (r,teg)	For learning technologies teg when ETL is used.	Maximum cumulative cost based on CCAPM.
COEF_AF (r,v,t,p,s,bd)	For each technology, at the level of process operation (PRC_TSL).	Availability coefficient of the capacity (new investment variable VAR_NCAP plus still existing past investments NCAP_PASTI) in EQ(l)_CAPACT; COEF_AF is derived from the availability input parameters NCAP_AF, NCAP_AFA and NCAP_AFS taking into account any specified MULTI or SHAPE multipliers.

[31] The first row contains the parameter name, the second row contains in brackets the index domain, for which the parameter is defined.

Internal parameter[31] (Indexes)	Instances (Required / Omit / Special conditions)	Description
COEF_CPT (r,v,t,p)	For each technology the amount of an investment (VAR_NCAP) available in the period.	Fraction of capacity built in period v that is available in period t; might be smaller than 1 due to NCAP_ILED in vintage period or the fact that the lifetime ends within a period.
COEF_ICOM (r,v,t,p,c)	Whenever there is a commodity required during construction, the consuming being taken from the balance constraint (EQ(l)_COMBAL). Applied to the investment variable (VAR_NCAP) of period v in the commodity balance (EQ(l)_COMBAL) of period t. The duration during which the commodity is produced starts in the year B(v)+NCAP_ILED(v)–NCAP_CLED(v) and ends in the year B(v)+NCAP_ILED(v)–1.	Coefficient for commodity requirement during construction in period t due to investment decision in period v (see also NCAP_ICOM).
COEF_OCOM (r,v,t,p,c)	Whenever there is a commodity released during decommissioning, the production being added to the balance constraint (EQ(l)_COMBAL). Applied to the investment variable (VAR_NCAP) of period v in the commodity balance (EQ(l)_COMBAL) of period t. The release occurs during the decommissioning lifetime NCAP_DLIFE.	Coefficient for commodity release during decommissioning time in period t due to investment made in period v.
COEF_PTRAN (r,v,t,p,cg,c,com_grp)	For each flow through a process.	Coefficient of flow variable of commodity c belonging to commodity group cg in EQ_PTRANS equation between the commodity groups cg and com_grp.
COEF_PVT (r,t)	For each region, the present value of the time in each period.	Coefficient for the present value of periods, used primarily for undiscounting the solution marginals.

Internal parameter[31] (Indexes)	Instances (Required / Omit / Special conditions)	Description
COEF_RPTI (r,v,p)	For each technology whose technical life (NCAP_TLIFE) is shorter than the period.	Number of repeated investment of process p in period v when the technical lifetime minus the construction time is shorter than the period duration; Rounded to the next largest integer number.
COR_SALVD (r,v,p,cur)	For each technology existing past the end of the modelling horizon with decommissioning costs, adjustment in the objective function.	Correction factor for decommissioning costs taking into account technical discount rates and economic decommissioning times.
COR_SALVI (r,v,p,cur)	For each process extending past the end of the modelling horizon adjustment in the objective function.	Correction factor for investment costs taking into account technical discount rates, economic lifetimes and a user-defined discount shift (triggered by the control switch MIDYEAR (see Section 6.2 EQ_OBJ).
D (t)	For each period, D(t) = E(t)–B(t)+1.	Duration of period t.
DUR_MAX	For the model.	Maximum of NCAP_ILED + NCAP_TLIFE + NCAP_DLAG + NCAP_DLIFE + NCAP_DELIF over all regions, periods and processes.
M (v)	For each period, if the duration of the period is even, the middle year of the period is B(t) + D(t)/2 – 1, if the period is uneven, the middle year is B(t) + D(t)/2 – 0.5.	Middle year of period t.
MINYR	For the model	Minimum year over t = M(t) – D(t) +1; used in objective function.
MIYR_V1	For the model	First year of model horizon.
MIYR_VL	For the model	Last year of model horizon.
NTCHTEG (r,teg)	For learning technologies teg when ETL with technology clusters is used.	Number of processes using the same key technology teg.
OBJ_ACOST (r,y,p,cur)	For each process with activity costs. Enters the objective function (EQ_OBJVAR).	Inter-/Extrapolated variable costs (ACT_COST) for activity variable (VAR_ACT) for each year.

Internal parameter[31] (Indexes)	Instances (Required / Omit / Special conditions)	Description
OBJ_COMNT (r,y,c,s,type,cur)	For each commodity with costs, taxes or subsidies on the net production. Enters the objective function (EQ_OBJVAR).	Inter-/Extrapolated cost, tax and subsidy (distinguished by the type index) on net production of commodity (c) for each year associated with the variable VAR_COMNET. Cost types (type) are COST, TAX and SUB.
OBJ_COMPD (r,y,c,s,type,cur)	For each commodity with costs, taxes or subsidies on the commodity production. Enters the objective function (EQ_OBJVAR).	Inter-/Extrapolated cost, tax and subsidy (distinguished by the type index) on production of commodity (c) for each year associated with the variable VAR_COMPRD. Cost types (type) are COST, TAX and SUB.
OBJ_CRF (r,y,p,cur)	For each technology with investment costs. Enters objective function (EQ_OBJINV).	Capital recovery factor of investment in technology p in objective function taking into account the economic lifetime (NCAP_ELIFE) and the technology specific discount rate (NCAP_DRATE) or, if the latter is not specified, the general discount rate (G_DRATE).
OBJ_CRFD (r,y,p,cur)	For each technology with decommissioning costs. Enters objective function (EQ_OBJINV).	Capital recovery factor of decommissioning costs in technology p taking into account the economic lifetime (NCAP_DELIF) and the technology specific discount rate (NCAP_DRATE) or, if the latter is not specified, the general discount rate (G_DRATE).
OBJ_DCEOH (r,cur)	Enters objective function (EQ_OBJSALV).	Discount factor for the year EOH + 1 based on the general discount rate (G_DRATE).
OBJ_DCOST (r,y,p,cur)	For each technology with decommissioning costs. Enters objective function (EQ_OBJINV).	Inter-/Extrapolated decommissioning costs (NCAP_DCOST) for each year related to the investment (VAR_NCAP) of process p.
OBJ_DISC (r,y,cur)	Enters objective function (EQ_OBJINV, EQ_OBJVAR, EQ_OBJFIX, EQ_OBJSALV, EQ_OBJELS).	Annual discount factor based on the general discount rate (G_DRATE) to discount costs in the year y to the base year (G_DYEAR).
OBJ_DIVI (r,v,p)	Enters objective function (EQ_OBJINV).	Divisor for investment costs (period duration, technical lifetime or investment lead time depending on the investment cases 1a, 1b, 2a, 2b).

Internal parameter[31] (Indexes)	Instances (Required / Omit / Special conditions)	Description
OBJ_DIVIII (r,v,p)	Enters objective function (EQ_OBJINV).	Divisor for decommissioning costs and salvaging of decommissioning costs (period duration, technical lifetime or decommissioning time depending on the investment cases 1a, 1b, 2a, 2b).
OBJ_DIVIV (r,v,p)	Enters objective function (EQ_OBJFIX).	Divisor for fixed operating and maintenance costs and salvaging of investment costs.
OBJ_DLAGC (r,y,p,cur)	Enters objective function (EQ_OBJFIX).	Inter-/Extrapolated fixed capacity (VAR_NCAP+NCAP_PASTI) costs between the end of the technical lifetime and the beginning of the decommissioning for each year.
OBJ_FCOST (r,y,p,c,s,cur)	For each flow variable with flow related costs. Enters objective function (EQ_OBJVAR).	Inter-/Extrapolated flow costs (FLO_COST) for each year for the flow or trade variable (VAR_FLO, VAR_IRE) as well as capacity related flows (specified by NCAP_COM, NCP_ICOM, NCAP_OCOM).
OBJ_FDELV (r,y,p,c,s,cur)	For each flow with delivery costs. Enters objective function (EQ_OBJVAR).	Inter-/Extrapolated delivery costs (FLO_DELIV) for each year for the flow or trade variable (VAR_FLO, VAR_IRE) as well as capacity related flows (specified by NCAP_COM, NCP_ICOM, NCAP_OCOM).
OBJ_FOM (r,y,p,cur)	For each process with fixed operating and maintenance costs. Enters the objective function (EQ_OBJFIX).	Inter-/Extrapolated fixed operating and maintenance costs (NCAP_FOM) for the installed capacity (VAR_NCAP+NCAP_PASTI) for each year.
OBJ_FSB (r,y,p,cur)	For each process with subsidy on existing capacity. Enters objective function (EQ_OBJFIX).	Inter-/Extrapolated subsidy (NCAP_FSUB) on installed capacity (VAR_NCAP+NCAP_PASTI) for each year.
OBJ_FSUB (r,y,p,c,s,cur)	For each flow variable with subsidies. Enters objective function (EQ_OBJVAR).	Inter-/Extrapolated subsidy (FLO_SUB) for the flow or trade variable (VAR_FLO, VAR_IRE) for each year as well as capacity related flows (specified by NCAP_COM, NCP_ICOM, NCAP_OCOM).
OBJ_FTAX (r,y,p,c,s,cur)	For each flow variable with taxes. Enters objective function (EQ_OBJVAR).	Inter-/Extrapolated tax (FLO_TAX) for flow or trade variable (VAR_FLO, VAR_IRE) for each year as well as capacity related flows (specified by NCAP_COM, NCP_ICOM, NCAP_OCOM).

Internal parameter[31] (Indexes)	**Instances (Required / Omit / Special conditions)**	**Description**
OBJ_FTX (r,y,p,cur)	For each process with taxes on existing capacity. Enters objective function (EQ_OBJFIX).	Inter-/Extrapolated tax (NCAP_FTAX) on installed capacity (VAR_NCAP+NCAP_PASTI) for each year.
OBJ_ICOST (r,y,p,cur)	For each process with investment costs. Enters objective function (EQ_OBJINV).	Inter-/Extrapolated investment costs (NCAP_COST) for investment variable (VAR_NCAP) for each year.
OBJ_IPRIC (r,y,p,c,s,all_r,ie,cur)	For each import/export flow with prices assigned to it. Enters objective function (EQ_OBJVAR).	Inter-/Extrapolated import/export prices (IRE_PRICE) for import/export variable (VAR_IRE) for each year.
OBJ_ISUB (r,y,p,cur)	For each process with subsidy on new investment. Enters objective function (EQ_OBJINV).	Inter-/Extrapolated subsidy (NCAP_ISUB) on new capacity (VAR_NCAP) for each year.
OBJ_ITAX (r,y,p,cur)	For each process with taxes on new investment. Enters objective function (EQ_OBJINV).	Inter-/Extrapolated tax (NCAP_ITAX) on new capacity (VAR_NCAP) for each year.
OBJ_PASTI (r,v,p,cur)	Enters objective function (EQ_OBJINV).	Correction factor for past investments.
OBJ_PVT (r,t,cur)	Used as a multiplier in objective function in a few sparse cases.	Present value of time (in years) in period **t**, according to currency **cur** in region **r**, discounted to the base year.
OBJSIC (r,v,teg)	For learning technologies. Enters objective function (EQ_OBJINV).	Investment cost related salvage value of learning technology teg with vintage period v at year EOH+1.
OBJSSC (r,v,p,cur)	For processes with investment costs. Enters objective function (EQ_OBJSALV).	Investment cost related salvage value of process p with vintage period v at year EOH+1.
PAT (r,teg)	For learning technologies teg when ETL is used.	Learning curve coefficient in the relationship: SC = PAT * VAR_CCAP^(-PBT).

Internal parameter[31] (Indexes)	Instances (Required / Omit / Special conditions)	Description
PBT (r,teg)	For learning technologies teg when ETL is used.	Learning curve exponent PBT(r,teg) = LOG(PRAT(r,teg))/LOG(2).
PYR_V1	For the model	Minimum of pastyears and MINYR.
RS_FR (r,s,ts)	Defined for all commodities. Applied to flow variables in all equations in order to take into account cases where the variables may be defined at a different timeslice level than the level of the equation.	Fraction of timeslice s in timeslice ts, if s is below ts, otherwise 1. In other words, RS_FR(r,s,ts) = G_YRFR(r,s) / G_YRFR(r,ts), if s is below ts, and otherwise 1.
RS_STG (r,s)	Mainly applied for the modelling of storace cycles, but also in dispatching equations.	Lead from previous timeslice in the same cycle under the parent timeslice.
RS_STGAV (r,s)	Only applicable to storage processes (STG): timeslice storage devices, to calculate activity costs in proportion to the time the commodity is stored.	Average residence time of storage activity.
RS_STGPRD (r,s)	Only applicable to storage processes (STG): timeslice storage, inter-period storage or night storage devices.	Number of storage periods in a year for each timeslice.
RS_UCS (r,s,side)	Applied in timeslice-dynamic user constraints, to refer to the previous timeslice in the same cycle.	Lead from previous timeslice in the same cycle under the parent timeslice.
RTP_FFCX (r,v,t,p,cg,c,cg)	The efficiency parameter COEF_PTRAN is multiplied by the factor (1+RTP_FFCX). Enters EQ_PTRANS equation.	Average SHAPE multiplier of the parameter FLO_FUNC and FLO_SUM efficiencies in the EQ_PTRANS equation in the period (t) for capacity with vintage period (v). The SHAPE curve that should be used is specified by the user parameter FLO_FUNCX. The SHAPE feature allows to alter technical parameter given for the vintage period as a function of the age of the installation.

Internal parameter[31] (Indexes)	Instances (Required / Omit / Special conditions)	Description
RTCS_TSFR (r,t,c,s,ts)	Defined for each commodity with COM_FR. Applied to flow variables in all equations in order to take into account cases where some of the variables may be defined at a different timeslice level than the level of the equation.	The effective handling of timeslice aggregation/disaggregation. If ts is below s in the timeslice tree, the value is 1, if s is below ts the value is COM_FR(r,s) / COM_FR(r,ts) for demand commodities with COM_FR given and G_YRFR(r,s) / G_YRFR(r,ts) for all other commodities. The parameter is used to match the timeslice resolution of flow variables (VAR_FLO/VAR_IRE) and commodities. RTCS_TSFR is the coefficient of the flow variable, which is producing or consuming commodity c, in the commodity balance of c. If timeslice s corresponds to the commodity timeslice resolution of c and timeslice ts to the timeslice resolution of the flow variable two cases may occur: The flow variables are on a finer timeslice level than the commodity balance: in this case the flow variables with timeslices s being below ts in the timeslice tree are summed to give the aggregated flow within timeslice ts. RTCS_TSFR has the value 1. The flow variables are on coarser timeslice level than the commodity balance: in this case the flow variable is split-up on the finer timeslice level of the commodity balance according to the ratio of the timeslice duration of s to ts: RTCS_TSFR has the value = COM_FR(r,s) / COM_FR(r,s1) for demand commodities and G_YRFR(r,s) / G_YRFR(r,s1) otherwise. When COM_FR is used, the demand load curve is moved to the demand process. Thus, it is possible to model demand processes on an ANNUAL level and ensure at the same time that the process follows the given load curve COM_FR.
SALV_DEC (r,v,p,k,ll)	For those technologies with salvage costs incurred after the model horizon the contribution to the objective function.	Salvage proportion of decommissioning costs made at period v with commissioning year k.
SALV_INV (r,v,p,k)	For those technologies with salvage costs incurred after the model horizon the contribution to the objective function.	Salvage proportion of investment made at period v with commissioning year k.
YEARVAL (y)	A value for each year.	Numerical value of year index (e.g. YEARVAL('1984') equals 1984).

3.3 Report parameters

3.3.1 Overview of report parameters

The parameters generated internally by TIMES to document the results of a model run are listed in Table 15. These parameters can be imported into the **VEDA-BE** tool for further result analysis. They are converted out of the **GDX**[32] file via the **gdx2veda** GAMS utility into a **VEDA-BE** compatible format according to the file **times2veda.vdd**[33]. Note that some of the results are not transferred into parameters, but are directly accessed through the **times2veda.vdd** file (levels of commodity balances and peaking equation, total discounted value of objective function). The following naming conventions apply to the prefixes of the report parameters:

- CST_: detailed annual undiscounted cost parameters; note that also the costs of past investments, which are constants in the objective function, are being reported;
- PAR_: various primal and dual solution parameters;
- EQ(l)_: directly accessed GAMS equation levels/marginals
- REG_: regional total cost indicators.

Table 15: Report parameters in TIMES

Report parameter[34] (Indexes)	VEDA-BE attribute name	Description
AGG_OUT (r,t,c,s)	VAR_FOut	Commodity production by an aggregation process: Production of commodity (c) in period (t) and timeslice (s) from other commodities aggregated into c.
CAP_NEW (r,v,p,t,uc_n)	Cap_New	Newly installed capacity and lumpsum investment by vintage and commissioning period: New capacity and lumpsum investment of process (p) of vintage (v) commissioned in period (t).
CM_RESULT (c,t)	VAR_Climate	Climate module results for the levels of climate variable (c) in period (t).

[32] GDX stands for GAMS Data Exchange. A GDX file is a binary file that stores the values of one or more GAMS symbols such as sets, parameters variables and equations. GDX files can be used to prepare data for a GAMS model, present results of a GAMS model, store results of the same model using different parameters etc. They do not store a model formulation or executable statements.

[33] The use of the **gdx2veda** tool together with the **times2veda.vdd** control file and the **VEDA-BE** software are described in Part V.

[34] First row: parameter name; second row (in brackets): the index domain, for which the parameter is defined.

Report parameter[34] (Indexes)	VEDA-BE attribute name	Description
CM_MAXC_M (c,t)	Dual_Clic	Climate module results for the duals of constraint related to climate variable (c) in period (t).
CST_ACTC (r,v,t,p,uc_n)	Cost_Act	Annual activity costs: Annual undiscounted variable costs (caused by ACT_COST) in period (t) associated with the operation (activity) of a process (p) with vintage period (v). Additional indicator (uc_n) for start-up costs.
CST_COMC (r,t,c)	Cost_Com	Annual commodity costs: Annual undiscounted costs for commodity (c) (caused by COM_CSTNET and COM_CSTPRD) in period (t).
CST_COME (r,t,c)	Cost_Els	Annual elastic demand cost term: Annual costs (losses) due to elastic demand changes of commodity (c). When elastic demands are used the objective function describes the total surplus of producers and consumers, which reaches its maximum in the equilibrium of demand and supply.
CST_COMX (r,t,c)	Cost_Comx	Annual commodity taxes/subsidies: Annual undiscounted taxes and subsidies for commodity (c) (caused by COM_TAXNET, COM_SUBNET, COM_TAXPRD, COM_SUBPRD) in period (t).
CST_DAM (r,t,c)	Cost_Dam	Annual damage cost term: Annual undiscounted commodity (c) related costs, caused by DAM_COST, in period (t).
CST_DECC (r,v,t,p)	Cost_Dec	Annual decommissioning costs: Annual undiscounted decommissioning costs (caused by NCAP_DCOST and NCAP_DLAGC) in period (t), associated with the dismantling of process (p) with vintage period (v).
CST_FIXC (r,v,t,p)	Cost_Fom	Annual fixed operating and maintenance costs: Annual undiscounted fixed operating and maintenance costs (caused by NCAP_FOM) in period (t) associated with the installed capacity of process (p) with vintage period (v).
CST_FIXX (r,v,t,p)	Cost_Fixx	Annual fixed taxes/subsidies: Annual undiscounted fixed operating and maintenance costs (caused by NCAP_FTAX, NCAP_FSUB) in period (t) associated with the installed capacity of process (p) with vintage period (v).
CST_FLOC (r,v,t,p,c)	Cost_Flo	Annual flow costs (including import/export prices): Annual undiscounted flow related costs (caused by FLO_COST, FLO_DELV, IRE_PRICE) in period (t) associated with a commodity (c) flow in/out of a process (p) with vintage period (v) as well as capacity related commodity flows (specified by NCAP_COM, NCAP_ICOM, NCAP_OCOM).
CST_FLOX (r,v,t,p,c)	Cost_Flox	Annual flow taxes/subsidies: Annual undiscounted flow related costs (caused by FLO_TAX, FLO_SUB) in period (t) associated with a commodity (c) flow in/out of a process (p) with vintage period (v) as well as capacity related commodity flows (specified by NCAP_COM, NCAP_ICOM, NCAP_OCOM).
CST_INVC (r,v,t,p,uc_n)	Cost_Inv	Annual investment costs: Annual undiscounted investment costs (caused by NCAP_COST) in period (t) spread over the economic

Report parameter[34] (Indexes)	VEDA-BE attribute name	Description
		lifetime (NCAP_ELIFE) of a process (p) with vintage period (v).
CST_INVX (r,v,t,p,uc_n)	Cost_Invx	Annual investment taxes/subsidies: Annual undiscounted investment costs (caused by NCAP_ITAX, NCAP_ISUB) in period (t) spread over the economic lifetime (NCAP_ELIFE) of a process (p) with vintage period (v).
CST_IREC (r,v,t,p,c)	Cost_ire	Annual implied costs of endogenous trade: Annual undiscounted costs from endogenous imports/exports of commodity (c) in period (t) associated with process (p) and vintage period (v), valued according to the marginal(s) of the trade equation of process p.
CST_PVC (uc_n,r,c)	Cost_NPV	Total discounted costs by commodity (optional): Total present value of commodity-related costs in the base year, by type (with types COM, ELS, DAM). See Part III, Section 3.10 on the reporting options, and Table 16 below for acronym explanations.
CST_PVP (uc_n,r,p)	Cost_NPV	Total discounted costs by process (optional): Total present value of process-related costs in the base year, by type (with types INV, INV+, FIX, ACT, FLO, IRE, where INV+ is only used for the split according to hurdle rate). See Part III, Section 3.10 on the reporting options, and Table 16 below for acronym explanations.
CST_SALV (r,v,p)	Cost_Salv	Salvage values of capacities at EOH+1: Salvage value of investment cost, taxes and subsidies of process (p) with vintage period (v), for which the technical lifetime exceeds the end of the model horizon, value at year EOH+1.
CST_TIME (r,t,s,uc_n)	Time_NPV	Discounted value of time by period: Present value of the time in each model period (t) by region (r), with s='ANNUAL' and uc_n='COST'/'LEVCOST' depending on whether the $SET ANNCOST LEV reporting option has been used.
EQ_PEAK.L (r,t,c,s)	EQ_Peak	Peaking Constraint Slack: Level of the peaking equation (EQ_PEAK) of commodity (c) in period (t) and timeslice (s).
EQE_COMBAL.L (r,t,c,s)	EQ_Combal	Commodity Slack/Levels: Level of the commodity balance equation (EQE_COMBAL) of commodity (c) in period (t) and timeslice (s), where the equation is a strict equality.
EQG_COMBAL.L (r,t,c,s)	EQ_Combal	Commodity Slack/Levels: Level of the commodity balance equation (EQG_COMBAL) of commodity (c) in period (t) and timeslice (s), where the equation is an inequality.
F_IN (r,v,t,p,c,s)	VAR_FIn	Commodity Consumption by Process: Input flow (consumption) of commodity (c) in period (t) and timeslice (s) into process (p) with vintage period (v), including exchange processes.
F_OUT (r,v,t,p,c,s)	VAR_FOut	Commodity Production by Process: Output flow (production) of commodity (c) in period (t) and timeslice (s) into process (p) with vintage period (v), including exchange processes.
OBJZ.L	ObjZ	Total discounted system cost:

Report parameter[34] (Indexes)	VEDA-BE attribute name	Description
()		Level of the ObjZ variable, equal to the value of the objective function.
PAR_ACTL (r,v,t,p,s)	VAR_Act	Process Activity: Level value of activity variable (VAR_ACT) in period (t), timeslice (s) of process (p) in vintage period (v).
PAR_ACTM (r,v,t,p,s)	VAR_ActM	Process Activity – Marginals: Undiscounted annual reduced costs of activity variable (VAR_ACT) in period (t) and timeslice (s) of process (p) with vintage period (v); when the variable is at its lower (upper) bound, the reduced cost describes the increase (decrease) in the objective function caused by an increase of the lower (upper) bound by one unit; the reduced cost can also be interpreted as the necessary decrease or increase of the cost coefficient of the activity variable in the objective function, for the activity variable to leave its lower (upper) bound.
PAR_CAPL (r,t,p)	VAR_Cap	Technology Capacity: Capacity of process (p) in period (t), derived from VAR_NCAP in previous periods summed over all vintage periods. For still existing past investments, see PAR_PASTI.
PAR_CAPLO (r,t,p)	PAR_CapLO	Capacity Lower Limit: Lower bound on capacity variable (CAP_BND('LO')), only reported, if the lower bound is greater than zero.
PAR_CAPM (r,t,p)	VAR_CapM	Technology Capacity – Marginals: Undiscounted reduced costs of capacity variable (VAR_CAP); only reported in those cases, in which the capacity variable is generated (bound CAP_BND specified or endogenous technology learning is used); the reduced costs describe in the case, that the capacity variable is at its lower (upper) bound, the cost increase (decrease) of the objective function caused by an increase of the lower (upper) bound by one unit. The reduced cost is undiscounted with COEF_PVT.
PAR_CAPUP (r,t,p)	PAR_CapUP	Capacity Upper Limit: Upper bound on capacity variable (CAP_BND('UP')), only reported, if upper bound is smaller than infinity.
PAR_COMBALEM (r,t,c,s)	EQ_CombalM	Commodity Slack/Levels – Marginals: Undiscounted annual shadow price of commodity balance (EQE_COMBAL) being a strict equality. The marginal value describes the cost increase in the objective function, if the difference between production and consumption is increased by one unit. The marginal value can be determined by the production side (increasing production), but can also be set by the demand side (e.g., decrease of consumption by energy saving or substitution measures).
PAR_COMBALGM (r,t,c,s)	EQ_CombalM	Commodity Slack/Levels – Marginals: Undiscounted annual shadow price of commodity balance (EQG_COMBAL) being an inequality (production being greater than or equal to consumption); positive number, if production equals consumption; the marginal value describes the cost increase in the objective function, if the difference between production and consumption is increased by one unit. The marginal value can be determined by the production side (increasing production), but can also be set by the demand side (e.g., decrease of consumption by energy saving or substitution measures).

Report parameter[34] (Indexes)	**VEDA-BE attribute name**	**Description**
PAR_COMNETL (r,t,c,s)	VAR_Comnet	Commodity Net: Level value of the variable corresponding the net level of a commodity (c) (VAR_COMNET). The net level of a commodity is equivalent to the total production minus total consumption of said commodity. It is only reported, if a bound or cost is specified for it or it is used in a user constraint.
PAR_COMNETM (r,t,c,s)	VAR_ComnetM	Commodity Net – Marginal: Undiscounted annual reduced costs of the VAR_COMNET variable of commodity (c). It is only reported, if a bound or cost is specified for it or it is used in a user constraint.
PAR_COMPRDL (r,t,c,s)	VAR_Comprd	Commodity Total Production: Level value of the commodity production variable (VAR_COMPRD). The variable represents the total production of a commodity. It is only reported, if a bound or cost is specified for it or it is used in a user constraint.
PAR_COMPRDM (r,t,c,s)	VAR_ComprdM	Commodity Total Production – Marginal: Undiscounted annual reduced costs of the commodity production variable (VAR_COMPRD). It is only reported, if a bound or cost is specified for it or it is used in a user constraint.
PAR_CUMCST (r,v,t,uc_n,c)	VAR_CumCst	Cumulative costs by type (if constrained); Level of cumulative constraint for costs of type (uc_n) and currency (c) in region (r).
PAR_CUMFLOL (r,p,c,v,t)	EQ_Cumflo	Cumulative flow constraint – Levels: Level of cumulative constraint for flow of commodity (c) of process (p) between the year range (v–t).
PAR_CUMFLOM (r,p,c,v,t)	EQ_CumfloM	Cumulative flow constraint – Marginals: Shadow price of cumulative constraint for flow of commodity (c) of process (p) between the year range (v–t). Not undiscounted.
PAR_EOUT (r,v,t,p,c)	VAR_Eout	Electricity supply by technology and energy source (optional): Electricity output of electricity supply processes by energy source; based on using NRG_TMAP to identify electricity commodities, but excludes standard and storage processes having electricity as input. (Opted out by default – set RPT_OPT('FLO','5')=1 to activate; see Part III, Section 3.10).
PAR_FLO (r,v,t,p,c,s)	see: F_IN/F_OUT	Flow of commodity (c) entering or leaving process (p) with vintage period (v) in period (t).
PAR_FLO (r,v,t,p,c,s)	none	Discounted reduced costs of flow variable of commodity (c) in period (t) of process (p) with vintage period (v); the reduced costs describe that the flow variable is at its lower (upper) bound, and give the cost increase (decrease) of the objective function caused by an increase of the lower (upper) bound by one unit; the undiscounted reduced costs can be interpreted as the necessary decrease / increase of the cost coefficient of the flow variable, such that the flow will leave its lower (upper) bound.
PAR_IRE (r,v,t,p,c,s,ie)	see: F_IN/F_OUT	Inter-regional exchange flow of commodity (c) in period (t) via exchange process (p) entering region (r) as import (ie='IMP') or leaving region (r) as export (ie='EXP').
PAR_IREM	none	Discounted reduced costs of inter-regional exchange flow variable of commodity (c) in period (t) of

Report parameter[34] (Indexes)	VEDA-BE attribute name	Description
(r,v,t,p,c,s,ie)		exchange process (p) with vintage period (v); the reduced costs describe that the flow variable is at its lower (upper) bound, and give the cost increase (or decrease) of the objective function caused by an increase of the lower (upper bound) by one unit; the undiscounted reduced costs can be interpreted as the necessary decrease / increase of the cost coefficient of the flow variable in the objective function, such that the flow will leave its lower (upper) bound.
PAR_IPRIC (r,t,p,c,s,uc_n)	EQ_IreM	Inter-regional trade equations – Marginals: Undiscounted shadow price of the inter-regional trade equation of commodity (c) via exchange process (p) in period (t) and timeslice (s). The undiscounted shadow price can be interpreted as the import/export price of the traded commodity. Note: ucn={IMP/EXP}.
PAR_NCAPL (r,t,p)	VAR_Ncap	Technology Investment – New capacity: Level value of investment variable (VAR_NCAP) of process (p) in period (v).
PAR_NCAPM (r,t,p)	VAR_NcapM	Technology Investment – Marginals: Undiscounted reduced costs of investment variable (VAR_NCAP) of process (p); only reported, when the capacity variable is at its lower or upper bound; the reduced costs describe in the case, that the investment variable is at its lower (upper) bound, the cost increase (decrease) of the objective function caused by an increase of the lower (upper) bound by one unit; the undiscounted reduced costs can be interpreted as the necessary decrease / increase in the investment cost coefficient, such that the investment variable will leave its lower (upper) bound.
PAR_NCAPR (r,t,p,uc_n)	VAR_NcapR	Technology Investment – BenCost + ObjRange (see Part III, Section 3.10 for more details): Cost-benefit and ranging indicators for process (p) in period (t), where uc_n is the name of the indicator: • COST - the total unit costs of VAR_NCAP (in terms of an equivalent investment cost) • CGAP - competitiveness gap (in terms of investment costs), obtained directly from the VAR_NCAP marginals (and optional ranging information) • GGAP - competitiveness gap (in terms of investment costs), obtained by checking also the VAR_ACT, VAR_FLO and VAR_CAP marginals, in case VAR_NCAP is basic at zero • RATIO - benefit / cost ratio, based on CGAP • GRATIO - benefit / cost ratio, based on GGAP • RNGLO - ranging information (LO) for VAR_NCAP (if ranging is activated; in terms of investment costs) • RNGUP - ranging information (UP) for VAR_NCAP (if ranging is activated; in terms of investment costs)
PAR_PASTI (r,t,p,v)	VAR_Cap	Technology Capacity: Residual capacity of past investments (NCAP_PASTI) of process (p) still existing in period (t), where vintage (v) is set to '0' to distinguish residual capacity from new capacity.
PAR_PEAKM (r,t,c,s)	EQ_PeakM	Peaking Constraint Slack – Marginals: Undiscounted annual shadow price of peaking equation (EQ_PEAK) associated with commodity (c); since the peaking equation is at most only binding for one timeslice (s), a shadow price only exists for one timeslice. The shadow price can be interpreted as an additional premium to the shadow price of the

Report parameter[34] (Indexes)	VEDA-BE attribute name	Description
		commodity balance that consumers of commodity (c) have to pay for consumption during peak times. The premium is used (besides other sources) to cover the capacity related costs (e.g., investment costs) of capacity contributing reserve capacity during peak times.
PAR_TOP (r,t,p,c,uc_n)	PAR_Top	Process topology: Process topology indicators for reporting use. Values are all zero, period (t) is the first milestone year, and uc_n = IN/OUT. (Opted out by default – SET RPT_TOP YES to activate.)
PAR_UCMRK (r,t,uc_n,c,s)	User_conFXM	Marginal cost of user constraint: Undiscounted shadow price of group-wise market share constraint (defined with PRC_MARK) for commodity c, identified with name uc_n, in period t and timeslice s.
PAR_UCRTP (uc_n,r,t,p,c)	User_DynbM	Marginal cost of dynamic process bound constraint: Undiscounted shadow price of dynamic process-wise bound constraint, identified with name uc_n, for variable c (CAP / NCAP / ACT), in period t and timeslice s.
PAR_UCSL (uc_n,r,t,s)	User_con	Level of user constraint (or its slack) (only reported when the VAR_UC variables are used): The level of user constraint (uc_n) by region (r), period (t) and timeslice (s). The levels should be zero whenever the RHS constant is zero and the equation is binding. If the constraint is not binding, the level together with the RHS constant gives the gap for the equation to become binding.
PAR_UCSM (uc_n,r,t,s)	User_conFXM	Marginal cost of fixed bound user constraint Marginal of user constraint (uc_n) by region (r), period (t) and timeslice (s). The marginals are undiscounted, if the constraint is defined by region and period. The marginals of cumulative and multi-region user constraints are thus not undiscounted, due to ambiguity.
REG_ACOST (r,t,uc_n)	Reg_ACost	Regional total annualized costs by period: Total annualized costs in region (r) by period (t) and cost category. The cost categories are INV, INVX, FIX, FIXX, VAR, VARX, IRE, ELS and DAM (see Table 16 below for more information).
REG_IREC (r)	Reg_irec	Regional total discounted implied trade cost: Total discounted implied trade costs in region (r), derived by multiplying the shadow prices of the trade equations by the trade volumes. The sum of REG_IREC over regions is zero.
REG_OBJ (r)	Reg_obj	Regional total discounted system cost: Discounted objective value (EQ_OBJ) for each region (r).
REG_WOBJ (r,uc_n,c)	Reg_wobj	Regional total discounted system cost by component: Discounted objective value (EQ_OBJ) for each region (r), by cost type (uc_n) and currency (c). The cost types are: INV, INVX, FIX, FIXX, VAR, VARX, ELS, DAM (see Table 16 below for more information).
VAL_FLO (r,v,t,p,c)	Val_Flo	Annual commodity flow values: Flows of process (p) multiplied by the commodity balance marginals of those commodities (c) in period (t); the values can be interpreted as the market values of the process inputs and outputs.

3.3.2 Acronyms used in cost reporting parameters

The acronyms used in the reporting parameters for referring to certain types of costs are summarized in Table 16. The acronyms are used as qualifiers in the **uc_n** index of each reporting attribute, and are accessible in VEDA-BE through that same dimension.

Table 16: Acronyms used in the cost reporting parameters.

Cost parameter	Component acronyms	
CST_PVC (uc_n,r,c)	Total discounted costs by commodity (optional):	
	COM	Commodity-related costs, taxes and subsidies
	ELS	Losses in elastic demands
	DAM	Damage costs
CST_PVP (uc_n,r,p)	Total discounted costs by process (optional):	
	INV	Investment costs, taxes and subsidies, excluding portions attributable to hurdle rates in excess of the general discount rate
	INV+	Investment costs, taxes and subsidies, portions attributable to hurdle rates in excess of the general discount rate
	FIX	Fixed costs, taxes and subsidies
	ACT	Activity costs
	FLO	Flows costs taxes and subsidies (including exogenous IRE prices)
	IRE	Implied trade costs minus revenues
REG_ACOST (r,t,uc_n)	Regional total annualized costs by period:	
	INV	Annualized investment costs
	INVX	Annualized investment taxes and subsidies
	FIX	Annual fixed costs
	FIXX	Annual fixed taxes and subsidies
	VAR	Annual variable costs
	VARX	Annual variable taxes and subsidies
	IRE	Annual implied trade costs minus revenues
	ELS	Annual losses in elastic demands
	DAM	Annual damage costs
REG_WOBJ (r,uc_n,c)	Regional total discounted system cost by component:	
	INV	Investment costs
	INVX	Investment taxes and subsidies
	FIX	Fixed costs
	FIXX	Fixed taxes and subsidies
	VAR	Variable costs
	VARX	Variable taxes and subsidies
	ELS	Losses in elastic demands
	DAM	Damage costs

3.3.3 The levelized cost reporting option

As indicated in Table 15 above, the reporting of levelized costs for each process can be requested by setting the option RPT_OPT('NCAP', '1'). The results are stored in the VEDA-BE **Var_NcapR** result attribute, with the qualifier 'LEVCOST' (with a possible system label prefix).

The levelized cost calculation option looks to weight all the costs influencing the choice of a technology by TIMES. It takes into consideration investment, operating, fuel, and other costs as a means of comparing the full cost associated with each technology.

Levelized cost can be calculated according to the following general formula:

$$LEC = \frac{\sum_{t=1}^{n} \frac{IC_t}{(1+r)^{t-1}} + \frac{OC_t + VC_t + \sum_i FC_{i,t} + FD_{i,t} + \sum_j ED_{j,t}}{(1+r)^{t-0.5}} - \frac{\sum_k BD_{k,t}}{(1+r)^{t-0.5}}}{\sum_{t=1}^{n} \frac{\sum_m MO_{m,t}}{(1+r)^{t-0.5}}} \qquad (1)$$

where

- r = discount rate (e.g. 5%)
- IC_t = investment expenditure in (the beginning of) year t
- OC_t = fixed operating expenditure in year t
- VC_t = variable operating expenditure in year t
- FC_{it} = fuel-specific operating expenditure for fuel i in year t
- FD_{it} = fuel-specific acquisition expenditure for fuel i in year t
- ED_{jt} = emission-specific allowance expenditure for emission j in year t (optional)
- BD_{kt} = revenues from by-product k in year t (optional; see below)
- MO_{mt} = output of main product m in year t

The exponent *t–0.5* in the formula indicates the good practice of using mid-year discounting for continuous streams of annual expenditures.

In TIMES, the specific investment, fixed and variable O&M costs and fuel-specific flow costs are calculated directly from the input data. However, for the fuel acquisition prices, emission prices and by-product prices, ***commodity marginals*** from the model solution are used. All the unit costs are multiplied by the corresponding ***variable levels*** as given by the model solution: investment cost and fixed operating costs are multiplied by the amounts of capacity installed / existing, variable operation costs by the activity levels, and fuel-specific costs by the process flow levels. Mid-year discounting can also be activated.

The outputs of the main products are taken from the flow levels of the commodities in the primary group (PG) of the process. An exception is CHP processes, for which the electricity output is considered the sole main output, and heat is considered as a by-product.

Options for variants of levelized cost reporting:

1. <u>Do not include emission prices or by-product revenues in the calculation</u> (RPT_OPT('NCAP','1') = –1):

   In this option emission prices are omitted from the calculation, in accordance with the most commonly used convention for LEC calculation. Consequently, any by-product revenues need to be omitted as well, because if emissions have prices, the by-product prices in the solution would of course be polluted by those prices, and thus it would be inconsistent to use them in the calculation. Instead, in this case any amount of by-product energy produced by ELE, CHP and HPL processes is indirectly credited by reducing the fuel-specific costs in the calculation to the fraction of the main output in the total amount of energy produced.

2. Include both emission prices and by-product revenues in the calculation (RPT_OPT('NCAP','1') = 1):

   In this option both emission prices and by-product revenues are included in the calculation. The levelized cost thus represents the unit cost after subtracting the levelized value of all by-products from the gross value of the levelized cost. This approach of crediting for by-products in the LEC calculation has been utilized, for example, in the IEA *Projected Costs of Generating Electricity* studies.

3. Include not only emission prices and by-product revenues, but also the revenues from the main product in the calculation (RPT_OPT('NCAP','1') = 2):

   This option is similar to option (2) above, but in this case all product revenues are included in the calculation, including also the peak capacity credit from the TIMES peaking equation (when defined). The calculated LEC value thus represents the levelized **net** unit cost after subtracting the value of all products from the gross levelized cost. For competitive new capacity vintages, the resulting levelized cost should in this case generally be *negative*, because investments into technologies that enter the solution are normally profitable. For the marginal technologies the levelized cost can be expected to be very close to zero. Only those technologies that have been in some way forced into the solution, e.g. by specifying lower bounds on the capacity or by some other types of constraints, should normally have a positive levelized cost when using this option.

In the TIMES calculation, the expenditures for technology investments and process commodity flows include also taxes minus subsidies, if such have been specified. The levelized costs are calculated by process vintage, but only for new capacity vintages, as for them both the full cost data influencing technology choice and the operating history starting from the commissioning date are available, which is rarely the case for existing vintages.

4 Usage notes on special types of processes

4.1 Combined heat and power

4.1.1 Overview

Cogeneration power plants or combined heat and power plants (CHP) are plants that consume one or more commodities and produce two commodities, electricity and heat. One can distinguish two different types of cogeneration power plants according to the flexibility of the outputs, a back-pressure turbine process and a pass-out turbine process.

Back-pressure turbines are systems where the ratio of heat production to electricity production is fixed, and the electricity generation is therefore directly proportional to the heat generation. Pass-out turbines are systems where the ratio of heat production to electricity production is flexible, usually having a minimum value of zero and a maximum value usually in the range of 0.8–3 (but can be even smaller or larger).

However, both types of CHP systems often additionally support so-called reduction operation, where the turbine can be by-passed, whereby all the steam is directed to a heat exchanger for producing heat. As a result, in a back-pressure turbine system, the ratio of heat production to electricity production may in such systems vary from the fixed value to infinity, and in a pass-out turbine system it may vary from zero to infinity.

All these different cases are illustrated in Figure 10 below, which shows the relations between heat and electricity production in different modes of a flexible CHP system, of which the back-pressure turbine system is a special case. Taking into account that thermal power plants usually have a minimum stable operation level, the operating area of the fixed back-pressure turbine system is represented by the line E–F in the Figure. The corresponding operating area of a pass-out turbine system (without reduction operation) is represented by the polygon A–B–F–E. In some cases the turbine characteristics require a minimum level of heat production in proportion to electricity, and with such a constraint the feasible operating area is reduced to C–D–F–E. Finally, with a reduction operation the feasible operating area is expanded to the polygon C–D–F–H–G–E in the Figure. Similarly, the operating area of a back-pressure turbine system with a reduction operation capability would be expanded to E–F–H–G.

Denoting the electrical efficiency in the full condensing mode (point B) by η_B, the total efficiency in the full CHP mode (point F) by η_F, the heat-to-power ratio (inverse slope of line E–F) by *R*, and the slope of the iso-fuel line (B–F) by *S*, we can easily write the relations between these as follows:

$$\eta_B = \frac{\eta_F \times (1 + R \times S)}{(1 + R)}$$

$$\eta_F = \frac{\eta_B \times (1 + R)}{1 + R \times S}$$

$$S = \frac{\eta_B \times (1 + R) - \eta_F}{\eta_F \times R}$$

The core TIMES parameters for modeling the CHP attributes are listed in Table 17.

Table 17: Core TIMES parameters related to the modelling of CHP processes.

Attribute name	Description
ACT_EFF	Efficiency: amount of activity produced by 1 unit of input flow
ACT_MINLD	Minimum stable level of operation
NCAP_CHPR	Heat-to-power ratio *
NCAP_CEH	Coefficient of electricity to heat *
NCAP_CDME	Efficiency in full condensing mode
NCAP_BPME	Efficiency in back-pressure mode (full CHP mode) *
NCAP_AFA / NCAP_AFC	Bound on the annual utilization factor

* Only taken into account for processes defined to be of type CHP with the set **prc_map**.

4.1.2 Defining CHP attributes in TIMES

4.1.2.1 Back-pressure turbine systems

For modelling a fixed back-pressure turbine system in TIMES, the following approach is recommended:

- Define the PCG of the process to consist of both the electricity and heat output commodities (using the set **prc_actunt**);
- Define the process type to be CHP (using the set **prc_map**);
- Use the electrical output as the basis of the process activity, and choose the capacity unit accordingly (using the parameter PRC_CAPACT).
- Define the process electrical efficiency (by using the parameter ACT_EFF);

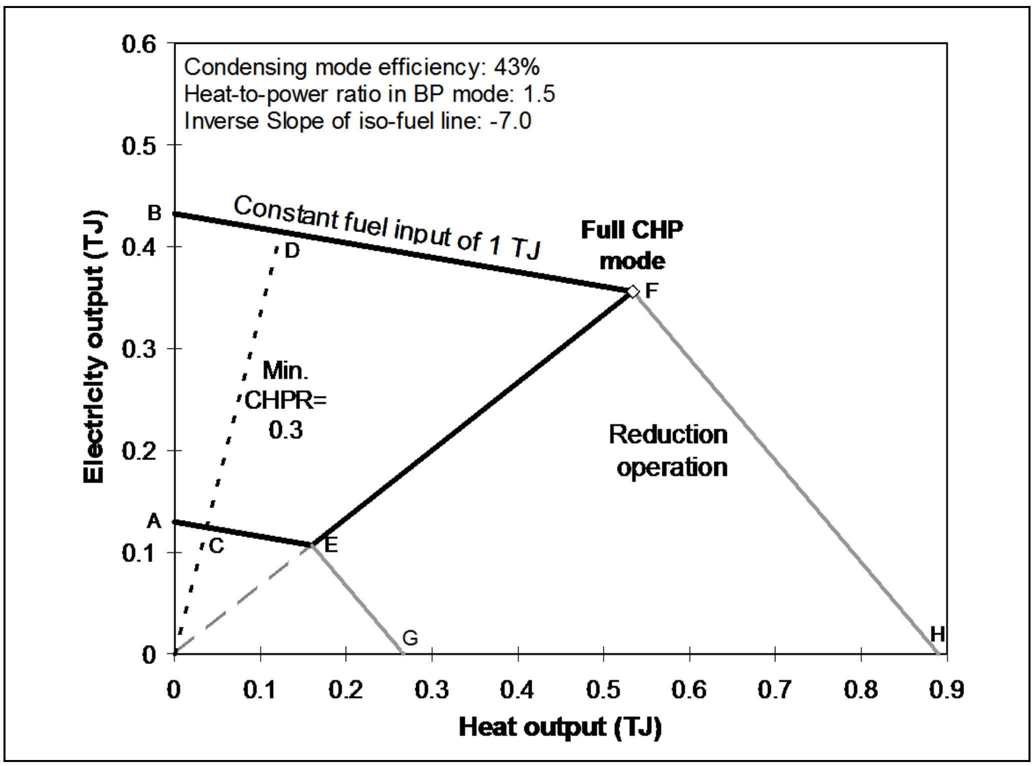


Figure 10: Illustration of basic CHP characteristics supported in TIMES.

- Define the process cost parameters accordingly; for example, specify the investment and fixed O&M costs per electrical capacity;
- Define the fixed heat-to-power ratio (using the parameter NCAP_CHPR);
- Optionally, define also a maximum annual utilization factor considering the typical optimal sizing of CHP plants in proportion to the heat demand in the heat network represented (using the parameter NCAP_AFA);
- Optionally, define a minimum stable operation level (using ACT_MINLD).

All the input data specifications mentioned above should be quite straightforward. Note that the NCAP_CEH parameter is not needed at all in the fixed turbine case.

For back-pressure turbine technologies that have a reduction operation capability, one can enable the reduction option by adding to the process a third output of a dummy commodity, which is of type NRG and has a limit type 'N', and is also a member of the PCG. The model generator will automatically assign such a dummy output to the reduction operation, and will adjust the process transformation equation accordingly.

4.1.2.2 Pass-out turbine systems

For modelling a flexible pass-out turbine system in TIMES, the following approach is recommended (but see additional remarks below):

- Define the PCG of the process to consist of both the electricity and heat output commodities (using the set **prc_actunt**);
- Define the process type to be CHP (using the set **prc_map**);
- Use the maximum electrical output as the basis of the process activity, and choose the capacity unit accordingly (using the parameter PRC_CAPACT). [35]
- Define the process electrical efficiency according to the maximum electrical efficiency (at point D in Figure 10), by using the parameter ACT_EFF;
- Define the process cost parameters accordingly, for example, specify the investment and fixed O&M costs per unit of electrical capacity;
- Define the maximum heat-to-power ratio (excluding any reduction operation), and optionally also the minimum heat-to-power ratio (using the parameter NCAP_CHPR);
- Define the slope S of the iso-fuel line (the line B–F in Figure 10) by specifying NCAP_CEH=S (where $-1 < S < 0$, as in Figure 10);
- Optionally, define also a maximum annual utilization factor considering the typical optimal sizing of CHP plants in the heat network represented (using the parameter NCAP_AFA and/or NCAP_AFC);
- Optionally, define a minimum stable operation level (using ACT_MINLD).

Again, the specifications should be quite straightforward. The slope S of the iso-fuel line represents the amount of electricity lost per heat gained. In the example of Figure 10, the inverse of the slope has the value 7 and so one would define NCAP_CEH = –1/7.

[35] The activity remains constant over the iso-fuel line, but the electricity output varies when moving along it. Maximum electrical output is thus usually the most convenient quantity along this line for defining the basis of the process activity and capacity. This choice should then be consistently reflected in the input data (see Table 18).

Alternatively, if it would seem more convenient to define both the condensing mode efficiency and the full CHP efficiency, that can be done by using the parameters NCAP_CDME (condensing mode efficiency) and NCAP_BPME (back-pressure mode efficiency). When these two parameters are used, the NCAP_CEH and ACT_EFF parameters should then not be used at all. The activity will in this alternative approach always represent the electricity output in condensing mode.

For pass-out turbine technologies that have a reduction operation capability, one can enable the reduction option by adding to the process a third output of a dummy commodity, which is of type NRG and has a limit type 'N', and is also a member of the PCG. The model generator will automatically assign such a dummy output to the reduction operation, and will adjust the process transformation equation accordingly.

4.1.2.3 Alternative choices for defining the activity basis

As indicated above, the recommended basis of the activity of a CHP technology is the maximum electricity output, because the available technology data is usually best suited for using the electricity output as the basis for the activity. However, also the total energy output in full CHP mode can be used as the basis for the activity, should that be a more convenient way of defining the process data.

The below summarizes the different options modelling CHP processes according to the choice of the main efficiency parameters. Note that the cases with $-1 < \text{CEH} \leq 0$ and $0 \leq \text{CEH} < 1$ are identical when there is no lower bound for NCAP_CHPR specified.

Table 18: Alternative ways of modelling efficiencies of CHP processes.

Characteristic	**Choices of parameters for modelling CHP efficiencies**			
Efficiency parameters	ACT_EFF + NCAP_CEH			NCAP_CDME+ NCAP_BPME
Value of CEH	−1<CEH≤0	0≤CEH<1	CEH ≥ 1	None
Interpretation of CEH	Decrease in electricity output per unit of heat gained (when moving towards full CHP mode)	Loss in electricity output per unit of heat gained (when moving towards full CHP mode)	Loss in heat output per unit of electricity gained (when moving towards con-densing mode)	None
Activity	Max. electricity output	Electricity output in full condensing mode	Total energy output in full CHP mode	Electricity output in condensing mode
Capacity	Electrical capacity	Electrical capacity	Electrical+heat capacity	Electrical capacity
Efficiency specification	Max. electrical efficiency (=ACT_EFF) + the CEH specification	Electrical efficiency in full condensing mode (=ACT_EFF) + the CEH specification	Total efficiency in full CHP mode (=ACT_EFF) + the CEH specification	Electrical efficiency in condensing mode + total efficiency in full CHP mode
Investment & fixed O&M costs	Per electrical capacity	Per electrical capacity	Per electrical+heat capacity	Per electrical capacity
Variable costs	Per activity (see above)	Per activity (see above)	Per activity (see above)	Per activity (see above)

4.2 Inter-regional exchange processes

4.2.1 Structure and types of endogenous trade

In TIMES, the inter-regional trading structure of a given commodity basically consists of one or several exchange processes (called IRE processes), each of which defines a portion of the trading network for the commodity. The individual sub-networks can be linked together through common intermediating regions. As an example, electricity trade can be conveniently described by bi-lateral exchange processes (see Figure 12). But bi-lateral trading between all pairs of regions may become onerous in terms of data and model size. It is therefore useful to consider the other trade structure of TIMES, called multi-lateral trade, where regions trade with a common market (Figure 11). For either structure, the topology of the trading possibilities are all defined via the set **top_ire** of quintuples {r1,c1,r2,c2,p}, where **r1**, **r2** are the exporting and importing regions respectively, **c1**, **c2** are the names of the traded commodity in regions **r1** and **r2** respectively, and **p** is the process identifier. Process **p** is a process in both regions. It has to be defined only once, but one can add parameters to it in both regions (e.g. costs, bounds, etc.). Nearly every piece of data in TIMES has to be assigned to a region.

TIMES provides considerable flexibility in the definition of trading structures. Each sub-network defined for a single exchange process can have the general structure shown in Figure 11. A trading structure that involves both several supply (export) regions and several demand (import) regions cannot be defined without introducing an intermediating 'market' region (R_M). Whenever such an intermediate region is defined between (at least) two different regions, the model generator will assume that the structure is actually meant to ignore the intermediate node-region shown in Figure 11, by generating a single trade

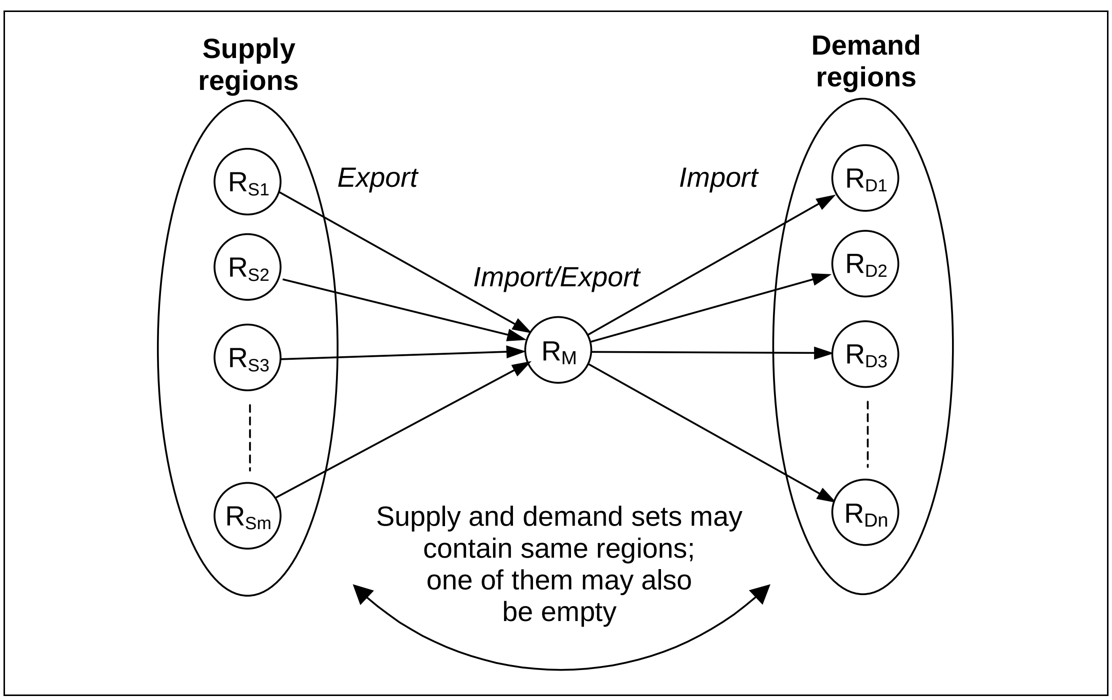


Figure 11. General structure of the pair-wise specification of the trading sub-network allowed in TIMES for a single exchange process.

balance equation directly between all the export and all the import flows. If the intermediate step should nonetheless be included, for example, to reflect a physical market hub in the region R_M, this can be accomplished by dividing the sub-network into two parts, by using two exchange processes. Consequently, depending on the user's choice, the trading relationships shown in Figure 11 can be modeled both with and without the intermediate transportation step through the market region.

The general structure allowed for the trading sub-networks can be further divided into four cases, which will be discussed below in more detail:

- Case 1: Bi-lateral trading.
- Case 2: Unidirectional trade from some export regions into a single importing region
- Case 3: Multi-directional trade from a single export region to several importing regions
- Case 4: General multi-lateral trading structure

Trading without need for explicit marketplace definition

Cases 1, 2 and 3 fall in this category. Bi-lateral trade takes place between pairs of regions. An ordered pair of regions together with an exchange process is first identified, and the trade through the exchange process is balanced between these two regions. Whatever amount is exported from region *i* to region *j* is imported by region *j* from region *i* (possibly with an adjustment for transportation losses). The basic structure is shown in Figure 12. Bi-lateral trading can be fully described in TIMES by specifying the two pair-wise connections in **top_ire**. The capacity and investment costs of the exchange process can be described individually for both regions. For Cases 2 and 3, the general structure of the trade relationships is shown in Figure 13. Also in these cases the definition of the trading structure is easy, because the relationships can be unambiguously described by pair-wise **top_ire** specifications between two regions.

Trading based on marketplace

Case 4 is covered by the generic structure shown in Figure 11. Trading occurs in this case between at least three regions, and involves both several exporting regions and several importing regions. In this type of trade, the commodity is ‘put on the market’ by each region participating in the supply side of the market and may be bought by any region participating in the demand side of the market. This case is convenient for global commodities such as emission permits or crude oil where the transportation cost from R_i to

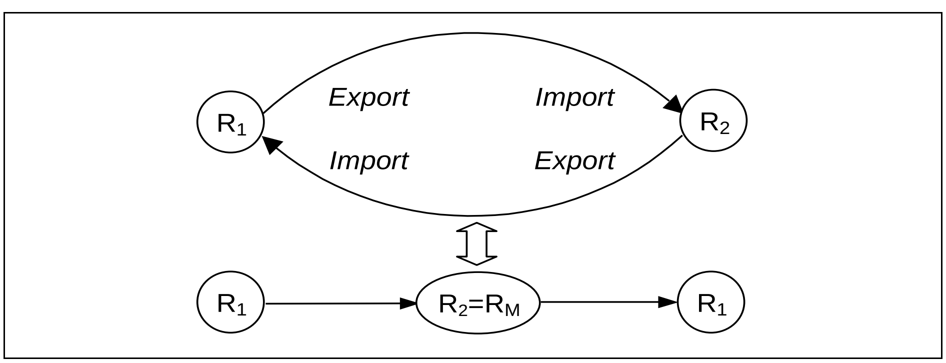


Figure 12. Case 1: Bi-lateral trade (both R_1 and R_2 qualify as R_M).

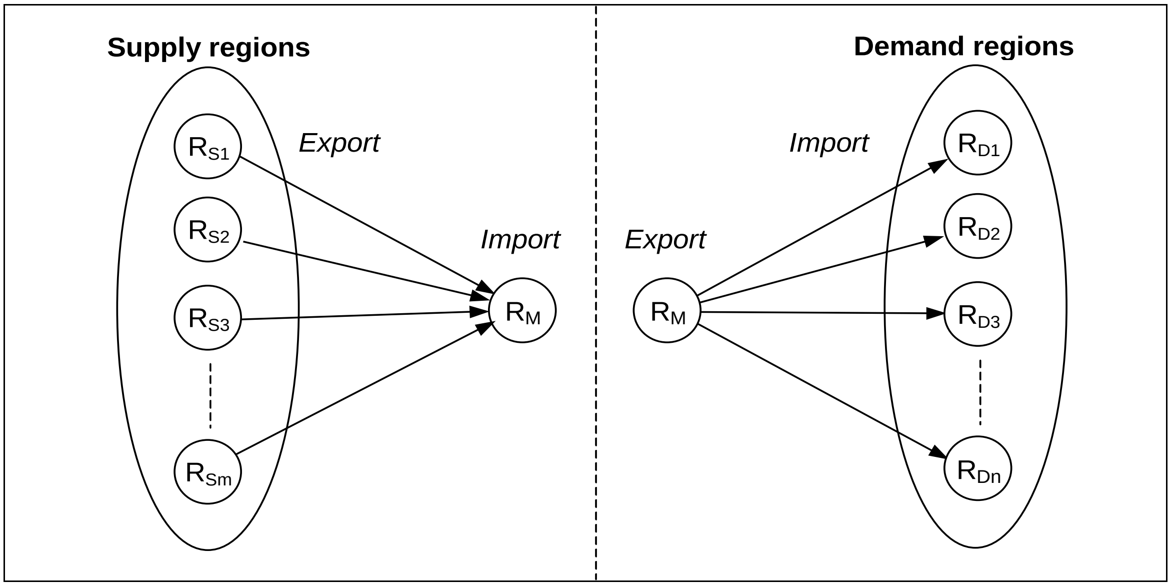


Figure 13. General structure of unidirectional trade into a single import region (Case 2, left) and multidirectional trade from a single export region (Case 3, right).

R_j may be approximated by $Cost_i$+$Cost_j$ (rather than a more accurate cost such as C_{ij}). When the exact cost (or losses) are strictly dependent on the pair i,j of trading regions, it may be more accurate to use bilateral trade.

In general, there are many different possibilities for defining the multi-lateral structure by using the pair-wise **top_ire** specifications. In order to comply with the structure allowed in TIMES, the user has to decide which of the regions represents the 'marketplace', i.e. is chosen to be the R_M shown in Figure 11. Note that the market region will participate both in the supply and demand side of the market. The TIMES model generator automatically identifies this general type of trading on the basis of the **top_ire** topology defined by the user. Therefore, the user only needs to define the possible trading relationships between regions into the set **top_ire**. If there are ***n*** supply regions and ***m*** demand regions, the total number of entries needed in **top_ire** for defining all the trade possibilities is ***n+m–2*** (counting the market region to be included in both the supply and demand regions. Although the market region has to be defined to be an intermediate node in the structure, the model generator will actually <u>not</u> introduce any intermediate step between the export and import regions.

The timeslice levels of the traded commodity may be different in each region (as well as the commodity name). However, some appropriate common timeslice level must be chosen for writing the market balance equation. *That common level is the level attached to the exchange process in the market region*. In all other respects, ***the market region is not treated in any way differently*** from the other regions participating in the market. Never-theless, the user can of course provide different data for the different regions, for example investment costs or efficiencies for the exchange process can be differentiated by region.

If the sets of supply and demand regions participating in the market should actually be disjoint, even in that case the user has to choose one of the regions to be used as the intermediate market region. The imports to or exports from the market region can then be switched off by using an IRE_XBND parameter, if that is considered necessary.

Remarks on flexibility

1. Any number of exchange processes can be defined for describing the total trade relationships of a single commodity (but see warning 1 below).
2. The names of traded commodities can be different in each region participating in the trade. In addition, also the import and export names of the traded commodities can be different (but see warning 2 below). This could be useful e.g. in the case of electricity, for which it is common to assume that the export commodity is taken from the system after grid transport, while the import commodity is introduced into the system before the grid.
3. Any number of commodities can be, in general imported to a region or exported from a region through the same process (but see warning 2 below).

Warnings

1. For each exchange process of any traded commodity, the total structure of the trading sub-network, as defined in **top_ire**, must comply with one of the basic structures supported by TIMES (Cases 1–4). If, for example, several bi-lateral trading relationships are defined for the same commodity, they should, of course, not be defined under the same process, but each under a different process.
2. If the export and import names for a market-based commodity (c) are different in the market region, no other commodities should be imported to the market region through the same exchange process as commodity c.
3. The model generator combines the trading relationships of a single process into a single market whenever there is an intermediate region between two different regions. If, however, the intermediate exchange step should be explicitly included in the model, the trading sub-network should be divided between two different exchange processes.

Example

Assume that we want to set up a market-based trading where the commodity CRUD can be exported by regions A, B, C, and D, and that it can be imported by regions C, D, E and F. First, the exchange process and marketplace should be defined. For example, we may choose (C,XP,CRUD) as the marketplace, where XP has been chosen to be the name of the exchange process (recall that process XP is declared only once but exists in all trading regions, possibly with different parameters). The trade possibilities can then be defined simply by the following six **top_ire** entries:

```
SET PRC / XP /;
SET TOP_IRE /
  A .CRUD .C .CRUD .XP
  B .CRUD .C .CRUD .XP
  D .CRUD .C .CRUD .XP
  C .CRUD .D .CRUD .XP
  C .CRUD .E .CRUD .XP
  C .CRUD .F .CRUD .XP
/;
```

To complete the RES definition needed for the exchange process, in addition only the set **prc_actunt(r,p,c,u)** needs be defined for the exchange process XP:

```
SET PRC_ACTUNT /
   A .XP .CRUD .PJ
   B .XP .CRUD .PJ
   C .XP .CRUD .PJ
   D .XP .CRUD .PJ
   E .XP .CRUD .PJ
   F .XP .CRUD .PJ
/;
```

These definitions are sufficient for setting up of the market-based trade. Additionally, the user can, of course, specify various other data for the exchange processes, for example investment and distribution costs, efficiencies and bounds.

4.2.2 Input sets and parameters specific to trade processes

TIMES input SETs that have a special role in trade processes are the following:

- **top_ire(r1,c1,r2,c2,p):** For bi-lateral trade, unidirectional trade into a single destination region, and multidirectional trade from a single source region, **top_ire** should contain the corresponding entries from the exporting region(s) **r1** to the importing region(s) **r2**.
  For market-based trade, **top_ire** must contain entries for each exporting region to the intermediate market region, and from the market region to each importing region. Each region may be both exporting and importing. One may thus force even a bi-lateral exchagne to be modeled as market-based trade, by introducing an additional **top_ire** entry within the desired market region between the exported and imported commodity. .Instead of two trade balance equations, only one market balance equation is then generated.
- **prc_aoff(r,p,y1,y2):** Override used to control in what years (not periods) a process is unavailable. This set is not specifically related to exchange processes. However, in the case of market-based trading it can be used to switch off the entire commodity market for periods that fall within the range of years given by **prc_aoff**. The market will be closed for all commodities exchanged through the process (**p**). If trading should be possible only between certain years, even multiple entries of **prc_aoff** can be specified.

All the **top_ire** specifications are handled for the user by the user shell (VEDA/ANSWER) according to the characterization of the trade processes.

Additional remarks:

1. Commodity type can be used as the primary group of IRE processes. All commodities of that type, traded through the process, will then be included in the PCG.
2. Topology entries are automatically created on the basis of IRE_FLOSUM and FLO_EMIS defined for IRE processes (the latter only for ENV commodities).
3. In any non-bilateral trade, the marketplaces are automatically set by the model generator for any trade that involves an intermediate region between two different regions for the same exchange process (**p**) and same commodity (**c**), or if there are multiple destination (importing) regions for the same exporting region.

4. In market-based trade with **r** as the market region, the import/export regions participating in the market consist of all those regions that import/export commodity **c** from/into region **r** through process **p** (as defined in **top_ire**). The market region **r** by itself always participates in the market both as an importing and exporting region. However, the imports/exports of commodity (**c**) to/from the market region (**r**) can be switched off by using an IRE_XBND parameter, if necessary.

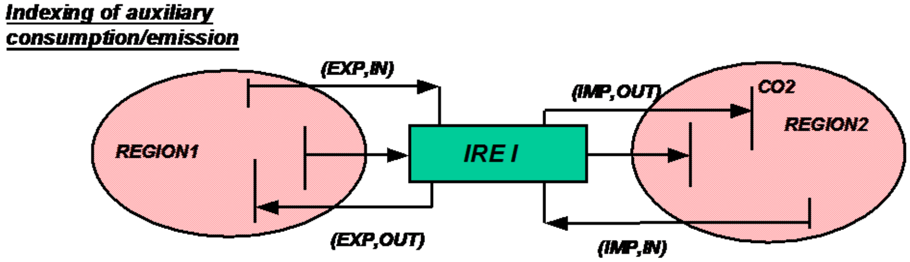
Input parameters

Input parameters specific to inter-regional exchange processes are listed in Table 19.

Table 19: Specific TIMES parameters related to the modelling of trade processes.

Attribute name (indexes)	Description
IRE_FLO (r1,y,p,c1,r2,c2,s2)	Coefficient that represents the efficiency of exchange from r1 to r2, inside an inter-regional process where both regions are internal. Note that separate IRE_FLOs are required for import and export. Default =1 for each top_ire direction specified. Time slice s2 refers to the region where the commodity arrives. Units: none
IRE_FLOSUM (r,y,p,c1,s,ie,c2,io)	Special attribute to represent auxiliary consumption (**io** = 'IN'), or production/emission (**io** = 'OUT') of commodity c2 due to the IMPort / EXPort (index **ie**) of the commodity c1 in region r by an inter-regional process **p**[36]. It is a fixed FLO_SUM with (one of) the pcg in that region. These relate commodities on the same side of the process. Auxiliary flows can also be specified on the process activity, by setting c1='ACT' in the IRE_FLOSUM parameter (or in a FLO_EMIS parameter).
IRE_BND (r1,y,c,s,r2,ie,bd)	Bound on the total import/export (index **ie**) into/from internal region **r1**, from/to region **r2**, where region **r2** may be internal or external[37]; **c** is the name of commodity in region **r1**. Default none.
IRE_XBND (r,y,c,s,ie,bd)	Bound on total imports/exports of commodity **c** in region **r**, to/from all destinations/sources, where **r** may be an internal or external region. (Default value: none)
IRE_CCVT (r1,c1,r2,c2)	Conversion factor between commodity units, from unit of **c1** in region **r1** to unit of **c2** in region **r2**, as part of inter-regional exchanges. Default = 1, when exchange permitted. Units: none.
IRE_TSCVT (r1,s1,r2,s2)	A matrix that transforms timeslices of region **r1** to region **r2** as part of inter-regional exchanges, including both internal and external. Default value = 1 when exchange permitted. Units: none.

[36] The indexing of auxiliary consumption flows or emissions of inter-regional exchange processes is illustrated in the figure below.



[37] The equation EQ(l)_XBND may have an external regional as region index (bounding the import from one external regions to all other regions).

Remarks:

1. In market-based trading the IRE_FLO parameter is taken into account on the export side only (representing the efficiency from the export region to the common marketplace). By using this convention, any bi-lateral exchange can be represented by a fully equivalent market-based exchange simply by choosing one of the two regions to be the marketplace, and adding the corresponding entry to the set rpc_market(r,p,c). The efficiency of the exports from the market region itself to the marketplace should also be specified with an IRE_FLO parameter, when necessary (r1=r2=market region).
2. If the user wants to specify efficiency on the import side of a market-based exchange, this can be done by using an IRE_FLOSUM parameter on the import side.
3. Similarly to any other pair of regions, the total amount of commodity imported to a region from the commodity market can be constrained by the IRE_BND parameter, by specifying the market region as the export region. Correspondingly, the total amount of commodity exported from a supply region to the marketplace can be constrained by the IRE_BND parameter by specifying the market region as the import region.

4.2.3 Availability factors for trade processes

In TIMES, capacity by default bounds only the activity. However, with the NCAP_AFC / NCAP_AFCS attributes, one can bound the import / export flows instead. Capacity then also refers to the nominal maximum import (or export) capacity, e.g. the capacity of a transmission line in either direction. One can thus simultaneously bound the import and export flows by the same capacity but with different availabilities, which can be useful with bi-directional exchange links with different availabilities in the import/export direction. All these availability factors can be defined either on a desired timeslice level (NCAP_AFC), or on individual timeslices (NCAP_AFCS).

The rules for defining the availabilities for trade flows can be summarized as follows:

- If the import/export commodities are different (c1/c2): Use NCAP_AFC(c1) for bounding the import flow and NCAP_AFC(c2) for bounding the export flow, or use NCAP_AFC('NRG') for applying the same availability to both flows.
- If input=output=c, specifying ***either*** NCAP_AFC(c) ***or*** NCAP_AFC('NRG') alone applies to both imports and exports (unless the process type is DISTR, see Section 4.2.4 below). However, if they are both specified, then NCAP_AFC(c) applies to the import flow while NCAP_AFC('NRG') applies to the export flow.

Remarks:

1. As any process has only a single capacity variable, the availabilities specified for the import/export flows are always proportional to the same overall capacity.
2. Note that any the availability factors defined by NCAP_AFC are multiplied by any NCAP_AF/NCAP_AFS/NCAT_AFA value if defined for the same timeslice.

4.2.4 Notes on other attributes for trade processes

There are important limitations of using the parameters for standard processes for IRE processes. The most important limitations are summarized Table 20 with regard to the parameters with the prefixes 'ACT_', 'FLO_' and 'PRC_'. In addition, none of the CHP parameters, storage parameters (STG_*), or dispatching parameters (ACT_MINLD, ACT_UPS, ACT_CSTUP, ACT_LOSPL, ACT_CSTPL, ACT_TIME), can be used for IRE processes, and are ignored if used.

Table 20: Limitations of using standard process parameters for IRE processes.

Attribute name	Description	Limitations
ACT_EFF	Activity efficiency	Can not be used
FLO_BND	Bound on a process flow variable	The bound will apply to the sum of both imports and exports of the given commodity, or, alternatively, to the net imports when a true commodity group is specified in the parameter (e.g. NRG).
FLO_EFF	Amount of process flow per unit of other process flow(s) or activity.	Same as for FLO_EMIS.
FLO_EMIS	Amount of emissions per unit of process flow(s) or activity.	Can only be used on the activity, by specifying 'ACT' as the source group.
FLO_FR	Process flow fraction	Can not be used
FLO_FUNC	Relationship between 2 groups of flows	Can not be used
FLO_MARK	Process market share bound	The bound will apply to import flow if FLO_MARK≥0, and to export flow if FLO_MARK≤0.
FLO_SHAR	Process flow share	Can not be used
FLO_SUM	Multiplier for a commodity flow in a relationship between 2 groups of flows	Can not be used
PRC_MARK	Process group-wise market share bound	Same as for FLO_MARK.

Additional remarks with respect to inter-regional trade (IRE) processes:

- By using the process type indicator 'DISTR', the activity and capacity of an IRE process will be based on the import flow only, if the same commodity is both imported and exported. In this case also NCAP_AFC(c) will only apply to the import flow of c.
- In peaking equations, IRE processes are by default taken into account by having gross imports on the supply side and gross exports on the consumption side. By defining the IRE process as a member of the set PRC_PKNO, and also defining NCAP_PKCNT>0, only the net imports are taken into account on the supply side, which can be useful for regions having trade flows passing through the region.

4.3 Storage processes

4.3.1 Overview

The TIMES model generator provides tools for specifying the following types of storage processes:

- Standard timeslice storage (STG without additional storage type qualifier)
- Generalized timeslice storage (STG+STS)
- Day/night storage (STG+NST, or just NST if at ANNUAL level)
- Inter-period storage (STG+STK)

The process type indicator STG is automatically assigned also to all processes that have been defined to be of type STS, NST or STK, with the exception of ANNUAL level NST processes, which are implemented as normal processes (see Section 4.3.3 below). Therefore, the user only needs to specify one of {STG, STS, NST, STK} as the process type of a storage process.

In addition to the charged and discharged commodity, storage processes can also produce and consume auxiliary commodities (emissions, electricity, fuels, waste etc.). The flows of such auxiliary commodities can be defined to be proportional either to the activity, the main input flows, or the main output flows of the storage (see Section 4.3.5 below).

4.3.2 Timeslice storage

The standard timeslice storage operates within the timeslice cycles under the timeslices of the level immediately above the process timeslice level. Consequently, the commodity charged can be only stored over the cycle of timeslices under a single parent timeslice, and not between timeslices under different parent timeslices. For example, a standard DAYNITE level storage can only store the charged commodity over the timeslices under one season, and not between seasons.

The activity of a timeslice storage represents the storage level, i.e. the amount of energy/material stored in the storage, measured at the beginning of each timeslice. However, one should note that for a DAYNITE level storage, the level of the activity variable for each timeslice is the actual storage level multiplied by the number of days under the parent timeslice, in the same way as the level of the activity variables for standard processes is the daily activity in that timeslice multiplied by the number of days under the parent timeslice.

If a storage technology is capable of storing energy for longer periods than over daily cycles, one may consider combining a SEASON/WEEKLY level storage process with a DAYNITE storage. However, a DAYNITE level storage may also be generalized to provide a storage capability between seasons, and even between periods, by using the generalized timeslice storage type qualifier 'STS' (and both 'STS' and 'STK', if the inter-period storage capability should be included). Because the same storage capacity can be utilized on all timeslice levels, the general storage process type may thus provide a somewhat improved modeling of a multi-cycle storage.

4.3.3 Day/Night storage

Day/Night storage (NST) is a timeslice storage, which can store energy over the day-night cycles, but not over weekly or seasonal cycles. In its basic functionality, an NST storage does not differ much from a standard timeslice storage, the main difference being that one can define the charging timeslices by specifying them in the set **prc_nstts**.

Day/Night storage processes that produce ANNUAL level demand commodities can be modeled either as genuine storage processes or as standard processes with a night storage capability. In both cases 'NST' should be specified as the process type. If the process itself is defined to operate at the DAYNITE level, the process will be a genuine storage process, but if it is defined to operate at the ANNUAL level, it will be a standard process. For any such night storage devices, the charging and discharging commodity may be different, as defined via the set **top**.

When the NST process **p** is a genuine storage process, the input set **prc_nstts**(r,p,s) may be used for defining the charging timeslices **s**. Discharging can then only occur in timeslices other than the charging timeslices. Defining **prc_nstts** is required for all other genuine NST processes, except those serving an ANNUAL level demand, which can always discharge at the level of the demand, regardless of any **prc_nstts** defined.

In both types of NST storage, if the process is serving any ANNUAL level demand, the demand commodity is produced according to the load curve, while the charging can be optimized so that it occurs at night timeslices only. However, when the NST process is a normal process, it can be described in all other respects just as any other end-use technologies. For example, electric heating systems with accumulators can be described basically in the same way as direct electric heating systems, but with the additional night storage capability.

4.3.4 Inter-period storage

An inter-period storage process is able to store energy or material over periods. For example, a coal stockpile or a waste disposal site can be modeled as an inter-period storage. All inter-period storage processes should be defined to operate at the ANNUAL level, unless the generic timeslice process characterization (STS) is also specified.

The initial stock of an inter-period storage process can be specified by using the STG_CHRG parameter, which is interpolated such that it always includes the year at the beginning of the model horizon (B(t1)–1). The value of STG_CHRG in the year B(t1)–1 is used as the initial stock for inter-period storage. The allocation of the initial stock between the process vintages that are available at the beginning of the model horizon is left to be optimized by the model.

The activity of an inter-period storage is measured at the end of each period. Therefore, either by setting a lower bound on the activity or on the process availability, the storage can be prevented from getting fully discharged during any period. However, as there is no explicit accounting of the salvage value of the remaining contents of an inter-period storage, it may also be considered reasonable to allow discharging the storage fully in the last period, for taking into account the value of the storage.

4.3.5 Auxiliary storage flows

Storage processes can have any amount of auxiliary input or output commodities, as long as they are distinct from the main storage commodity. The flows of the auxiliary commodities can only be defined to be fixedly proportional either to the activity, the main input flows, or the main output flows. The main flows of timeslice and inter-period storage processes are the flows of the charged and discharged commodities included in the set primary commodity group PCG of the process. In the day/night storage processes, the main flows consist of all commodities in the primary and shadow groups of the process (see documentation).

The relation between the auxiliary flows and the activity or main flows should be defined by using the PRC_ACTFLO and the FLO_FUNC parameters. For example, if the main storage flows of the process consist of the commodity 'STORED', and the auxiliary commodity is 'AUX', the auxiliary flow can be defined in the three following ways, corresponding to the cases where the auxiliary flow is proportional to the activity, the input flow, or the output flow, respectively:

```
PRC_ACTFLO(r,t,p,'AUX')                 ! AUX proportional to activity
FLO_FUNC(r,t,p,'STORED','AUX',s)        ! AUX proportional to input flow
FLO_FUNC(r,t,p,'AUX','STORED',s)        ! AUX inversely proportional to output flow
```

These auxiliary storage flow relations have been implemented by adding a new TIMES equation EQ_STGAUX(r, v, t, p, c, s). As the auxiliary storage flows are represented by standard flow variables, any flow-related cost attributes and UC constraints can be additionally defined on these auxiliary flows. However, no transformation equations can be defined between any auxiliary storage flows. Therefore, if, for example, some auxiliary flows should also produce emissions, also these emissions should be defined on the basis of the activity or main flows, and not by defining a relation between the auxiliary flow and the emission flow. Consequently, it is required that all auxiliary commodity flows related to storage processes, whether energy, material, or emissions, are described by using the three types of relations shown above.

A concrete example where these enhancements to the storage processes can be very useful is the modeling of waste management, and, in particular, the modeling of landfilling of different types of waste. Using inter-period storage processes for this purpose makes it possible to conveniently incorporate e.g. the following features in the waste management model:

- Modeling of methane emissions from landfilling in a dynamic way by using first-order decay functions for the gradual waste decomposition (optionally with different rates of decay for different waste qualities);
- Modeling of other waste management and emission reduction options both before and after landfilling;
- Incorporating gate fees to landfill sites (by defining costs on an input-based auxiliary storage flow).

4.3.6 Input sets and parameters specifically related to storage processes

Input sets

There is only one TIMES input SETs specifically related to storage: prc_nstts. However, there are important storage-specific aspects related to each of the following input sets:

- **prc_map(r,prc_grp,p):** Defines the process as a storage process, where **prc_grp**=STG/STS/NST/STK according to the desired storage type.
- **prc_actunt(r,p,cg,units_act):** Definition of the commodity/commodities in the PCG, i.e. those that are stored. Set of quadruples such that the members of the commodity group **cg** is used to define the charged and discharged commodity of storage process **p**, with activity units **units_act**, in region **r**. If the charged and discharged commodities are different, the group **cg** should preferably contain both of them, but if the user shell does not allow that, the model generator will automatically assign to the PCG any commodities on the shadow side that are of the same type than those already in the PCG, and are not verified to be auxiliary commodities. A commodity type can also be used as the primary group of storage processes. All commodities of that type will then be included in the PCG.
- **top(r,p,c,io):** Definition of the charged (io=IN) discharged (io=OUT) and optional auxiliary input/output commodities for storage process p in region r. The set **top_ire** should thus first and foremost contain the input/output indicators for the stored commodities defined by **prc_actunt** (see above), but should include also any auxiliary input/output commodities assumed for the process. When the charged and discharged commodity is the same, that commodity can optionally be defined only as an input or only as an output, and in that case it will be connected to the commodity balance equations either only on the production or only on the consumption side, instead of being connected on both sides.
- **prc_nstts(r,p,s):** For genuine night storage process **p** in region **r**, defines the timeslices **s** to be the charging timeslices, at which discharging cannot occur.

Remarks

In TIMES, the input (charge) and output (discharge) commodity of a storage process is usually the same commodity (input=output). When so, and this commodity is defined both as an input and an output of the process, the input and output flows will be taken into account in the commodity balance equations on different sides: the input on the consumption side, and the output on the production side.

However, in some cases this design has proven to be undesirable, because due to the nature of the storage processes, the input and output flows can usually be made arbitrarily large without affecting the storage operation or costs. That is so because the input flow may also by-pass the storage in the same timeslice or period, without being stored, and will then be directly converted into the output flow, without any costs or efficiency losses (unless STG_EFF is being used). Such arbitrary input/output flows can also make the total commodity production arbitrarily large, thereby rendering VAR_COMPRD a very unreliable measure of the size of the commodity market. This can be undesirable with respect to various market-share constraints that are usually defined on the basis of the VAR_COMPRD values.

In order to avoid any arbitrary storage flows on the production or consumption side, the input/output flows can be defined to be both connected either on the production or consumption side, instead of being on different sides. This will prevent the undesirable impacts of such arbitrary flows. The desired side can be chosen by the user by defining the commodity only as an output (production side) or as an input (consumption side).

Input parameters

The TIMES input parameters that are specific to storage processes or have a specific functionality for storage processes are summarized in Table 21.

Table 21: Specific TIMES parameters related to the modelling of storage processes.

Attribute name (indexes)	**Description**
STG_CHRG (r,y,p,s)	Exogenous amount assumed to be charged into storage **p**, in timeslice **s** and year **y**. For timeslice storage this parameter can be specified for each period, while for inter-period storage this parameter is only taken into account for the first period, to describe the initial content of the storage at the beginning of the model horizon. Units: Unit of the storage input flow.
STG_EFF (r,y,p)	Coefficient that represents the storage efficiency of a storage process **p** in region **r**. Applied at the commodity balance to the output flow.
STG_LOSS (r,y,p,s)	Coefficient that represents the annual storage losses of a storage process **p** in region **r**, as a fraction of the (average) amount stored, corresponding to a storage time of one year. If the value specified is negative, the corresponding annual losses are interpreted as an annual equilibrium loss (under exponential decay).
STGIN_BND (r,y,p,c,s,bd)	Bound on the input flow of commodity **c** of storage process **p** in a timeslice **s**. Units: Unit of the storage input flow. (Default value: none)
STGOUT_BND (r,y,p,c,s,bd)	Bound on the output flow of commodity **c** of storage process **p** in a timeslice **s**. Units: Unit of the storage input flow. (Default value: none)
FLO_FUNC (r,y,p,c1,c2,s)	Defines the ratio between the flow of commodity **c2** and the flow of commodity **c1**, in timeslice **s**, in other words, an efficiency coefficient giving the flow of commodity **c2** per one unit of flow of commodity **c1**. For storage processes, can be used for defining amount of discharge in **c2** per unit of auxiliary flow of **c1**, or amount of auxiliary flow of **c2** per unit of charging in **c1**.
PRC_ACTFLO (r,y,p,c)	Defines a conversion coefficient between the activity and the flow in commodity **c**. For storage processes, PRC_ACTFLO can be used for the commodities in the PCG in the standard way, but also for defining the amount of auxiliary flow of **c** per unit of activity.
NCAP_AFC (r,y,p,cg,tslvl)	Can be used for defining availability factors for the process activity (amount stored), process output flow, or process input flow, or any combination of these. See Section 6.3 for additional information.
NCAP_AFCS (r,y,p,cg,s)	As NCAP_AFC above, but can be specified for individual timeslices. NCAP_AFCs values override NCAP_AFC values defined at the same level.

4.3.7 Availability factors for storage processes

In TIMES, capacity by default bounds only the activity. For storage, this means the amount of stored energy. However, with the NCAP_AFC/NCAP_AFCS attributes, one can bound the output (or input) flows instead. Capacity then also refers to the nominal output (or input) capacity, e.g. electrical capacity of a pumped hydro power plant. In addition, one can bound simultaneously both the output and input flows by the capacity, which can be useful if the charging rate is limited by the capacity as well. Moreover, one can simultaneously define a bound also for the activity (the amount stored) in proportion to the same capacity variable. All these availability factors can be defined either on a desired timeslice level (NCAP_AFC), or on individual timeslices (NCAP_AFCS).

The rules for defining the availabilities for storage flows/activity can be summarized as follows:

- If the input/output commodities are different (c1/c2): Use NCAP_AFC(c1) for bounding the input flow and NCAP_AFC(c2) for bounding the output flow.
- If input=output=c, NCAP_AFC('NRG') will define the availability factor for both the input and output flow, while NCAP_AFC(c) will define the availability factor for the output flow only, overriding any NCAP_AFC('NRG') value if that is also specified (assuming NRG is the type of the stored commodity).
- NCAP_AFC('ACT') can additionally be used for bounding the activity (the amount stored); in this case one must bear in mind that any capacity expressed in power units (e.g. MW/GW) is assumed to represent a storage capacity equivalent to the amount produced by full power during one full year/week/day for SEASON/WEEKLY/DAYNITE level storage processes, respectively. Knowing this, the availability factor can be adjusted to correspond to the assumed real storage capacity. For example, a capacity of 1 GW is assumed to represent a storage capacity of 24 GWh for a DAYNITE storage, and if the real daily storage capacity is, say 8 GWh / GW, the maximum availability factor should be 0.333.

Remarks:

1. As any storage process has only a single capacity variable, the assumption is that the availabilities specified for the output/input flows and the activity are all proportional to the same capacity.
2. Note that any the availability factors defined by NCAP_AFC are multiplied by any NCAP_AF/NCAP_AFS/NCAT_AFA value if defined for the same timeslice.

4.3.8 Notes on other attributes for storage processes

There are important limitations of using standard processes parameters for storage processes. The most important limitations are summarized in Table 22, with regard to the parameters with the prefixes 'ACT_', 'FLO_' and 'PRC_'. In addition, none of the CHP parameters, IRE parameters (IRE_*), or dispatching parameters (ACT_MINLD, ACT_UPS, ACT_CSTUP, ACT_LOSPL, ACT_CSTPL, ACT_TIME), can be used for storage processes, and are ignored if used.

Table 22: Limitations of using standard process parameters for storage processes.

Attribute name	Description	Limitations
ACT_EFF	Activity efficiency	Can not be used
FLO_BND	Bound on a process flow variable	Can only be used for bounding auxiliary storage flows.
FLO_COST	Added variable cost for commodity flow	Can only be used for the charging (input) flow(s), and for all auxiliary flows.
FLO_DELIV	Delivery cost for commodity flow	Can only be used for the discharge (output) flow(s), and for all auxiliary flows.
FLO_EFF, FLO_EMIS (r,y,p,cg,c,s)	Amount of process flow per unit of other process flow(s) or activity.	Can only be used for defining an auxiliary flow per unit of activity, by specifying 'ACT' as the source group (cg).
FLO_FR	Process flow fraction	Can only be used for auxiliary storage flows.
FLO_FUNC	Relationship between 2 groups of flows	Can only be used for defining auxiliary storage flows.
FLO_MARK	Process market share bound	For a stored commodity, the bound will apply to discharge flow when FLO_MARK≥0, and to charging flow if FLO_MARK≤0.
FLO_SHAR (r,y,p,c,cg,s,bd)	Process flow share	Can only be used among auxiliary flows, and for bounding the output flow (c) in proportion to the activity (cg='ACT')
FLO_SUM	Multiplier for a commodity flow in a relationship between 2 groups of flows	Can only be used among auxiliary flows.
FLO_TAX, FLO_SUB	Tax/subsidy for the production/use of commodity by process	Can only be used for auxiliary storage flows
PRC_MARK	Process group-wise market share bound	Same limitations as for FLO_MARK.

Additional remark on peaking equations:

- In peaking equations, storage processes producing the peaking commodity are by default taken into account by their capacity on the supply side, and not at all by their flows (charging/discharging). By defining the storage process as a member of the set PRC_PKNO, and also defining NCAP_PKCNT>0, the discharge from the storage is taken into account on the supply side instead of the capacity, and the charging into the storage is included on the consumption side (should such happen in the peak timeslice). That can be recommended, whenever the capacity represents the amount stored, and not the output capacity, and may be reasonable even for storage processes where the capacity represents the nominal maximum output flow.

5 Variables

This chapter describes each variable name, definition, and role in the TIMES Linear Program. To facilitate identification of the variables when examining the model's source code, all variable names start with the prefix VAR_. The value assigned to each variable indexed by some time period, represents the average value in that time period, but the case of VAR_NCAP(v) is an exception, since that variable represents a point-wise investment decided at time period **v**. VAR_NCAP is discussed in detail below.

Table 23 is a list of TIMES variables by category, with brief description of each variable.

Remarks on Table 23:

- Many variables that are related to a process have two period indexes: **t** represents the current period, and **v** represents the vintage of a process, i.e. the period when the investment in that process was decided. For the VAR_NCAP variable, **t** is by definition equal to **v**. For other variables, **t** ≥ **v**, if the process is vintaged (**prc_vint**), i.e., the characteristics of the process depend on the vintage year. If the process is non-vintaged, the characteristics of the capacity of a process are not differentiated by its vintage structure, so that the vintage index is actually not needed for the variables of a non-vintaged process. In these cases, the vintage index **v** is by convention set equal to the period index **t**.
- In Table 23, the variables are listed according to five categories, depending on what TIMES entity they represent. In the rest of the chapter, the variables are listed and fully described in alphabetical order.
- Table 23 does not list the variables used in the Climate Module, Damage Cost and ETL extensions of TIMES, which are fully documented in Appendices A, B, and C, respectively.
- In the Objective function category, Table 23 also lists several parameters that stand for certain portions of the objective functions. These are not bona fide GAMS variables, but mostly serve as convenient placeholders for this documentation, and also as useful parameters that may be reported in the solution.

Table 23. List of TIMES variables by category

Category	Variable name	Brief description
Region related		
	VAR_CUMCST	Cumulative amount of regional cost/tax/subsidy
Process related		
	VAR_ACT	Annual activity of a process
	VAR_CAP	Current capacity of a process, all vintages together
	VAR_NCAP	Investment (new capacity) in a process
	VAR_DNCAP VAR_SNCAP	Binary variable (VAR_DNCAP) and semi-continuous variable (VAR_SNCAP) used with the discrete investment option (see EQ_DSCNCAP)
	VAR_RCAP	Retired capacity of a process in a period by vintage
	VAR_SCAP	Cumulative retired capacity of a process in a period
	VAR_DRCAP	Binary variable for discrete capacity retirements
	VAR_UPS	Started-up, shut-down, and off-line capacities
Commodity related		
	VAR_BLND	Blending variable (for oil refining)
	VAR_COMNET	Net amount of a commodity
	VAR_COMPRD	Gross production of a commodity
	VAR_CUMCOM	Cumulative gross/net production of commodity
	VAR_ELAST	Variables used to linearize elastic demand curves
Flow (Process and Commodity) related		
	VAR_FLO	Flow of a commodity in or out of a process
	VAR_CUMFLO	Cumulative amount of process flow/activity
	VAR_IRE	Flow of a commodity in or out of an exchange process (trade variable)
	VAR_SIN/OUT	Flow of a commodity in or out of a storage process
Objective function related		
	VAR_OBJ	Variable representing the overall objective function (all regions together)
The following 10 parameters are not true variables of the LP matrix		
	OBJR	Parameter representing a regional component of the objective function.
	INVCOST	Parameter representing the investments portion of a regional component of the objective function
	INVTAXSUB	Parameter representing the taxes and subsidies attached to the investments portion of a regional component of the objective function
	INVDECOM	Parameter representing the capital cost attached to the dismantling (decommissioning) portion of a regional component of the objective function
	FIXCOST	Parameter representing the fixed annual costs portion of a regional component of the objective function
	FIXTAXSUB	Parameter representing the taxes and subsidies attached to fixed annual costs of a regional component of the objective function
	VARCOST	Parameter representing the variable annual cost portion of a regional component of the objective function
	VARTAXSUB	Parameter representing the variable taxes and subsidies of a regional component of the objective function

Category	Variable name	Brief description
	ELASTCOST	Variable representing the demand loss portion of a regional component of the objective function
	LATEREVENUES	Parameter representing the late revenue portion of a regional component of the objective function.
	SALVAGE	Parameter representing the salvage value portion of a regional component of the objective function
User Constraint related[38]		
	VAR_UC	Variable representing the LHS expression of a user constraint summing over regions (**uc_r_sum**), periods (**uc_t_sum**) and timeslices (**uc_ts_sum**).
	VAR_UCR	Variable representing the LHS expression of a user constraint summing over periods (**uc_t_sum**) and timeslices (**uc_ts_sum**) and being generated for the regions specified in **uc_r_each**.
	VAR_UCT	Variable representing the LHS expression of a user constraint summing over regions (**uc_r_sum**) and timeslices (**uc_ts_sum**) and being generated for the periods specified in **uc_t_each**.
	VAR_UCRT	Variable representing the LHS expression of a user constraint summing over timeslices (**uc_ts_sum**) and being generated for the regions specified in **uc_r_each** and periods in **uc_t_each**.
	VAR_UCTS	Variable representing the LHS expression of a user constraint summing over regions (**uc_r_sum**) and being generated for the periods specified in **uc_t_each** and timeslices in **uc_ts_each**.
	VAR_UCRTS	Variable representing the LHS expression of a user constraint summing over periods being generated for the regions specified in **uc_r_each**, the periods in **uc_t_each** and timeslices in **uc_ts_each**.

Notation for indexes: The following indexes are used in the remainder of this chapter:

r, r' = region; **v** = vintage; **t, t'** = time period; **y** = year; **p** = process; **c, c'** = commodity; **s**, **s'** = timeslice; **ie** = import or export; ***l*** = sense of a constraint (≥, =, or ≤). In addition, some indexes (**u; ble; opr; j; uc_n**) are used for specific variables only and are defined in their context.

5.1 VAR_ACT(r,v,t,p,s)

Definition: the overall activity of a process. VAR_ACT is defined by the EQ_ACTFLO equation either as the sum of outflows or as the sum of inflows of a particular (user selected) group of commodities, adequately normalized. If the process is not vintaged, the vintage index **v** is by convention set equal to the period index **t**.

Role: reports the activity of a process and implicitly defines how the capacity is measured, since the activity is bounded by the available capacity in the constraint

[38] In case the dollar control parameter VAR_UC is set to YES, the user constraints are always strict equalities (***l***=E) with the RHS constants replaced by the user constraint variables given in the table. The RHS bound parameter (UC_RHS(R)(T)(S)) are then applied to these user constraint related variables. See Section 5.20.

EQ(*l*)_CAPACT, e.g. if the activity of a coal power plant is defined over its electricity output, the capacity is measured in terms of the output commodity, e.g. $MW_{electric}$. Similarly, if the activity variable represents the input flow of coal, the capacity of the coal plant is measured in terms of the input commodity, e.g. MW_{coal}.

Bounds: Can be directly bounded by ACT_BND
User constraints: Can be directly referred to by UC_ACT

5.2 VAR_BLND(r,ble,opr)

Definition: amount of the blending stock **opr** in energy, volume or weight units needed for the production of the blending product **ble** in oil refinery modeling.

Role: used for specifying constraints on quality of the various refined petroleum products.

Bounds: Cannot be bounded.
User constraints: Cannot be referred to in user constraints.

5.3 VAR_CAP(r,t,p)

Definition: the installed capacity in place in any given year **t**, of all vintages of a process determined by the equation EQ(*l*)_CPT. The variable is equal to the sum of all previously made investments in new capacity, plus any remaining residual capacity installed before the modeling horizon,that has not yet reached the end of its technical lifetime, and minus any capacity that has been retired early.

Role: Its main purpose is to allow the total capacity of a process to be bounded. The variable is only created when

- capacity bounds (CAP_BND) for the total capacity installed are specified. In case only one lower or one upper capacity bound is specified, the variable is not generated, but the bound is directly used in the EQ(l)_CPT constraint.
- the capacity variable is needed in a user constraint, or
- the process is a learning technology (**teg**) in case that endogenous technological learning is used.

Bounds: Can be directly bounded by CAP_BND
User constraints: Can be directly referred to by UC_CAP

5.4 VAR_COMNET(r,t,c,s)

Definition: the net amount of a commodity at period **t**, timeslice **s**. It is equal to the difference between amount procured (produced plus imported) minus amount disposed (consumed plus exported).

Role: The variable is only created if a bound is imposed, or a cost is explicitly associated with the net level of a commodity.

Bounds: Can be directly bounded by COM_BNDNET
User constraints: Can be directly referred to by UC_COMNET

5.5 VAR_COMPRD(r,t,c,s)

Definition: the amount of commodity **c** procured at time period **t**, timeslice **s**.

Role: this variable is only created if a bound is imposed on total production of a commodity, or a cost is explicitly associated with production level of a commodity. The variable is defined through the equation EQE_COMPRD.

Bounds: Can be directly bounded by COM_BNDPRD
User constraints: Can be directly referred to by UC_COMPRD

5.6 VAR_CUMCOM(r,c,type,y1,y2)

Definition: the cumulative amount of commodity **c** produced in region **r** between years **y1** and **y2**, over all timeslices. The **type** indicator (PRD/NET) distinguishes between gross and net production.

Role: this variable is only created if a bound is imposed on cumulative gross/net production of a commodity. The variable is defined through the equations EQ_CUMPRD and EQ_CUMNET.

Bounds: Can be directly bounded by COM_CUMNET/ COM_CUMPRD
User constraints: Can be directly referred to by UC_CUMCOM

5.7 VAR_CUMCST(r, y1,y2,costagg,cur)

Definition: the cumulative amount of costs/taxes/subsidies according to the aggregation **costagg** in region **r** between years **y1** and **y2**, over all timeslices. The available cost aggregations are identified by the pre-defined members of the fixed index set **costagg**.

Role: this variable is only created if a bound is imposed on the cumulative amount of regional costs, taxes, and/or subsidies. The variable is defined through the equation EQ_BNDCST.

Bounds: Can be directly bounded by REG_CUMCST
User constraints: Cannot be referred to in user constraints

5.8 VAR_CUMFLO(r,p,c,y1,y2)

Definition: the cumulative amount of flow in commodity **c** by process **p** in region **r** between years **y1** and **y2**, over all timeslices. With the commodity name **c**='ACT' (reserved system label), the variable represents the cumulative amount of process activity.

Role: this variable is only created if a bound is imposed on the cumulative amount of process flow or activity. The variable is defined through the equation EQ_CUMFLO.

Bounds: Can be directly bounded by FLO_CUM / ACT_CUM
User constraints: Can be directly referred to by UC_CUMFLO/UC_CUMACT

5.9 VAR_DNCAP(r,t,p,u) / VAR_SNCAP(r,t,p)

Definition: VAR_DNCAP is only used for processes selected by the user as being discrete, i.e. for which the new capacity in period **t** may only be equal to one of a set of discrete sizes, specified by the user. For such processes, VAR_DNCAP is a binary decision variable equal to 1 if the investment is equal to size **'u'** and 0 otherwise. Thanks to an additional constraint, only one of the various potential sizes allowed for the investment at period **t** is indeed allowed.

VAR_SNCAP is only used for processes selected by the user as having semi-continuous amounts of new capacity, i.e. for which new capacity in period **t** may only be zero or between positive lower and upper bounds specified by the user.

Role: useful to mathematically express the fact that investment in process **p** at period **t** may only be done in discrete or semi-continuous sizes. See equation EQ_DSCNCAP in Chapter 6.

Bounds: Direct bounding not available, indirectly by NCAP_BND
User constraints: Not available

5.10 VAR_DRCAP(r,v,t,p,j)

Definition: this variable is used only for processes selected by the user as having discrete early capacity retirements, i.e. for which the retirement at period **t** may only be a multiple of a block size, specified by the user. For such processes, VAR_DRCAP is an integer decision variable equal to the number of blocks retired.

Role: needed for mathematically expressing the fact that early retirement in capacity of process **p** at period **t** may only be done in discrete amounts. See equation EQ_DSCRET in Chapter 6.

Bounds: Direct bounding not available, indirectly by RCAP_BND
User constraints: Not available

5.11 VAR_ELAST(r,t,c,s,j,l)

Definition: these variables are defined whenever a demand is declared to be price elastic. These variables are indexed by **j**, where **j** runs over the number of steps used for discretizing the demand curve of commodity **c** (**c** = energy service only). The **j**th variable stands for the portion of the demand that lies within discretization interval **j**, on side ***l*** (***l*** indicates either increase or decrease of demand w.r.t. the reference case demand). Each ELAST variable is bounded upward via virtual equation EQ_BNDELAS.

Role: Each elastic demand is expressed as the sum of these variables. In the objective function, these variables are used to bear the cost of demand losses as explained in Part I, Chapter 4.

Bounds: Direct bounding not available, indirectly by COM_VOC/COM_STEP
User constraints: Not available

5.12 VAR_FLO(r,v,t,p,c,s)

Definition: these variables stand for the individual commodity flows in and out of a process. If the process is not vintaged, the vintage index **v** is by convention set equal to the period index **t**.

Role: The flow variables are the fundamental quantities defining the detailed operation of a process. They are used to define the activity of a process (VAR_ACT) in a user chosen manner. They are also essential for expressing various constraints that balance the flows of a commodity, or that control the flexibility of processes.

Bounds: Can be directly bounded by FLO_BND
User constraints: Can be directly referred to by UC_FLO

5.13 VAR_IRE(r,v,t,p,c,s,ie)

Definition: the inter-regional exchange variable (i=IMPort, e=EXPort) that tracks import (**ie**=i) or export (**ie**=e) of a commodity between region **r** and other regions. The region(s) **r'** trading with **r** is (are) not specified via this variable, but rather via the process(es) **p** through which the import/export is accomplished. The topology set **top_ire(r,c,r',c',p)** of an exchange process indicates the (single) region **r'** with which region **r** is trading commodity **c** (which may have a different name **c'** in region **r')**. Each trade process may trade more than one commodity. Otherwise, VAR_IRE operates in a manner similar to VAR_FLO for conventional processes. An option exists for trading with an external region that is not modeled explicitly (exogenous trading). If the process is not vintaged, the vintage index **v** is by convention set equal to the period index **t**.

Role: the role of an IRE variable is to embody the amount of a commodity in or out of a trading process.

Bounds: Can be bounded by IRE_BND (directly for bilateral trade)
User constraints: Can be directly referred to by UC_IRE

5.14 VAR_NCAP(r,v,p)

Definition: the amount of new capacity (or what has traditionally been called "investment" in new capacity, or capacity build-up) at period **v**. As will be explained in Section 6.2.2, VAR_NCAP represents the total investment in technology **p** at period **v** only when ILED+TLIFE ≥ D(v), where D(v) is the period length. And, as discussed further in that Section, when ILED+TLIFE < D(v), the model assumes that the investment is repeated as many times as necessary within the period so that the life of the last repetition is beyond the end of period **v**. In this case VAR_NCAP represents the capacity level of the single investments. Figure 1 illustrates a case where the investment is made twice in period **v** (and some capacity still remains after period **v**). The average capacity in period **v** resulting from the investment VAR_NCAP(v) is less than VAR_NCAP(v), due to the delay ILED (it is equal to VAR_NCAP(v)* D(v)/TLIFE). The average capacity in period **v+1** due to VAR_NCAP(v) is also less than VAR_NCAP(v) because the end of life of the second round of investment occurs before the end of period **v+1**. These adjustments are made in every equation involving VAR_NCAP by the internal parameter COEF_CPT.

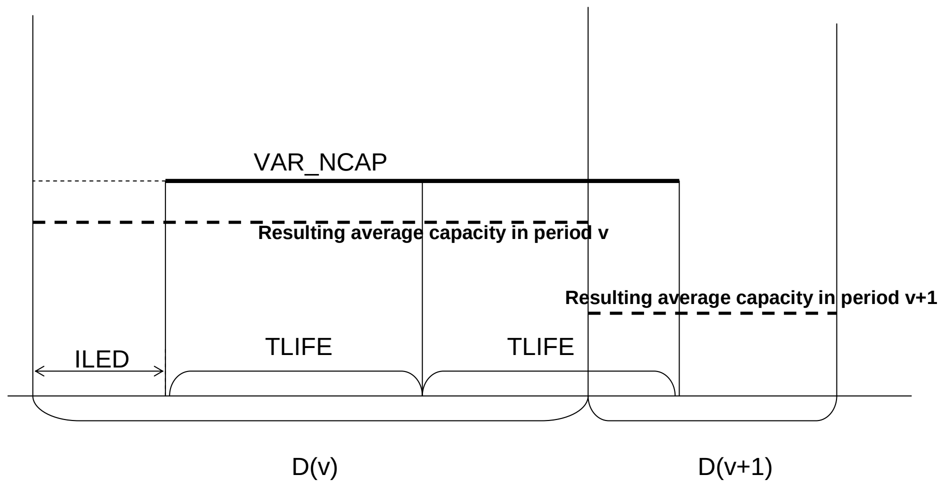


Figure 1: Example of a repeated investment in same period

Role: The new capacity (i.e. investment) variables are fundamental in defining the investment decisions, and many other quantities derived from it (for instance process capacities). They play a key role in the model structure and intervene in the majority of constraints. They are notably used in equations that define the conservation of capacity and those that tie the activity of a process to its capacity. The omnipresence of VAR_NACP is in part due to the fact that the VAR_CAP variable is not always defined in TIMES, by design. Note that residual capacity, or capacity in place prior to the initial model year, is handled as a constant in place of VAR_NCAP given by the input parameter NCAP_PASTI(y), which describes the investment made prior to the first period in the pastyear **y**.

Bounds: Can be directly bounded by NCAP_BND
User constraints: Can be directly referred to by UC_NCAP

5.15 VAR_OBJ(y_0) and related variables

Definition: equal to the objective function of the TIMES LP, i.e. the total cost of all regions, discounted to year $\mathbf{y_0}$.

Role: this is the quantity that is minimized by the TIMES optimizer.

Remark: The next 10 'variables' do not directly correspond to GAMS variables. They are used in the documentation (especially Section 6.2) as convenient intermediate placeholders that capture certain portions of the cost objective function. The reader is invited to look at Section 6.2 for detailed explanations on how these various costs enter the composition of the objective function. Most of these 'variables' are defined as reporting parameters that are made available to the VEDA-BE results analyser, as shown in Section 3.3.

5.15.1 VAR_OBJR(r, y_0)

Definition: equal to the sum of the various pieces of the total cost of region **r** discounted to year **y_0.**

Role: this is not a true variable in the GAMS code. It is used only as a convenient placeholder for writing the corresponding portion of the objective function in this documentation. It may also be reported in VEDA-BE.

5.15.2 INVCOST(r,y)

Definition: equal to the portion of the cost objective for year **y**, region **r**, that corresponds to investments.

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.3 INVTAXSUB(r,y)

Definition: equal to the portion of the cost objective for year **y**, region **r**, that corresponds to investment taxes and subsidies.

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.4 INVDECOM(r,y)

Definition: equal to the portion of the cost objective for year **y**, region **r**, that corresponds to capital costs linked to decommissioning of a process.

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.5 FIXCOST(r,y)

Definition: equal to the portion of the cost objective for year **y**, region ***r*,** that corresponds to fixed annual costs.

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.6 FIXTAXSUB(r,y)

Definition: equal to the portion of the cost objective for year **y**, region ***r*,** that corresponds to taxes and subsidies attached to fixed annual costs.

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.7 VARCOST(r,y)

Definition: equal to the portion of the cost objective for year **y**, region ***r*,** that corresponds to variable annual costs.

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.8 VARTAXSUB(r,y)

Definition: equal to the portion of the cost objective for year **y**, region ***r*,** that corresponds to variable annual taxes and subsidies.

Role: it is used mainly as a convenient place holder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.9 ELASTCOST(r,y)

Definition: equal to the portion of the cost objective for year **y**, region ***r*,** that corresponds to the cost incurred when demands are reduced due to their price elasticity.

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.15.10 LATEREVENUES(r,y)

Definition: equal to the portion of the cost objective for year **y**, region ***r***, that corresponds to certain late revenues from the recycling of materials from dismantled processes that occur after the end-of-horizon.

Role: this is not a true variable in the GAMS code. It is used only as a convenient placeholder for writing the corresponding portion of the objective function in this documentation. It may also be reported in VEDA-BE as a convenient replacement for the sum of the components of the total cost.

5.15.11 SALVAGE(r,y_0)

Definition: equal to the portion of the cost objective for region *r*, that corresponds to the salvage value of investments and other one-time costs. It is discounted to some base year y_0

Role: it is used mainly as a convenient placeholder for writing the corresponding portion of the objective function. It may also be reported in VEDA-BE.

5.16 VAR_RCAP(r,v,t,p)

Definition: this variable is used only for processes selected by the user as having early capacity retirements. For such processes, VAR_RCAP represents the amount of capacity of vintage **v** retired in period **t**.

Role: introduced for supporting bounds on the amount of retired capacity of process **p** and vintage **v** in period **t**.

Bounds: Can be directly bounded by RCAP_BND
User constraints: Not available

5.17 VAR_SCAP(r,v,t,p)

Definition: this variable is used only for processes selected by the user as having early capacity retirements. For such processes, VAR_SCAP represents the cumulative amount of capacity of vintage **v** retired in periods **tt** ≤ **t**.

Role: needed in several TIMES equations for adjusting the overall available capacity of process **p** at period **t** according to the amount of capacity already retired.

Bounds: Not directly available; indirectly by RCAP_BND / CAP_BND
User constraints: Not available

5.18 VAR_SIN/SOUT(r,v,t,p,c,s)

Definition: flow entering/leaving at period **t** a storage process **p**, storing commodity **c**. The process may be vintaged. If the process is not vintaged, the vintage index **v** is by convention set equal to the period index **t**. For storages between timeslices (**prc_stgtss**) and night-storage devices (**prc_nsttss**) the timeslice index s of the storage flows is determined by the timeslice resolution of the storage (e.g. DAYNITE for a day storage). For a storage operating between periods (**prc_stgips**), the storage flows are always on an annual level and hence the timeslice **s** is then always set to ANNUAL.

Role: to store some commodity so that it may be used in a time slice or period different from the one in which it was procured; enters the expressions for the storage constraints.

Bounds: Can be directly bounded by STGIN_BND/ STGOUT_BND
User constraints: Not directly available; indirectly by using auxiliary storage flows

5.19 VAR_UPS(r,v,t,p,s,l)

Definition: amount of off-line capacity (l='N'), started-up capacity (l='UP'), shut-down capacity (l='LO'), or efficiency losses due to partial loads (l='FX') in period **t** process **p** vintage index **v**.

Role: used for modeling capacity dispatching, start-up costs as well as partial load efficiencies, but only when requested so by the user.

Bounds: Not available
User constraints: Not directly available; in timeslice-dynamic constraints on-line capacity can be referred by UC_CAP, using the ONLINE modifier for CAP

5.20 Variables used in User Constraints

The remaining TIMES variables are all attached to user constraints. User constraints are quite flexible, and may involve any of the usual TIMES variables. Two variants of formulating user constraints exist. In the first case a LHS expression, containing expressions involving the different TIMES variables, are bounded by a RHS constant (given by the input parameter UC_RHS(R)(T)(S)). In the second case, the constant on the RHS is replaced by a variable. The bound UC_RHS(R)(T)(S) is then applied to this variable. In the latter case, the user constraints are always generated as strict equalities, while in the first case the equation sign of the user constraint is determined by the bound type.

- Case 1 (RHS constants): <LHS expression> ≤/=/≥ UC_RHS(R)(T)(S)
- Case 2 (UC variables): <LHS expression> = VAR_UC(R)(T)(S)

These user constraint variables are in fact redundant, but quite useful in providing streamlined expressions constraints (see Chapter 6), and allow for reporting the slack level of each UC. Morever, in the case of range constraints, they will reduce model size and the amount of input data. By setting the dollar control parameter VAR_UC to YES in the run-file, the variable based formulation is activated (second case). By default, the formulation without user constraint variables will be used, and only the marginals of the equations are reported.

Non-binding user constraints (intoduced for reporting purposes) can only be defined when the user constraint variables are used (i.e. VAR_UC == YES).

Each of the listed variables is related to a specific class of user constraint depending on whether the user constraint is created for each period, region, or time slice or only a subset of these indices. In addition, some user constraints are defined for pair of successive time periods (dynamic user constraint or growth constraint). Each variable has at least one index (representing the user constraint **uc_n** for which this variable is defined), and may have up to three additional indexes among **r**, **t**, and **s**.

5.20.1 VAR_UC(uc_n)

Variable representing the LHS expression of the user constraint EQE_UC(uc_n) summing over regions (**uc_r_sum**), periods (**uc_t_sum**) and timeslices (**uc_ts_sum**).

5.20.2 VAR_UCR(uc_n,r)

Variable representing the LHS expression of the user constraint EQE_UCR(r,uc_n) summing over periods (**uc_t_sum**) and timeslices (**uc_ts_sum**) and being generated for the regions specified in **uc_r_each**.

5.20.3 VAR_UCT(uc_n,t)

Variable representing the LHS expression of the user constraint EQE_UCT(uc_n,t) and the combined LHS–RHS expression of the user constraint EQE_UCSU(uc_n,t), summing over regions (**uc_r_sum**) and timeslices (**uc_ts_sum**) and being generated for the periods specified in **uc_t_each/uc_t_succ**.

5.20.4 VAR_UCRT(uc_n,r,t)

Variable representing the LHS expression of the user constraint EQE_UCRT(r,uc_n,t) and the combined LHS–RHS expression of the user constraint EQE_UCRSU(r,uc_n,t), summing over timeslices (**uc_ts_sum**) and being generated for the regions specified in **uc_r_each** and periods in **uc_t_each/uc_t_succ**.

5.20.5 VAR_UCTS(uc_n,t,s)

Variable representing the LHS expression of the user constraint EQE_UCTS(uc_n,t,s) and the combined LHS–RHS expression of the user constraint EQE_UCSUS(uc_n,t,s), summing over regions (**uc_r_sum**) and being generated for the periods specified in **uc_t_each/uc_t_succ** and timeslices in **uc_ts_each**.

5.20.6 VAR_UCRTS(uc_n,r,t,s)

Variable representing the LHS expression of the user constraint EQE_UCRTS(r,uc_n,t,s) and the combined LHS–RHS expression of the user constraint EQE_UCRSUS(r,uc_n,t,s), being generated for the regions specified in **uc_r_each**, the periods in **uc_t_each/uc_t_succ** and the timeslices in **uc_ts_each**.

6 Equations

This chapter is divided into four sections: the first section describes the main notational conventions adopted in writing the mathematical expressions of the entire chapter. The next two sections treat respectively the TIMES objective function and the standard linear constraints of the model. The fourth section is devoted to the facility for defining various kinds of user constraints. Additional constraints and objective function additions that are required for the Climate Module, Damage Cost and Endogenous Technology Learning options are described in Appendices A, B and C, respectively.

Each equation has a unique name and is described in a separate subsection. The equations are listed in alphabetical order in each section. Each subsection contains successively the name, list of indices, and type of the equation, the related variables and other equations, the purpose of the equation, any particular remarks applying to it, and finally the mathematical expression of the constraint or objective function.

The mathematical formulation of an equation starts with the name of the equation in the format: $EQ\_XXX_{i,j,k,l}$, where *XXX* is a unique equation identifier, and *i,j,k,..*, are the *equation indexes*, among those described in chapter 2. Some equation names also include an index *l* controlling the sense of the equation. Next to the equation name is a *logical condition* that the equation indexes must satisfy. That condition constitutes the *domain of definition* of the equation. It is useful to remember that the equation is created in multiple instances, one for each combination of the equation indexes that satisfies the logical condition, and that each index in the equation's index list remains *fixed* in the expressions constituting each instance of the equation.

6.1 Notational conventions

We use the following mathematical symbols for the mathematical expressions and relations constituting the equations:

The conditions that apply to each equation are mathematically expressed using the $\ni$ symbol (meaning "such that" or "only when"), followed by a logical expression involving the usual logic operators: $\wedge$ (AND), $\vee$ (OR), and NOT.

Within the mathematical expressions of the constraints, we use the usual symbols for the arithmetic operators ($+,-,\times,/,\Sigma,$ etc).

However, in order to improve the writing and legibility of all expressions, we use some simplifications of the usual mathematical notation concerning the use of multiple indexes, which we describe in the next two subsections.

6.1.1 Notation for summations

When an expression *A(i,j,k,…)* is summed, the summation must specify the range over which the indexes are allowed to run. Our notational conventions are as follows:

When a single index j runs over a one-dimensional set A, the usual notation is used, as in: $\sum_{j \in A} Expression_j$ where *A* is a single dimensional set.

When a summation must be done over a subset of a multi-dimensional set, we use a simplified notation where some of the running indexes are omitted, if they are not active for this summation.

Example: consider the 3-dimensional set *top* consisting of all quadruples *{r,p,c,io}* such that process *p* in region *r*, has a flow of commodity *c* with orientation *io* (see table 3 of chapter 2). If is it desired to sum an expression $A_{r,p,c,io}$ over all commodities c, keeping the region (r), process (p) and orientation (io) fixed respectively at r_1, p_1 and 'IN', we will write, by a slight abuse of notation: $\sum_{c\in top(r_1,p_1,'IN')} A(r_1,p_1,c,'IN')$, or even more simply:

$\sum_{c\in top} A(r_1,p_1,c,'IN')$, if the context is unambiguous. Either of these notations clearly indicates that *r, p* and *io* are fixed and that the only active running index is *c*.

(The traditional mathematical notation would have been: $\sum_{\{r_1,p_1,c,'IN'\}\in top} A(r_1,p,c_1,'IN')$, but this may have hidden the fact that *c* is the only running index active in the sum).

6.1.2 Notation for logical conditions

We use similar simplifying notation in writing the logical conditions of each equation. A logical condition usually expresses that some parameter exists (i.e. has been given a value by the user), and/or that some indexes are restricted to certain subsets.

A typical example of the former would be written as: $\ni ACTBND_{r,t,p,s,bd}$, which reads: "the user has defined an activity bound for process *p* in region *r*, time-period *t*, timeslice *s* and sense *bd*". The indexes may sometimes be omitted, when they are the same as those attached to the equation name.

A typical example of the latter is the first condition for equation $EQ\_ACTFLO_{r,v,t,p,s}$ (see section 6.3.4), which we write simply as: **rtp_vintyr**, which is short for: $\{r,v,t,p\}\in$ **rtp_vintyr**, with the meaning that "some capacity of process *p* in region *r*, created at period *v*, exists at period *t*". Again here, the indices have been omitted from the notation since they are already listed as indices of the equation name.

6.1.3 Using Indicator functions in arithmetic expressions

There are situations where an expression A is either equal to B or to C, depending on whether a certain condition holds or not, i.e.:

$$A = B\ if\ Cond$$

$$A = C\ if\ NOT\ Cond$$

This may also be written as:

$$A = B\times(Cond) + C\times(NOT\ Cond)$$

where it is understood that the notation (Cond) is the *indicator function* of the logical condition, i.e. (Cond)=1 if Cond holds, and 0 if not.

This notation often makes equations more legible and compact. A good example appears in EQ_CAPACT.

6.2 Objective function EQ_OBJ

Equation EQ_OBJ
Indices: region (r); state of the world (w); process (p); time-slice (s); and perhaps others ...
Type: = Non Binding (MIN)

Related Variables: All

Purpose: the objective function is the criterion that is minimized by the TIMES model. It represents the total discounted cost of the entire, possibly multi-regional system over the selected planning horizon. It is also equal to the negative of the discounted total surplus (plus a constant), as discussed in PART I, chapters 3 and 4.

6.2.1 Introduction and notation

The TIMES objective function includes a number of innovations compared to those of more traditional energy models such as MARKAL, EFOM, MESSAGE, etc. The main design choices are as follows:

- The objective function may be thought of as the discounted sum of *net annual costs* (i.e. costs minus revenues), as opposed to *net period costs*[39]. Note that some costs and revenues are incurred after the end of horizon (EOH). This is the case for instance for some investment payments and more frequently for payments and revenues attached to decommissioning activities. The past investments (made before the first year of the horizon) may also have payments within horizon years (and even after EOH!) These are also reflected in the objective function. However, it should be clear that such payments are shown in OBJ only for reporting purposes, since such payments are entirely *sunk*, i.e. they are not affected by the model's decisions.
- The model uses a general discount rate $d(y)$ (year dependent), as well as technology specific discount rates $d_s(t)$ (period dependent). The former is used to: a) discount fixed and variable operating costs, and b) discount investment cost payments from the point of time when the investment actually occurs to the base year chosen for the computation of the present value of the total system cost. The latter are used only to calculate the annual payments resulting from a lump-sum

[39] The actual implementation of OBJ in the GAMS program is different from the one described in the documentation, since the annualizing of the various cost components is not performed in the GAMS code of the OBJ equation, but rather in the reporting section of the program, for improved code performance. However, despite the simplification, the GAMS code results in an objective function that is fully equivalent to the one in this documentation.

investment in some year. Thus, the only place where $d_s(t)$ intervenes is to compute the Capital Recovery Factors *(CRF)* discussed further down.

For convenience, we summarize below the notation which is more especially used in the objective function formulation (see Section 6.1 for general notes on the notation) .

6.2.1.1 Notation relative to time

MILESTONEYEARS: the set of all milestone years (by convention: middle years, see below ***M(t)***)

PASTYEARS: Set of years prior to start of horizon, for which there is a past investment (*after* interpolation of user data).

MODELYEARS: any year within the model's horizon

FUTUREYEARS: set of years posterior to EOH

YEARS set of years before during and after planning horizon

t any member of ***MILESTONEYEARS*** or ***PASTYEARS***. By convention, a period ***t*** is represented by its middle year (see below ***M(t)***). This convention can be changed without altering the expressions in this document.

B(t) : the first year of the period represented by ***t***

E(t) : the last year of the period represented by ***t***

D(t) : the number of years in period ***t*** . By default, ***D(t)=1*** for all past years. Thus, ***D(t)=E(t)–B(t)+1***

M(t): the "middle" year or milestone year of period ***t***. Since period ***n*** may have an even or an odd number of years, ***M(t)*** is not always exactly centered at the middle of the period. It is defined as follows: ***M(t) = [B(t)+(D(t)–1)/2],*** where ***[x]*** indicates the largest integer less than or equal to ***x***. For example, period from 2011 to 2020 includes 10 years, and its "middle year" is [2011+4.5] or 2015 (slightly left of the middle), whereas the period from 2001 to 2015 has 15 years, and its "middle year" is : [2001+7] or 2008 (i.e. the true middle in this example)

y : running year, ranging over ***MODELYEARS***, from B_0 to ***EOH***.

k : dummy running index of any year, even outside horizon

v: running index for a year, used when it represents a vintage year for some investment.

v(p) vintage of process ***p*** (defined only if ***p*** is vintaged)

B_0 : initial year (the single year of first period of the model run)

EOH : Last year in horizon for a given model run.

Similarly, by a slight abuse of notation, the above entities are extended as follows, when the argument is a particular year, rather than a model year:

B(y) : first year of the period containing year ***y*** (instead of ***B(T(y))***)

T(y) the milestone year of the period containing year ***y*** (same as ***M(y)*** in our present convention)

M(y) : "middle year" of the period containing year ***y*** (instead of ***M(T(y))***)

D(y) : number of years of the period containing year ***y*** (instead of ***D(T(y))***

6.2.1.2 Other notation

d(y) :	general (social) discount rate (time dependent, although not shown in notation)
r(y) :	general discount factor: $r(y)=1/(1+d(y))$ (time dependent, although not shown in notation)
$d_s(t)$:	technology specific discount rate (model year dependent)
$r_s(t)$:	technology specific discount factor: $r_s(t)=1/(1+d_s(t))$
DISC(y,z):	Value, discounted to the beginning of year ***z***, of a $1 payment made at beginning of year ***y***, using **general** discount factor. $DISC(y,z) = \Pi_{u=z \text{ to } y-1}\ r(u)$
$CRF_s(t)$:	Capital recovery factor, using a (technology specific) discount rate and an economic life appropriate to the payment being considered. This quantity is used to replace an investment cost by a series of annual payments spread over some span of time $CRF_s=\{1-r_s(t)\}/\{1-r_s(t)^{ELIFE}\}$[40]. Note that a *CRF* using the general discount rate is also defined and used in the SALVAGE portion of the objective function.
OBJ(z):	Total system cost, discounted to the beginning of year ***z***
INDIC(x):	***1*** if logical expression ***x*** is true, ***0*** if not
$\langle E \rangle$	is the smallest integer larger than of equal to *E*

6.2.1.3 Reminder of some technology attribute names (each indexed by *t*)

TLIFE	Technical life of a technology
ELIFE	Economic life of a technology, i.e. period over which investment payments are spread (default = ***TLIFE***)
DLAG	Lag after end of technical life, after which decommissioning may start
DLIFE	Duration of decommissioning for processes with ***ILED>0, (otherwise =1)***
DELIF	Economic life for decommissioning purposes (default ***DLIFE***).
ILED	Lead-time for the construction of a process. *TLIFE* starts *after* the end of *ILED*
$ILED_{Min}$	$=Min\ \{1/10 * D(t),\ 1/10 * TLIFE.\}$ This threshold serves to distinguish small from large projects; it triggers a different treatment of investment timing.

6.2.1.4 Discounting options

There are alternate discounting methods in TIMES. The default method is to assume that all payments occur at the beginning of some year. Alternate methods (activated by a switch, see PART III) assume that investments are incurred at the beginning of some year, but that all annual (or annualized) payments occur at the middle or at the end of the corresponding year. Section 0 explains the different methods.

[40] This is the default definition adopted for *CRF*, corresponding to beginning-of-year discounting. For other discounting options, see Section 0.

6.2.1.5 Components of the Objective function

The objective function is the sum of all regional objectives, all of them discounted to the same user-selected base year, as shown in equation (A) below

EQ_OBJ(z) $\quad\ni z \in ALLYEARS$

$$\boxed{VAR\_OBJ(z) = \sum_{r \in REG} REG\_OBJ(z,r)} \qquad (A)$$

Each regional objective *OBJ(z,r)* is decomposed into the sum of nine components, to facilitate exposition, as per expression (B) below.

EQ_OBJ(z,r) $\quad\ni z \in ALLYEARS,\ r \in REG$

$$\boxed{\begin{array}{l} REG\_OBJ(z,r) = \displaystyle\sum_{y \in (-\infty,+\infty)} DISC(y,z) \times \begin{Bmatrix} INVCOST(y) + INVTAXSUB(y) + INVDECOM(y) + \\ FIXCOST(y) + FIXTAXSUB(y) + SURVCOST(y) + \\ VARCOST(y) + VARTAXSUB(y) + ELASTCOST(y) - \\ LATEREVENUES(y) \end{Bmatrix} \\ \qquad\qquad\qquad\qquad -SALVAGE(z) \end{array}} \qquad (B)$$

The regional index *r* is omitted from the nine components for simplicity of notation.

The first and second terms are linked to investment costs. The third term is linked to decommissioning capital costs, the fourth and fifth terms to fixed annual costs, the seventh and eighth terms to all variable costs (costs proportional to some activity), and the ninth to demand loss costs. The tenth cost (actually a revenue) accounts for commodity recycling occurring after *EOH*, and the eleventh term is the salvage value of all capital costs of technologies whose life extends beyond *EOH*. The 11 components are presented in the nine subsections 6.2.2 to 6.2.10.

6.2.2 Investment costs: INVCOST(y)

This subsection presents the components of the objective function related to investment costs, which occur in the year an investment is decided and/or during the construction lead-time of a facility.

Remarks

a) The investment cost specified by using the input attribute NCAP_COST should be the overnight investment cost (excluding any interests paid during construction) whenever the construction lead time is explicitly modeled (i.e. cases 2 are used, see below). In such a case, the interests during construction are endogenously calculated by the model itself, as will be apparent in the sequel. If no lead-time is specified (and thus cases 1 are used), the full cost of investments should be used (including interests during construction, if any)[41].

b) Each individual investment physically occurring in year *k*, results in a *stream of annual payments* spread over several years in the future. The stream starts in year *k* and covers years *k, k+1, …, k+ELIFE-1,* where *ELIFE* is the economic life of the technology. Each yearly payment is equal to a fraction *CRF* of the investment cost (*CRF* = Capital Recovery Factor). Note that if the technology discount rate is equal to the general discount rate, then the stream of *ELIFE* yearly payments is equivalent to a single payment of the whole investment cost located at year *k*, inasmuch as both have the same discounted present value. If however the technology's discount rate is chosen different from the general one, then the stream of payments has a different present value than the lump sum at year *k*. It is the user's responsibility to choose technology dependent discount rates, and therefore to decide to alter the effective value of investment costs.

c) In addition to spreading the payments resulting from investment costs, a major TIMES refinement is that the physical investment itself does not occur in a single year, but rather as a series of annual increments. For instance, if the model invests 3 GW of electric capacity in a period extending from 2011 to 2020, the physical capacity increase may be delayed and/or may be spread over several years. The exact way the delaying and spreading are effected depends on several conditions, which are specified further down as four separate cases, and which are functions both of the nature of the technology and of the length of the period in which the investment takes place relative to the technology's technical life. The spreading of investments and the spreading of payments described in the previous paragraph help guarantee a smooth trajectory for most investment payments, a more realistic representation than what happens in other models. The Case 1.a example given below shows a case where the physical investment is spread over four years, and each increment's capital payments are further spread over 3 years.

d) The above two remarks entail that payments of investment costs may well extend beyond the horizon. We shall also see that some investment payments occur in years prior to the beginning of the planning horizon (cases 1 only).

e) Taxes and subsidies on investments are treated exactly as investment costs in the objective function.

f) Since the model has the capability to represent *sunk* materials and energy carriers (i.e. those embedded in a technology at construction time, such as the uranium core of a nuclear reactor, or the steel imbedded in a car), these sunk commodities have an impact on cost. Two possibilities exist: if the material is one whose production is explicitly modeled in the RES, then there is no need to indicate the cost corresponding to the sunk material, which will be implicitly accounted for by the model just like any other flow. If on the other hand the material is not specifically modeled in the RES, then the cost of the sunk material should be included in the technology's investment cost, and will then be handled exactly as investment costs.

[41] Ideally, it would be desirable that cases 1 be used only for those investments that have no lead time (and thus no interest during construction). However, if cases 1 are employed even for projects with significant IDC's, these should have their IDC included in the investment cost.

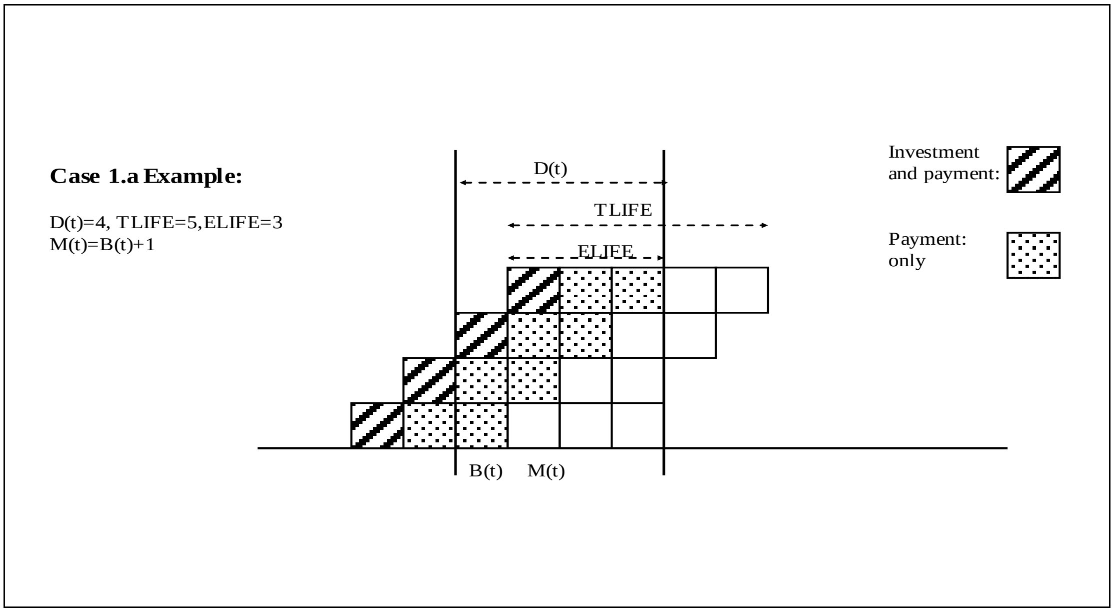
The four investment cases

As mentioned above, the timing of the various types of payments and revenues is made as realistic and as smooth as possible. All investment decisions result in increments and/or decrements in the capacity of a process, at various times. These increments or decrements may occur, in some cases, in one large lump, for instance in the case of a large project (hydroelectric plant, aluminum plant, etc.), and, in other cases, in small additions or subtractions to capacity (e.g. buying or retiring cars, or heating devices). Depending on which case is considered, the assumption regarding the corresponding streams of payments (or revenues) differs markedly. Therefore, the distinction between small and large projects (called cases 1 and 2 below) will be crucial for writing the capital cost components of the objective function. A second distinction comes from the relative length of a project's technical life vs. that of the period when the investment occurs. Namely, if the life of an investment is less than the length of the period, then it is clear that the investment must be repeated all along the period. This is not so when the technical life extends beyond the period's end. Altogether, these two distinctions result in four mutually exclusive cases, each of which is treated separately. In what follows, we present the mathematical expression for the INVCOST component and one graphical example for each case.

Case 1.a If $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED_t \ge D(t)$

(Small divisible projects, non-repetitive, progressive investment in period)

Here, we make what appears to be the most natural assumption, i.e. that the investment occurs in small yearly increments spread linearly over *D(t)* years. Precisely, the capacity additions start at year *M(t)-D(t)+1*, and end at year M*(t)*, which means that payments start earlier than the beginning of the period, and end at the middle of the period, see example. This seems a more realistic compromise than starting the payments at the beginning of the period and stopping them at the end, since that would mean that during the whole period, the paid for capacity would actually not be sufficient to cover the capacity selected by the model for that period.



EQ_INVCOST(y)

deals with linear investment buildup, over a span equal to period length, ending at middle of period

$$INVCOST(y) = \sum_{t \in MILESTONE \cup PASTYRS} INDIC(1.a) \times \sum_{v=Max\{M(t)-D(t)+1,\, y-ELIFE_t+1\}}^{Min\{M(t),y\}} \left(\frac{VAR\_NCAP_t}{D(t)} + NCAP\_PASTI_t \right)$$

$$\times CRF_s \times NCAP\_COST_v$$

ensures that payments stop after ELIFE

$Useful\ \ Range\ \ for\ \ y:$

$$\{M(t)-D(t)+1\ ,\ M(t)+ELIFE_t-1\}$$

(I.1.a)

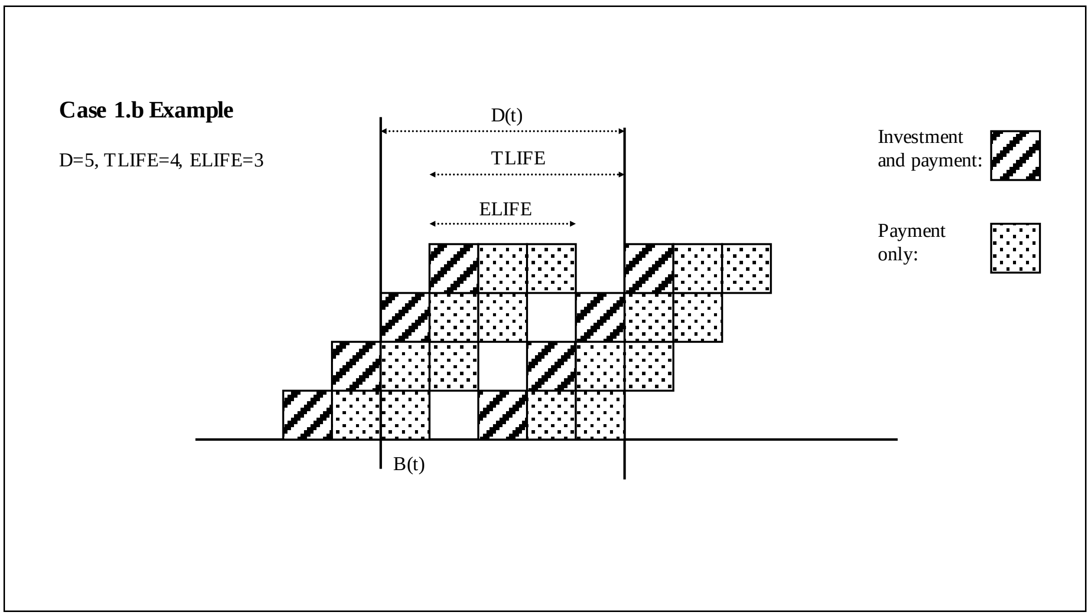
Comments: The summand represents the payment effected in year *y*, due to the investment increment that occurred in year *v* (recall that investment payments are spread over *ELIFE*). The summand consists of three factors: the first is the amount of investment in year *v*, the second is the capital recovery factor, and the third is the unit investment cost.

The outer summation is over all periods (note that periods later than *T(y)* are relevant, because when *y* falls near the end of a period, the next period's investment may have already started). The inner summation is over a span of *D(t)* centered at *B(t)*, but truncated at year *y*. Also, the lower summation bound ensures that an investment increment which occurred in year *v* has a payment in year *y* only if *y* and *v* are less than *ELIFE* years apart.

Case 1.b if $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED < D(t)$

Small projects, repeated investments in period

Note that in this case the investment is repeated as many times as necessary to cover the period length (see figure). In this case, the assumption that the investment is spread over *D(t)* years is not realistic. It is much more natural to spread the investment over the technical life of the process being invested in, because this ensures a smooth, constant stream of small investments during the whole period (any other choice of the time span over which investment is spread, would lead to an uneven stream of incremental investments). The number of re-investments in the period is called *C*, and is easily computed so as to cover the entire period. As a result of this discussion, the first investment cycle starts at year $\langle B(t) - TLIFE_t / 2 \rangle$ (meaning the smallest integer not less than the operand), and ends *TLIFE* years later, when the second cycle starts, etc, as many times as necessary to cover the entire period. The last cycle extends over the next period(s), and that is taken into account in the capacity transfer equations of the model. As before, each capacity increment results in a stream of *ELIFE* payments at years *v*, *v*+1, etc.



$$INVCOST(y) =$$

$$\sum_{t \in MILESTONE} INDIC(1.b) \times \sum_{v = Max\{\langle B(t) - TLIFE_t/2 \rangle, y - ELIFE_t + 1}^{Min\{y, \langle B(t) - TLIFE_t/2 \rangle + C \times TLIFE_t - 1\}} \frac{VAR\_NCAP_t}{TLIFE_t} \times CRF_s \times NCAP\_COST_v$$

Relevant range for *y*:

$$\left\{ \langle B(t) - TLIFE_t / 2 \rangle , \langle B(t) - TLIFE_t / 2 \rangle + C \times TLIFE_t + ELIFE_t - 2 \right\}$$

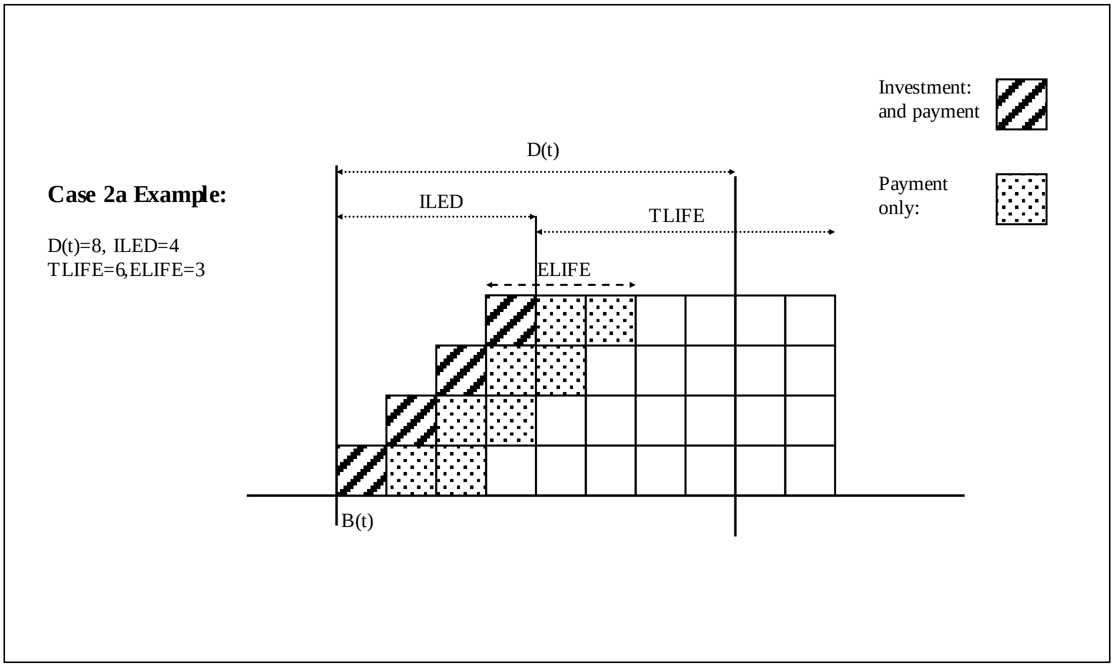
(I.1.b)

Comments: the expression is similar to that in case **1.a.**, except that i) the investment is spread over the technical life rather than the period length, and ii) the investment cycle is repeated more than once.

Case 2.a: $ILED_t > ILED_{Min,t}$ and $ILED_t + TLIFE_t \geq D(t)$

(Large, indivisible projects, unrepeated investment in period)

Here, it is assumed that construction is spread over the lead-time (a very realistic assumption for large projects), and capacity becomes available at the end of the lead time, **in a lump quantity** (see figure).



deals with linear investment buildup, over a span of ILED, starting at beginning of period

$$INVCOST(y) = \sum_{\substack{t \in MILESTONEYEARS \\ t \leq T(y)}} INDIC(2.a) \times \sum_{k=Max\{B(t),y-ELIFE_t+1\}}^{Min(B(t)+ILED_t-1,y)} \left(\frac{VAR\_NCAP_t}{ILED_t} \right) \times CRF_s \times NCAP\_COST_{B(t)+ILED_t}$$

$$+ \sum_{t \in PASTYEARS} INDIC(2.a) \times \sum_{k=Max\{t-ILED_t,y-ELIFE_t+1\}}^{Min(t-1,y)} \left(\frac{NCAP\_PASTI_t}{ILED_t} \right) \times CRF_s \times NCAP\_COST_t$$

Useful Range for y:

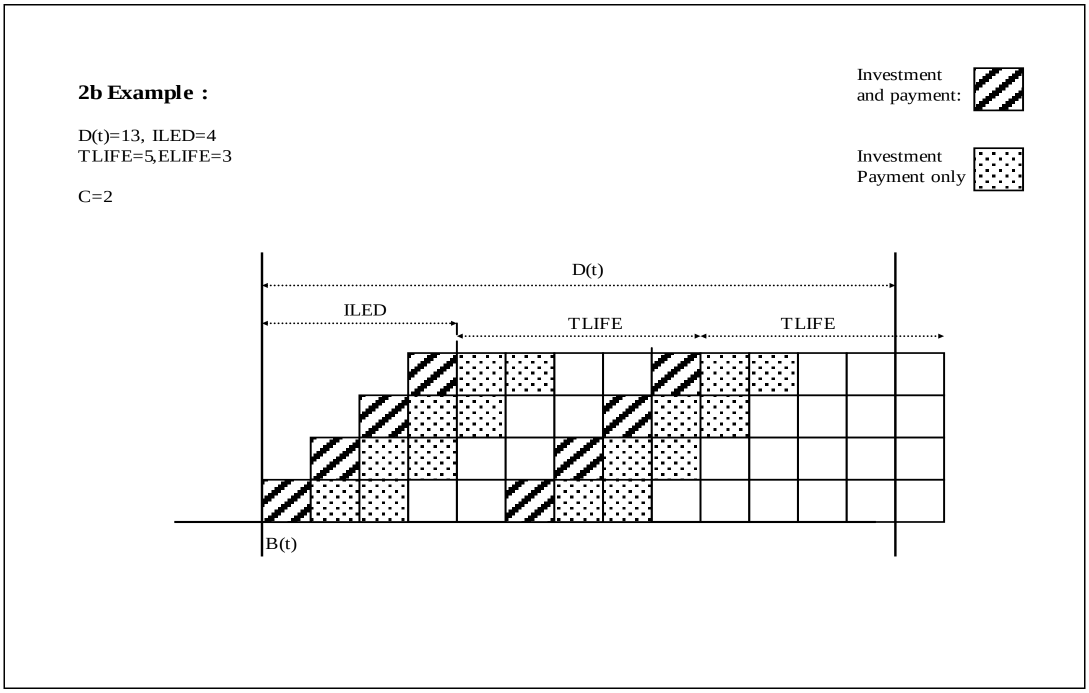
$\{B(t),\ B(t) + ILED_t + ELIFE_t - 2\}$

(I.2.a)

Comment: the main difference with case I.1.a) is that the investment's construction starts at year *B(t)* and ends at year $B(t)+ILED_t-1$ (see example). As before, payments for each year's construction spread over *ELIFE* years.

Case 2.b: $ILED > ILED_{Min,t}$ and $TLIFE_t + ILED_t < D(t)$

(Large, indivisible Projects, repeated investments in period)



This case is similar to case I.2.a, but the investment is repeated more than once over the period, each cycle being *TLIFE* years long. As in case I.2.a, each construction is spread over one lead time, *ILED*. In this case, the exact pattern of yearly investments is complex, so that we have to use an algorithm instead of a closed form summation.

<u>ALGORITHM</u> (Output: the vector of payments $P_t(y)$ at each year y, due to VAR_NCAP_t)

<u>Step 0:</u> Initialization (*NI(u)* represents the amount of new investment made in year *u*)

$$NI_t(u) := 0 \qquad \forall B(t) \le u \le B(t) + ILED_t + (C-1) \times TLIFE_t - 1$$

<u>Step 1:</u> Compute number of repetitions of investment

$$C = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

Step 2: for each year *u* in range:

$$B(t) \le u \le B(t) + ILED_t + (C-1) \cdot TLIFE_t - 1$$

Compute:

$$\begin{aligned}
&For\ I = 1\ to\, C \\
&\qquad For\ u = B(t) + (I-1) \cdot TLIFE_t \quad to \quad B(t) + (I-1) \cdot TLIFE_t + ILED_t - 1 \\
&\qquad\qquad NI_t(u) := NI_t(u) + \frac{NCAP_COST_{B(t)+(I-1)\times TLIFE_t + ILED_t}}{ILED_t} \\
&\qquad Next\ u \\
&Next\ I
\end{aligned}$$

Step 3: Compute payments incurred in year *y*, and resulting from variable VAR_NCAP_t
For each *y* in range:

$$B(t) \le y \le B(t) + (C-1) \cdot TLIFE_t + ILED_t + ELIFE_t - 2 \tag{I.2.b}$$

Compute:

$$P_t(y) = \sum_{u=Max\{B(t),\, y-ELIFE_t+1\}}^{y} NI_t(u) \times VAR\_NCAP_t \times CRF_s$$

END ALGORITHM

$$INVCOST(y) = \sum_{t \in MILESTONES,\ t \le T(y)} INDIC(2.b) \times P_t(y)$$

6.2.3 Taxes and subsidies on investments

We assume that taxes/subsidies on investments occur at precisely the same time as the investment. Therefore, the expressions *INVTAXSUB(y)* for taxes/subsidies are identical to those for investment costs, with *NCAP_COST* replaced by: *(NCAP_ITAX – NCAP_ISUB).*

6.2.4 Decommissioning (dismantling) capital costs: *INVDECOM(y)*

Remarks

a) Decommissioning physically occurs after the end-of-life of the investment, and may be delayed by an optional lag period *DLAG* (e.g. a "cooling off" of the process before dismantling may take place). The decommissioning costs follow the same patterns and rules as those for investment costs. In particular, the same four cases that were defined for investment costs are still applicable.

b) The same principles preside over the timing of payments of decommissioning costs as were defined for investment costs, namely, the decomposition of payments into a stream of payments extending over the economic life of decommissioning, *DELIF*.

c) At decommissioning time, the recuperation of embedded materials is allowed by the model. This is treated as explained for investment costs, i.e. either as an explicit commodity flow, or as a credit (revenue) subtracted *by the user* from the decommissioning cost.

g) Decommissioning activities may also receive taxes or subsidies which are proportional to the corresponding decommissioning cost.

$$EQ\_COSTDECOM(y) \quad \ni y \in ALLYEARS$$

Case 1.a) If $ILED_t \leq ILED_{Min,t}$ and $TLIFE_t + ILED_t \geq D(t)$

(Small divisible projects, non-repetitive, progressive investment in period)

In this case, decommissioning occurs exactly *TLIFE+DLAG* years after investment. For small projects (cases **1.a** and **1.b**), it is assumed that decommissioning takes exactly one year, and also that its cost is paid that same year (this is the same as saying that *DLIFE=DELIF=1*). Any user-defined DLIFE/DELIF is in this case thus ignored. This is a normal assumption for small projects. As shown in the example below, also payments made at year *y* come from investments made at period *T(y)* or earlier. Hence the summation stops at *T(y).*

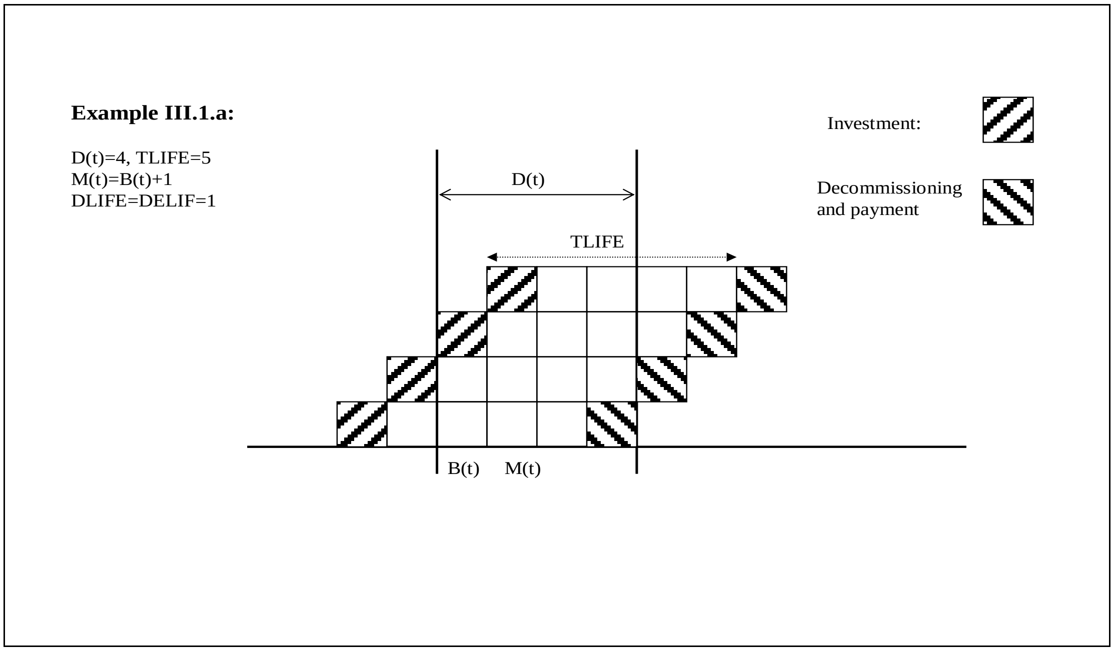
$$INVDECOM(y) =$$

$$\sum_{\substack{t \in MILESTONES \cup PASTYEARS \\ t \leq T(y)}} INDIC(1.a) \times \left(\frac{VAR\_NCAP_t}{D(t)} + NCAP\_PASTI_t \right) \times NCAP\_DCOST_{y-TLIFE_t}$$

$$\times \begin{cases} 1 & if \quad M(t) - D(t) + 1 + TLIFE_t + DLAG_t \leq y \leq M(t) + TLIFE_t + DLAG_t \\ 0 & otherwise \end{cases}$$

(III.1.a)

Comment: Note that the cost attribute is indexed at the year when the investment started to operate. We have adopted this convention throughout the objective function.



Case 1.b) if $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED < D(t)$

(Small projects, repeated investments in period)

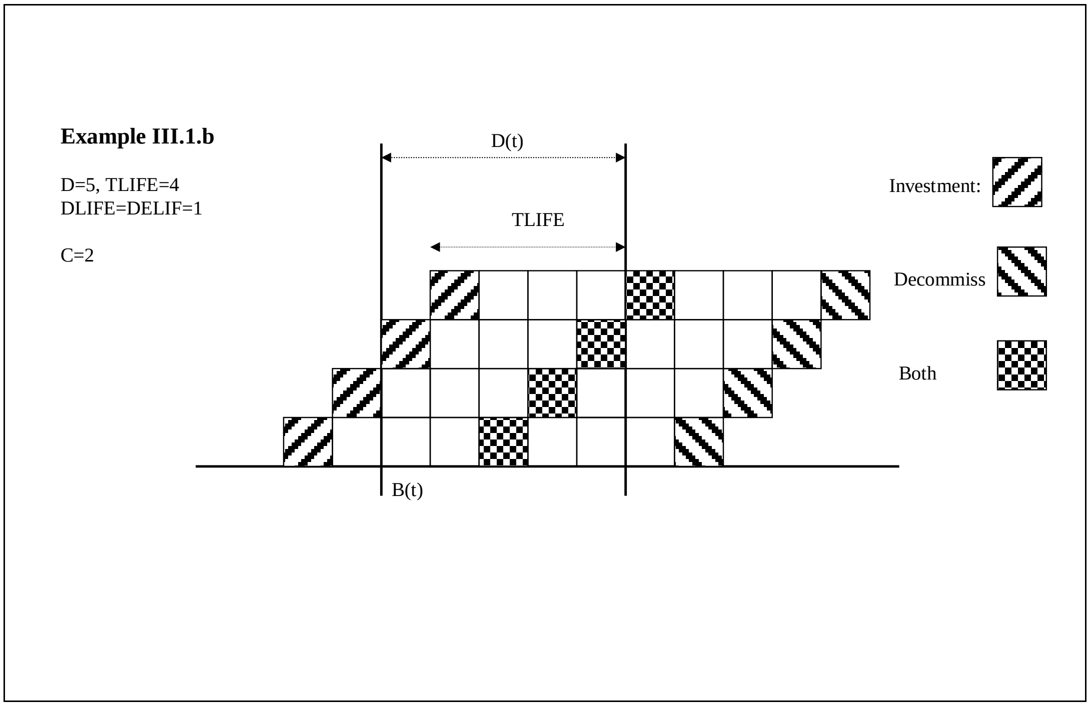
This cost expression is similar to I.1.b, but with payments shifted to the right by *TLIFE* (see example). The inner summation disappears because of the assumption that *DELIF=1*. Note also that past investments have no effect in this case, because this case does not arise *when D(t)=1,* which is always the case for past periods.

$$INVDECOM(y) = \sum_{\substack{t \in MILESTONES \\ t \le T(y)}} INDIC(1.b) \times \left(\frac{VAR\_NCAP_t}{TLIFE_t} \right) \times NCAP\_DCOST_{y-TLIFE_t}$$

$$\times \begin{cases} 1 \; if \;\; B(t) + \left\lceil \dfrac{TLIFE_t}{2} \right\rceil \le y \le B(t) + \left\lceil \dfrac{TLIFE_t}{2} \right\rceil + C \cdot TLIFE_t - 1 \\ 0 \quad otherwise \end{cases}$$

$$where \; C = \left\langle \frac{D(t)}{TLIFE_t} \right\rangle$$

(III.1.b)



Case 2.a: $ILED_t > ILED_{Min,t}$ and $ILED_t + TLIFE_t \geq D(t)$
(Large, indivisible projects, unrepeated investment in period)

In this situation, it is assumed that decommissioning of the plant occurs over a period of time called *DLIFE*, starting after the end of the technical process life *plus a time DLAG* (see example). *DLAG* is needed e.g. for a reactor to "cool down" or for any other reason. Furthermore, the payments are now spread over *DELIF,* which may be larger than one year.

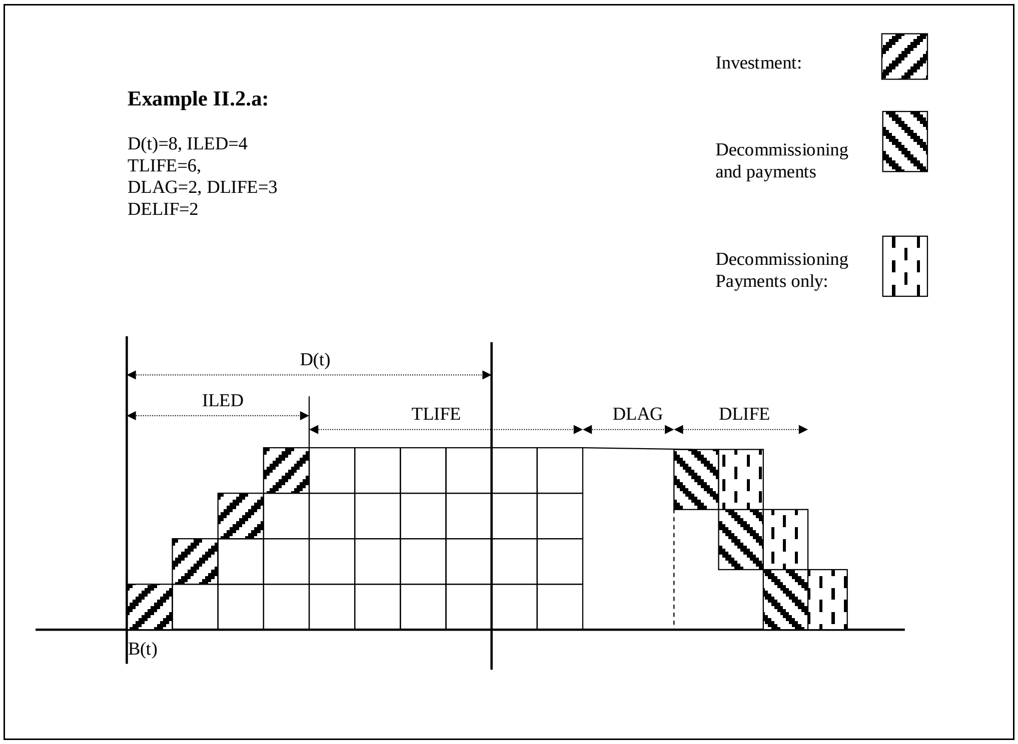
$$INVDECOM(y) =$$

$$\sum_{\substack{t \in MILESTONES \\ t \leq T(y)}} INDIC(2.a) \times \sum_{k=Max\{B(t)+ILED_t+TLIFE_t+DLAG_t,\, y-DELIF_t+1\}}^{Min\{y,B(t)+ILED_t+TLIFE_t+DLAG_t+DLIFE_t-1\}} \left(\frac{VAR\_NCAP_t}{DLIFE_t} \right) \times CRF_s \times NCAP\_DCOST_{B(t)+ILED_t}$$

$$+ \sum_{t \in PASTYEARS} INDIC(2.a) \times \sum_{k=Max\{t+TLIFE_t+DLAG_t,\, y-DELIF_t+1\}}^{Min\{y,t+TLIFE_t+DLAG_t+DLIFE_t-1\}} \left(\frac{NCAP\_PASTI_t}{DLIFE_t} \right) \times CRF_s \times NCAP\_DCOST_t$$

(III.2.a)

$Useful\ \ Range\ \ for\ \ y:$
$\{B(t) + ILED_t + TLIFE_t + DLAG_t - 1,\ same + DELIF_t - 1\}$



Case 2.b: $ILED_t > ILED_{Min,t}$ and $TLIFE_t + ILED_t < D(t)$
(Big projects, repeated investments in period)

Here too, the decommissioning takes place over *DLIFE*, but now, contrary to case 2.a, the process is repeated more than once in the period. The last investment has life extending over following periods, as in all similar cases. The resulting stream of yearly payments is complex, and therefore, we are forced to use an algorithm rather than a closed form summation. See also example below.

ALGORITHM (apply to each t such that $t \leq T(y)$)

Step 0: Initialization

$$P_t(y) := 0 \quad \forall \; B(t) + ILED_t + TLIFE_t + DLAG_t \leq y \leq same + (C-1) \times TLIFE_t + DLIFE_t + DELIF_t - 2$$

Where:

$$C = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

Step 1: Compute payment vector

$$
\begin{aligned}
&For\ \ I=1\ \ to\ \ C\\
&\quad For\ \ J=1\ \ to\ \ DLIFE_t\\
&\quad\quad For\ \ L=1\ to\ DELIF_t\\
&\quad\quad P_t\left(B(t)+ILED_t+I\times TLIFE_t+DLAG_t+J+L-2\right):=\\
&\quad\quad same+\frac{NCAP\_DCOST_{B(t)+ILED_t+(I-1)\times TLIFE_t}}{DLIFE_t}\\
&\quad\quad Next\ \ L\\
&\quad Next\ \ J\\
&Next\ \ I
\end{aligned}
$$

END ALGORITHM

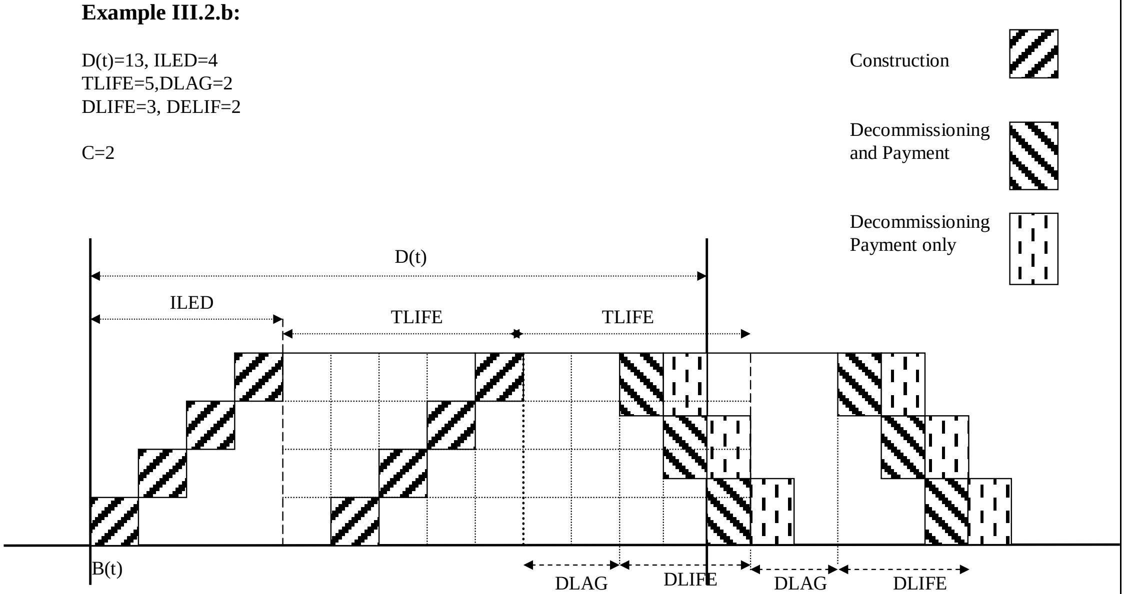
$$INVDECOM(y)=\sum_{t\in MILESTONES\ ,\ \ t\leq T(y)} INDIC(III.2.b)\times P_t(y)\times VAR\_NCAP_t\times CRF$$

III.2.b

Example III.2.b:

D(t)=13, ILED=4
TLIFE=5,DLAG=2
DLIFE=3, DELIF=2

C=2



6.2.5 Fixed annual costs: FIXCOST(y), SURVCOST(y)

The fixed annual costs are assumed to be paid in the same year as the actual operation of the facility. However, the spreading of the investment described in subsection 5.1.1 results in a tapering in and a tapering out of these costs. Taxes and subsidies on fixed annual costs are also accepted by the model.

There are two types of fixed annual costs, *FIXCOST(y)*, which is incurred each year for each unit of capacity still operating, and *SURVCOST(y)*, which is incurred each year for each unit of capacity in its *DLAG* state (this is a cost incurred for surveillance of the facility during the lag time before its demolition). Again here, the same classification of cases is adopted as in previous subsections on capital costs. Note that by assumption, *SURVCOST(y)* occurs only in cases 2. DLAG is allowed to be positive even in case 1a, but that in this case the surveillance costs are assumed to be negligible. Finally, note that *FIXCOST(y)* need be computed only for years *y* within the planning horizon, whereas *SURVCOST(y*) may exist for years beyond the horizon

Remark on early retirements:

In TIMES, any capacity may also be retired before the end of its technical lifetime, if so-called early retirements are enabled for a process. In such cases, the plant is assumed to be irrevocably shut down, and therefore fixed O&M costs would no longer occur. This situation is not taken into account in the standard formulations given below, but it has been taken into account in the model generator. To see that the expressions for the fixed annual costs, taxes and subsidies could be easily adjusted for early retirements, consider the standard expressions for *FIXCOST(y)*, which can all be written as follows.

$$FIXCOST(r,y) = \sum_{(r,v,p)\in \mathbf{rtp}} \begin{pmatrix} VAR\_NCAP_{r,v,p} \ (\ni \mathbf{t}_v) \\ + NCAP\_PASTI_{r,v,p} \end{pmatrix} \times CF_{r,v,p,y}$$

Here, $CF_{r,v,p,y}$ is the compound fixed cost coefficient for each capacity vintage in year *y*, as obtained from the original expressions for *FIXCOST(y)*. Recalling that fixed costs are accounted only within the model horizon, these expressions can be adjusted as follows:

$$FIXCOST°(r,y) = \sum_{(r,v,p)\in \mathbf{rtp}} \begin{pmatrix} VAR\_NCAP_{r,v,p} \quad (\ni \mathbf{t}_v) \\ + NCAP\_PASTI_{r,v,p} \\ - \sum_{\substack{\mathbf{prc\_rcap_{r,p}} \\ \mathbf{periodyr_{t,y}}}} VAR\_SCAP_{r,v,t,p} \end{pmatrix} \times CF_{r,v,p,y}$$

As one can see, the expressions for *FIXCOST(r,y)* can be augmented in a straightforward manner, obtaining the expressions *FIXCOST°(r,y)* that take into account early capacity retirements of each vintage, represented by the $VAR\_SCAP_{r,v,t,p}$ variables.

Case 1.a) If $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED_t \ge D(t)$
(Small projects, single investment in period)

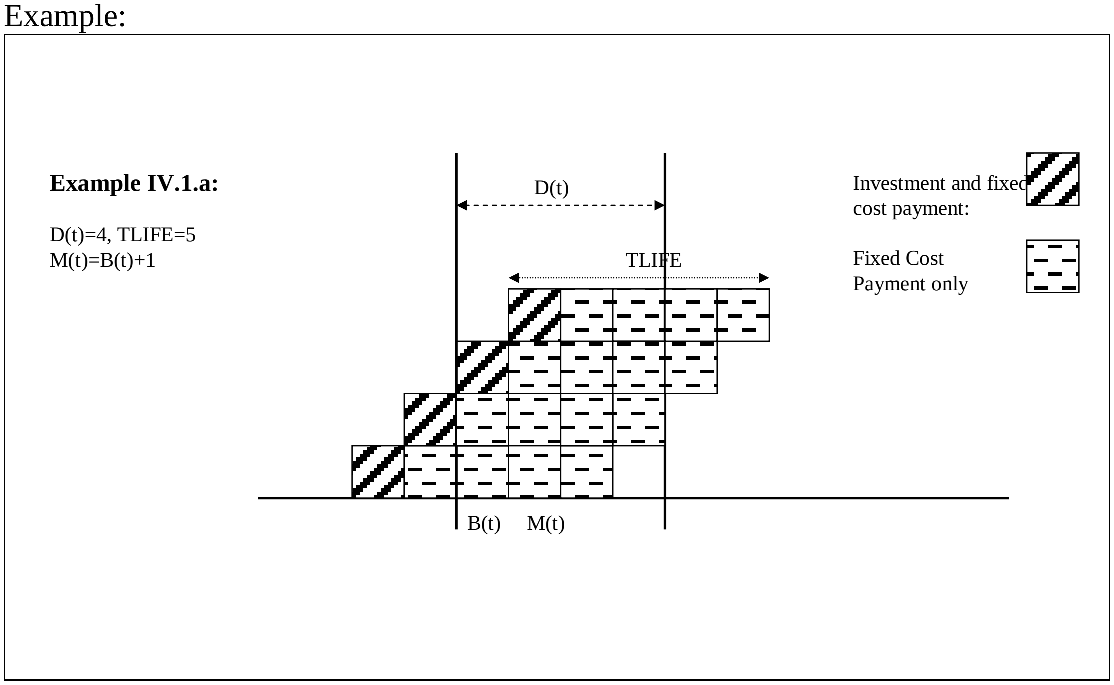
$EQ\_FIXCOST(y)$, $\quad y \le EOH$

The figure of the example shows that payments made in year *y* may come from investments made at periods before *T(y)*, at *T(y)* itself, or at periods after *T(y)*. Note that the cost attribute is multiplied by two factors: the *SHAPE*, which takes into account the vintage and age of the technology, and the *MULTI* parameter, which takes into account the pure time at which the cost is paid (the notation below for *SHAPE* and *MULTI* is simplified: it should also specify that these two parameters are those pertaining to the *FOM* attribute).

$$FIXCOST(y) =$$
$$\sum_{t \in MILESTONYR \cup PASTYEARS} INDIC(1.a) \times \sum_{v=Max\{M(t)-D(t)+1,\, y-TLIFE_t+1\}}^{Min(M(t),y)} \left(\frac{VAR\_NCAP_t}{D(t)} + NCAP\_PASTI_t \right)$$
$$\times NCAP\_FOM_v \times SHAPE(v, y-v) \times MULTI(y)$$

$$\begin{aligned} &The\ useful\ range\ for\ y\ is: \\ &\{M(t)-D(t)+1, M(t)+TLIFE_t-1\} \\ &and \\ &y \le EOH \end{aligned} \qquad \textbf{(IV.1.a)}$$

Example:



Case 1.b, if $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED < D(t)$
(Small projects, repeated investments in period)

The figure shows that payments made at year *y* may come from investments made at, before, or after period *T(y)*. Note that our expression takes into account the vintage and age of the *FOM* being paid, via the *SHAPE* parameter, and also the pure time via *MULTI*, both pertaining to the *FOM* attribute.

$$FIXCOST(y) = \sum_{t \in MILESTONYR} INDIC(1.b) \times \sum_{v=Max\{\langle B(t)-TLIFE_t/2\rangle,\, y-TLIFE_t+1\}}^{Min(y,\langle B(t)-TLIFE_t/2\rangle + C\times TLIFE_t - 1\}} \left(\frac{VAR\_NCAP_t}{TLIFE_t} \right) \times NCAP\_FOM_v$$

$$\times\ SHAPE(t, y-v) \times MULTI(y)$$

(IV.1.b)

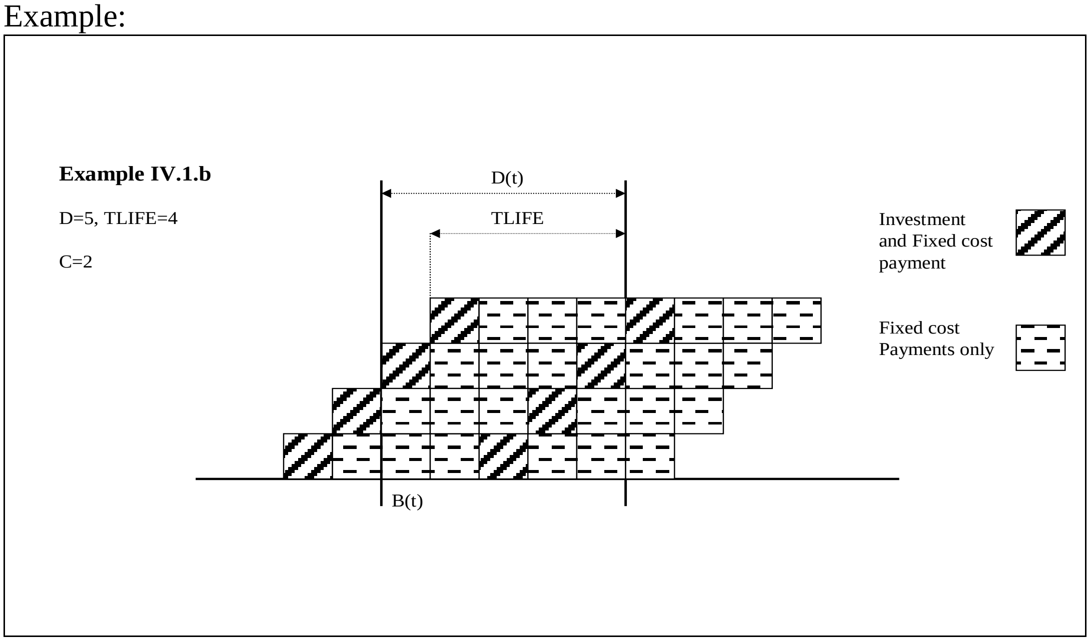
where

$$C = \left\langle \frac{D(t)}{TLIFE_t} \right\rangle$$

$Useful\ \ Range\ \ for\ \ y:$

$$\left\{ \left\langle B(t) - \frac{TLIFE_t}{2} \right\rangle, \left\langle B(t) - \frac{TLIFE_t}{2} \right\rangle + (C+1) \times TLIFE_t \right\}$$
and
$y \le EOH$

Example:



Case 2.a: $ILED_t > ILED_{Min,t}$ and $ILED_t + TLIFE_t \geq D(t)$

(Large, indivisible projects, unrepeated investment in period)

i) FIXCOST(y)

The figure of the example shows that payments made in year *y* may come from investments made at period *T(y)* or earlier, but not later. Again here the *SHAPE* has the correct vintage year and age, as its two parameters, whereas *MULTI* has the current year as its parameter. Both pertain to *FOM*.

$$FIXCOST(y) = \sum_{t \in MILESTONYR,\ t \leq T(y)} INDIC(2.a) \times \left(VAR\_NCAP_t\right) \times NCAP\_FOM_{B(t)+ILED_t}$$

$$\times \begin{Bmatrix} 1 \text{ if } B(t) + ILED_t \leq y \leq B(t) + ILED_t + TLIFE_t - 1 \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad otherwise \end{Bmatrix} \times SHAPE(t, y - B(t) + ILED_t) \times MULTI(y)$$

$$+ \sum_{t \in PASTYEARS} INDIC(2.a) \times \left(NCAP\_PASTI_t\right) \times NCAP\_FOM_t$$

$$\times \begin{cases} 1 \;\; if \qquad t \leq y \leq t + TLIFE_t - 1 \\ 0 \qquad\qquad\qquad\qquad otherwise \end{cases} \times SHAPE(t, y - t) \times MULTI(y)$$

(IV.2.a)

Useful Range for y:

$$\{B(t) + ILED_t,\ B(t) + ILED_t + TLIFE_t - 1\}$$
$$and$$
$$y \leq EOH$$

ii) SURVCOST (Surveillance cost for same case 2.a. See same example)

$$SURVCOST(y) = \sum_{\substack{t \in MILESTONYR, \\ t \leq T(y)}} INDIC(2.a) \times \left(VAR\_NCAP_t\right) \times NCAP\_DLAGC_{B(t)+ILED_t}$$

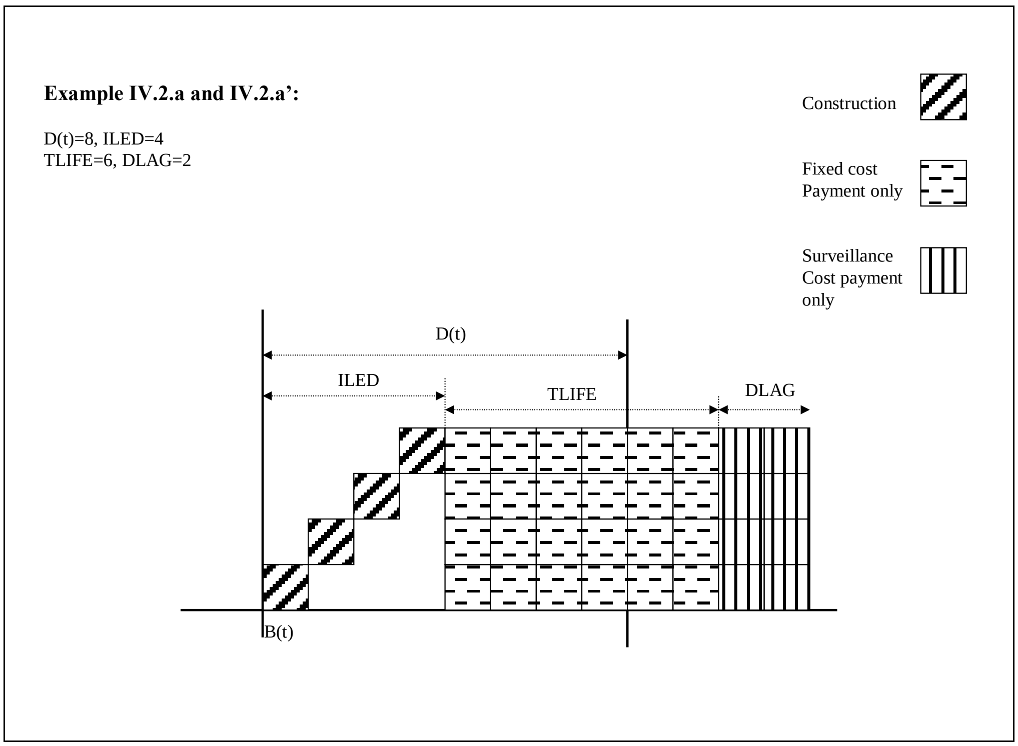
$$\times \begin{cases} 1 \;\; if \;\; B(t) + ILED_t + TLIFE_t \leq y \leq B(t) + ILED_t + TLIFE_t + DLAG_t - 1 \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad otherwise \end{cases}$$

$$
+ \sum_{t \in PASTYEARS} INDIC(2.a) \times \left(NCAP\_PASTI_t\right) \times NCAP\_DLAGC_t
$$

$$
\times \begin{cases} 1\, if\ t + TLIFE_t \leq y \leq t + TLIFE_t + DLAG_t - 1 \\ 0 \qquad\qquad\qquad\qquad otherwise \end{cases}
$$

(IV.2.a')

$$
\begin{aligned}
&Useful\ \ Range\ \ for\ \ y: \\
&\left\{B(t) + ILED_t + TLIFE_t,\ same + DLAG_t - 1\right\} \\
&note\ \ that\ \ y\ \ may\ \ be\ \ l\,\mathrm{arg}\,er\ \ than\, EOH
\end{aligned}
$$



Remark: again here, the cost attribute is indexed by the year when investment started its life. Also, note that, by choice, we have not defined the *SHAPE* or *MULTI* parameters for surveillance costs.

Case 2.b: $ILED_t > ILED_{Min,t}$ and $TLIFE_t + ILED_t < D(t)$

(Big projects, repeated investments in period)

i. Fixed O&M cost

The cost expression takes into account the vintage and the age of the *FIXOM* being paid at any given year *y*. See note in formula and figure for an explanation.

$$\sum_{t \in MILESTONES,\ t \le T(y)} INDIC(2.b) \times \left(VAR\_NCAP_t\right) \times NCAP\_FOM_{B(t)+ILED_t+I \cdot TLIFE_t}$$

$$\times SHAPE(t, y - B(t) - ILED_t - I \cdot TLIFE_t) \times \begin{cases} 1 & if \quad 0 \le I \le C-1 \\ 0 & otherwise \end{cases}$$

$$where: \quad I = \left\lfloor \frac{y - B(t) - ILED_t}{TLIFE_t} \right\rfloor \quad and \quad C = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

I is the index of the investment cycle where y lies.
I varies from 0 to C-1

Range for y:

$$\{B(t) + ILED_t,\ B(t) + ILED_t + C \times TLIFE_t - 1\} \quad and \quad y \le EOH \qquad \textbf{(IV.2.b)}$$

Remark: same as above, concerning the indexing of the cost attribute

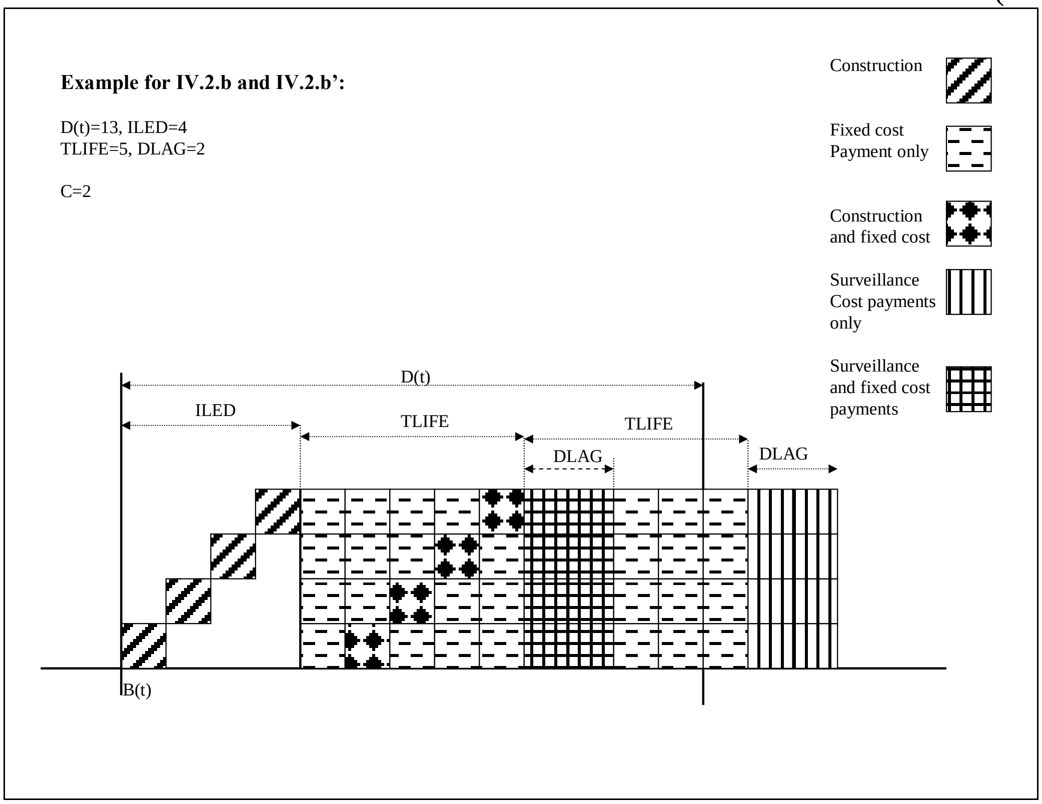
ii. SURVCOST(y) (surveillance cost for same case; the same example applies)

$$SURVCOST(y) = \sum_{\substack{t \in MILESTONES \\ t \le T(y)}} INDIC(2.b) \times \left(VAR\_NCAP_t\right) \times NCAP\_DLAGC_{B(t)+ILED_t+I \cdot TLIFE_t}$$

$$\times \begin{cases} 1 \ \ if \ \ B(t) + ILED_t + (I+1) \times TLIFE_t \le y \le same + DLAG_t - 1 \ and \ 0 \le I \le C-1 \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad otherwise \end{cases}$$

$$where: \\ I = \left\lceil \frac{y - B(t) - ILED_t - TLIFE_t}{TLIFE_t} \right\rceil \\ and \\ C = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle \\ Note\ that\ \ y\ may\ exceed\ EOH$$

(IV.2.b')



Remark: same as precedently regarding the indexing of the cost attribute *NCAP_DLAGC*

6.2.6 Annual taxes/subsidies on capacity: FIXTAXSUB(Y)

It is assumed that these taxes (subsidies) are paid (accrued) at exactly the same time as the fixed annual costs. Therefore, the expressions **IV** of subsection 5.1.4 are valid, replacing the cost attributes by *NCAP_FTAX – NCAP_FSUB.*

6.2.7 Variable annual costs *VARCOST(y)*, $y \le EOH$

Variable operations costs are treated in a straightforward manner (the same as in MARKAL), assuming that each activity has a constant activity over a given period.

In this subsection, the symbol VAR_XXX_t is any variable of the model that represents an activity at period *t*. Therefore, *XXX* may be *ACT*, *FLO*, *COMX*, *COMT*, etc. Note that, if and when the technology is vintaged, the variable has an index *v* indicating the vintage year, whereas *T(y)* indicates the period when the activity takes place. Similarly, the symbol XXX_COST_k represents the value in year *k* of any cost attribute that applies to variable *VAR_XXX*.

Finally, the expressions are written only for the years within horizon, since past years do not have a direct impact on variable costs, and since no variable cost payments occur after EOH. Note also that the SHAPE and MULTI parameters are not applicable to variable costs.

As stated in the introduction, the payment of variable costs is constant over each period. Therefore, the expressions below are particularly simple.

$$VARCOST(y) = VAR\_XXX_{v,T(y)} \times XXX\_COST_y$$
$$VARTAXSUB(y) = VAR\_XXX_{v,T(y)} \times (XXX\_TAX_y - XXX\_SUB_y)$$
$$y \le EOH$$

(VI)

6.2.8 Cost of demand reductions ELASTCOST(y)

When elastic demands are used, the objective function also includes a cost resulting from the loss of welfare due to the reduction (or increase) of demands in a given run compared to the base run. See PART I for a theoretical justification.

$$ELASTCOST(y) =$$
$$\sum_{j=1}^{COM\_STEP_{lo}} COM\_BPRICE_{T(y)} \times \left\{ \left(1 - \frac{(j-1/2) \times COM\_VOC_{lo,T(y)}}{COM\_STEP_{lo}} \right)^{\frac{1}{COM\_ELAST_{lo,T(y)}}} \right\} \times VAR\_ELAST_{lo,j,T(y)}$$
$$- \sum_{j=1}^{COM\_STEP_{up}} COM\_BPRICE_{T(y)} \times \left\{ \left(1 + \frac{(j-1/2) \times COM\_VOC_{up,T(y)}}{COM\_STEP_{up}} \right)^{\frac{1}{COM\_ELAST_{up,T(y)}}} \right\} \times VAR\_ELAST_{up,j,T(y)}$$
$$y \le EOH$$

(VII)

6.2.9 Salvage value: SALVAGE (EOH+1)

Investments whose technical lives exceed the model's horizon receive a SALVAGE value for the unused portion of their technical lives. Salvage applies to several types of costs: investment costs, sunk material costs, as well as decommissioning costs and surveillance costs. SALVAGE is reported as a single lump sum revenue accruing precisely at the end of the horizon (and then discounted to the base year like all other costs).

The salvaging of a technology's costs is an extremely important feature of any dynamic planning model with finite horizon. Without it, investment decisions made toward the end of the horizon would be seriously distorted, since their full value would be paid, but only a fraction of their technical life would lie within the horizon and produce useful outputs.

What are the costs that should trigger a salvage value? The answer is: any costs that are directly or indirectly attached to an investment. These include investment costs and decommissioning costs. Fixed annual costs and variable costs do not require salvage values, since they are paid each year in which they occur, and their computation involves only years within the horizon. However, surveillance costs should be salvaged, because when we computed them in section 6.2.5, we allowed *y* to lie beyond EOH (for convenience). Finally, note that any capacity prematurely retired within the model horizon is not assumed to have a salvage value (although this detail is not explicitly shown in the formulation below).

Thus, *SALVAGE* is the sum of three salvage values

$$SALVAGE(EOH+1) = SALVINV(EOH+1) + SALVDECOM(EOH+1) + SALVSURV(EOH+1)$$

We treat each component separately, starting with *SALVINV*.

A). Salvaging investment costs (from subsections 6.2.2 and 6.2.3)

The principle of salvaging is simple, and is used in other technology models such as MARKAL, etc: a technology with technical life *TLIFE*, but which has only spent *x* years within the planning horizon, should trigger a repayment to compensate for the unused portion *TLIFE-x* of its active life. The computation of the salvage value obeys a simple rule, described by the following result:

Result 1

The salvage value (calculated at year *k*) of a unit investment made in year *k*, and whose technical life is *TL*, is:

$$S(k,TL) = 0 \qquad if\ k + TL \le EOH$$

$$S(k,TL) = 1 \qquad if\ k > EOH$$

$$S(k,TL) = \frac{(1+d)^{TL-EOH-1+k} - 1}{(1+d)^{TL} - 1} \quad otherwise$$

where d is the general discount rate

Note that the second case may indeed arise, because some investments will occur even after *EOH.*

Since we want to calculate all salvages at the single year *(EOH+1)*, the above expressions for *S(k,TL)* must be discounted (multiplied) by:

$$(1+d)^{EOH+1-k}$$

Finally, another correction must be made to these expressions, whenever the user chooses to utilize a technology specific discount rate. The correction factor which must multiply every investment (and of course every salvage value) is:

$$\frac{CRF_s}{CRF} = \frac{\left(1-\frac{1}{1+i_s}\right)\times\left(1-\frac{1}{(1+i)^{ELIFE}}\right)}{\left(1-\frac{1}{1+i}\right)\times\left(1-\frac{1}{(1+i_s)^{ELIFE}}\right)}$$

where i *is the general discount rate*

i_s *is the technology specific discount rate*

and $ELIFE$ *is the economic life of the investment*

Note: the time indexes have been omitted for clarity of the expression.

The final result of these expressions is *Result 2* expressing the salvage value discounted to year *EOH+1*, of a unit investment with technical life *TL* made in year *k* as follows. Result 2 will be used in salvage expressions for investments and taxes/subsidies on investments.

Result 2

$$SAL(k,TL)=0 \qquad\qquad if\ k+TL\le EOH$$

$$SAL(k,TL)=\frac{CRF_s}{CRF} \qquad\qquad if\ k\ge EOH+1$$

$$SAL(k,TL)=\frac{1-(1+d)^{EOH+1-k-TL}}{1-(1+d)^{-TL}}\times\frac{CRF_s}{CRF} \quad otherwise$$

where d is the general discount rate
and d_s is the technology specific discount rate

These expressions may now be adapted to each case of investment (and taxes/subsidies on investments). We enumerate these cases below. Note that to simplify the equations, we have omitted the second argument in *SAL* (it is always $TLIFE_t$ in the expressions).

Case 1.a $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED_t \ge D(t)$
(Small divisible projects, non-repetitive, progressive investment in period)

$$SALVINV(EOH+1)=\sum_t INDIC(I.1.a)\times\sum_{v=M(t)-D(t)+1}^{M(t)}\left(\frac{VAR\_NCAP_t}{D(t)}+NCAP\_PASTI_t\right)\times NCAP\_COST_v\times SAL(v)$$

where $SAL(v)$ is equal to $SAL(v,TLIFE_t)$ defined in Result 2.

Note that $SAL(v)=0$ whenever $v+TLIFE_t\le EOH+1$ (see Result 2)

(VIII.1.a)

Case 1.b $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED < D(t)$
Small Projects, repeated investments in period

$$SALVINV(EOH+1)=\sum_t INDIC(I.1.b)\times\sum_{v=B(t)-\rangle TL/2\langle+(C-1)\times TLIFE_t}^{B(t)-\rangle TL/2\langle+C\times TLIFE_t-1}\frac{VAR\_NCAP_t}{TLIFE_t}\times NCAP\_COST_v\times SAL(v)$$

Note again here that $SAL(v)$ *equals* 0 *if* $v+TLIFE\le EOH+1$

(VIII.1.b)

Case 2.a: $ILED_t > ILED_{Min,t}$ and $ILED_t + TLIFE_t \geq D(t)$
(Large, indivisible projects, unrepeated investment in period)

$$SALVINV(EOH+1) = \sum_{t \in MILESTONESYEARS} VAR\_NCAP_t \times NCAP\_COST_{B(t)+ILED_t} \times SAL(B(t)+ILED_t)$$

$$\textit{Note that } SAL\ (B(t)+ILED_t) = 0 \textit{ whenever } B(t)+ILED_t + TLIFE_t \leq EOH+1$$

(VIII.2.a)

Case 2.b: $ILED > ILED_{Min,t}$ and $TLIFE_t + ILED_t < D(t)$
(Large, indivisible Projects, repeated investments in period)

$$SALVINV(EOH+1) = \sum_t VAR\_NCAP_t \times NCAP\_COST_{B(t)+(C-1)\times TLIFE_t + ILED_t} \times SAL\big(B(t)+(C-1)\times TLIFE_t + ILED_t\big)$$

$$\textit{Note again that } SAL\big(B(t)+(C-1)\times TLIFE_t + ILED_t\big) = 0 \textit{ whenever } B(t)+(C-1)\times TLIFE_t + ILED_t + TLIFE_t \leq EOH+1$$

(VIII.2.b)

NOTE: salvage cost of taxes/subsidies on investment costs are identical to the above, replacing NCAP_COST by {NCAP_ITAX – NCAP_ISUB}.

B). Savage value of decommissioning costs (from subsection 6.2.4)

For decommissioning costs, it should be clear that the triggering of salvage is still the fact that some residual life of the *investment itself* exists at *EOH+1*. What matters is *not* that the decommissioning occurs after EOH, but that some of the investment life extends beyond EOH. Therefore, Result 1 derived above for investment costs, still applies to decommissioning. Furthermore, the correction factor due to the use of technology specific discount rates is also still applicable (with *ELIFE* replaced by *DELIF*).

However, the further discounting of the salvage to bring it to *EOH+1* is now different from the one used for investments. The discounting depends on the year *l* when the decommissioning occurred and is thus equal to:

$$(1+d)^{EOH+1-l} \textit{ where } l \textit{ is the year when decommissioning occurs.}$$

l depends on each case and will be computed below

In cases 1.a and 1.b, $l = TLIFE + k$

In case 2.a	*k* is fixed at *B(t)+ILED*, but *l* varies from *(B(t)+ILED+TLIFE+DLAG)* to *(same +DLIFE–1)*
In case 2.b	*k* is fixed at *B(t)+ILED+(C–1)×TLIFE*, but *l* varies from *(B(t)+ILED+C×TLIFE+DLAG) to (same + DLIFE–1)*

It is helpful to look at the examples for each case in order to understand these expressions.

Finally, the equivalent of Result 2 is given as Result 3, for decommissioning.

Result 3

The Salvage Value of a decommissioning cost occuring at year l, for an investment taking place at year k, is:

$$SAL(k,l) = 0 \qquad if\ k + TL \le EOH$$

$$SAL(k,l) = \frac{CRF_s}{CRF} \times (1+i)^{EOH+1-l} \qquad if\ k \ge EOH+1$$

$$SAL(k,l) = \frac{(1+d)^{TLIFE+k-l} - (1+d)^{EOH+1-l}}{(1+d)^{TLIFE} - 1} \times \frac{CRF_s}{CRF} \quad otherwise$$

where d is the general discount rate
and d_s is the technology specific discount rate

We are now ready to write the salvage values of decommissioning cost in each case.

Case 1.a $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED_t \ge D(t)$
(Small divisible projects, non-repetitive, progressive investment in period)

$$SALVDECOM(EOH+1) =$$

$$\sum_t INDIC(1.a) \times \sum_{v=M(t)-D(t)+1}^{M(t)} \left(\frac{VAR\_NCAP_t}{D(t)} + NCAP\_PASTI_t \right) \times$$
$$NCAP\_DCOST_v \times SAL(v, v+TLIFE_t)$$

where $SAL(k,l)$ is defined in Result 3.

Note that $SAL(v,x)$ is always 0 whenever $v + TLIFE \le EOH + 1$ **(IX.1.a)**

Case 1.b $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED < D(t)$
(Small Projects, repeated investments in period)

$$SALVDECOM(EOH+1) =$$

$$\sum_t INDIC(1.b) \times \sum_{v=B(t)-\rangle TL/2\langle +(C-1)\times TLIFE_t}^{B(t)-\rangle TL/2\langle +C\times TLIFE_t -1} \frac{VAR\_NCAP_t}{TLIFE_t} \times NCAP\_DCOST_v \times SAL(v, v+TLIFE_t)$$

$Note\ \ again\ here\ that\ \ SAL(k,l)\ \ equals\ \ 0\ \ if\ \ k+TLIFE \le EOH+1$

(IX.1.b)

Case 2.a: $ILED_t > ILED_{Min,t}$ and $ILED_t + TLIFE_t \ge D(t)$
(Large, indivisible projects, unrepeated investment in period)

$$SALVDECOM(EOH+1) =$$

$$\sum_{t \in MILESTONESYEARS} INDIC(2.a) \times VAR\_NCAP_t \times NCAP\_COST_{B(t)+ILED_t} \times \sum_{l=B(t)+TLIFE+DLAG}^{same+DLIFE-1} SAL(B(t)+ILED_t, l)$$

$Note\ \ that\ \ SAL\ \ is\ \ 0\ \ whenever\ \ B(t)+ILED_t+TLIFE_t \le EOH+1$

(IX.2.a)

Case 2.b: $ILED_t > ILED_{Min,t}$ and $TLIFE_t + ILED_t < D(t)$

(Large, indivisible Projects, repeated investments in period)

$$SALVDECOM(EOH+1) = \sum_{t \in MILESTONYEARS} INDIC(2.b) \times VAR\_NCAP_t \times NCAP\_DCOST_{B(t)+(C-1)\times TLIFE_t + ILED_t}$$
$$\times \sum_{l=B(t)+ILED_t+C\times TLIFE_t+DLAG_t}^{same+DLIFE-1} SAL\left[B(t)+ILED_t+(C-1)\times TLIFE_t, l\right]$$

where

$$C = \left\langle \frac{D(t)-ILED_t}{TLIFE_t} \right\rangle$$

$Note\ \ again\ \ that\ \ SAL\ \ is\ \ 0\ \ whenever\ \ B(t)+C\times TLIFE_t + ILED_t \le EOH+1$

(IX.2.b)

C) Salvage Value of Surveillance Costs

Similarly to the salvaging of decommissioning costs, the basic salvage value fractions *S(k,m)* defined in *Result 1* at the beginning of Section 6.2.9 are used as the basis for the salvage value of surveillance costs. However, unlike with decommissioning costs, there is no need to make corrections for technology-specific discount rates, as the costs do not represent capital costs. In addition, the discounting to *EOH+1* must be made separately for each surveillance year. Note that only Cases 2 have surveillance costs.

Case 2.a: $ILED_t > ILED_{Min,t}$ and $ILED_t + TLIFE_t \ge D(t)$

(Large, indivisible projects, unrepeated investment in period)

$$SALVSURV(EOH+1) =$$
$$\sum_{t \in MILESTONESYEARS} INDIC(2.a) \times S(B(t)+ILED_t, TLIFE_t) \times$$
$$VAR\_NCAP_t \times NCAP\_DLAGC_{B(t)+ILED_t} \times \sum_{l=B(t)+ILED_t+TLIFE_t}^{same+DLAG_t-1} DISC(l, EOH+1)$$

Note that $S(k,m) = 0$ whenever $k+m \le EOH+1$.

(X.2.a)

Case 2.b: $ILED_t > ILED_{Min,t}$ and $TLIFE_t + ILED_t < D(t)$

(Large, indivisible projects, repeated investments in period)

$$SALVSURV(EOH+1) =$$

$$\sum INDIC(2.b) \times S[B(t) + ILED_t + (C-1) \times TLIFE_t, TLIFE_t] \times VAR\_NCAP_t \times$$

$$NCAP\_DLAGC_{B(t)+ILED+(C-1)\times TLIFE} \times \sum_{l=B(t)+ILED_t+C\times TLIFE_t}^{same+DLAG_t-1} DISC(l, EOH+1)$$

$$where: C = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

Note again that $S(k,m) = 0$ whenever $k + m \leq EOH + 1$.

(X.2.b)

6.2.10 Late revenues from endogenous commodity recycling after EOH LATEREVENUE(y)

Late revenues consist of revenues from any materials and energy which had been embedded in some processes, and which are released after *EOH*. Such revenues exist only if an exogenous salvage value was declared by the user for the sunk material.

Note: For materials released within the horizon, the revenue is either explicit (and then it is the user's responsibility to indicate a negative cost – credit – at dismantling time), or the revenue is implicit, and then the user must specify a physical release of the material at dismantling time, and the model will correctly 'price' this material within the RES.

LATEREVENUES(y) $y \geq EOH+1$

The late revenues come *only* from the resale at dismantling time, of materials and/or energy that were sunk at construction time. Therefore, the *LATEREVENUES* expressions are identical to the decommissioning cost expressions, with the NCAP_DCOST attribute replaced by

$$\sum_c -NCAP\_VAL(c) \times NCAP\_OCOM(c)$$

where the summation extends over all commodities c for which an *NCAP_OCOM* attribute is defined (defaults to zero if undefined)

LATEREVENUES(y) is reported as a lump sum discounted to the user selected base year.

6.2.11 Known issues in the standard objective function formulation

There are a few known issues in the standard objective function formulation that may cause small distortions in the cost accounting and, subsequently, in the relative competitiveness of technologies. The distortions only occur when using period lengths D(t) > 1. The issues can be briefly summarized as follows:

- In the investment cases 1.a and 1.b, the timing of the annual payments for the investment costs and fixed operation and maintenance costs are not fully in sync with the assumed amounts of available capacity. Although the effective difference is usually quite small, with longer periods having an even number of years, the distortion may become considerable.
- In the investment cases 1.a and 1.b, the spreading of the investment cost over D(t) or TLIFE(p) years causes some distortions in the salvage value accounting, which are at the highest in cases where B(v)+TLIFE = EOH+1, (capacity is retired exactly at the end of the horizon), because in such cases the capacity is assumed fully available within the model horizon, but it still has a salvage value according to the standard formulation.
- In all investment cases, the capacity is assumed to be available in each period according to the proportion of the period being covered by the years [B(v)+ILED(v),B(v)+ILED(v)+TLIFE(v)–1]. If all periods contain only a single year, this is quite accurate, but, due to discounting, it is no longer accurate with longer periods. That is because any capacity available in year y has a larger value than the same capacity available in year y+1. But again, this causes only a small distortion in the cost accounting.
- With variable period lengths, investments for period t can start even before the previous milestone year t–1. If the investment costs are changing over time, in such cases the costs are not accounted in a fully consistent way, because the investment cost data is taken from the start year of each investment step.

The first three of these issues have been addressed by introducing an optional switch ($SET OBLONG YES), which, when activated, will eliminate all those three issues. For the first two issues, the discounting of the annual payments for the investment costs and fixed operation and maintenance costs is slightly modified, such that the weighted average of the commissioning years over the investment steps is exactly equal to B(v) (the weights being the present value factors for the commissioning years). In other words, the modification introduces a small additional discounting multiplier, which moves the whole investment spread slightly in time, such that the resulting costs will effectively always be in sync with the assumed available capacity (and activity).

For the third issue, the capacity transfer coefficients are slightly modified to reflect the true value of the capacity in each period, based on the *discounted* proportion of the period being covered by the process lifetime.

The modified objective function has been verified to produce results that are fully consistent with single-year period results, assuming that process parameters do not change over time, which is the best what one can expect. The fourth issue can only be addressed by using any of the alternative objective formulations (see separate *Objective Function Variants* documentation, available at the ETSAP documentation website).

6.2.12 The discounting methods for annual payments

In the standard objective function of TIMES, all costs and payments are assumed to occur at the beginning of each year. In the case of investment costs, this means that the annualized payments made in the beginning of each year within the economic lifetime are equivalent to a lump-sum investment cost paid at the beginning of the first operation year, if the annual payments are discounted back to that point by the technology-specific discount rate (for instance, in case 1a, each lump sum is equal to *NCAP_COST/D*(*t*)). Similarly, in the case of operation costs (e.g. *NCAP_FOM)*, the total annual costs are assumed to occur at the beginning of each operating year.

Because the operating costs can nevertheless be assumed to be spread continuously throughout the year, this kind of 'beginning-of-year' discounting method introduces a small bias in the discounting of different cost components. For example, the operating costs in the first year of operation should be assumed to occur about half a year later in time compared to the investment, and not at the same time, as assumed in TIMES. One may well argue that this time-difference should be reflected in the discounting applied.

In TIMES, there is an option to correct this small bias by using mid-year discounting, or even end-of-year discounting. The options can be activated by the switch *MID_YEAR / DISCSHIFT* (see Part III, Control switches). The modifications needed in the discounting are basically quite similar for employing both mid-year and end-of-year discounting. Therefore, only the corrections for mid-year discounting are described in detail below.

The corrections needed for employing mid-year discounting in TIMES can be made in the following two steps:

1. First, simply assume that instead of the beginning of each year, all payments are made in the mid-point of each year in TIMES. As such, this assumption doesn't change the objective function in any way; it is only a change in thinking. However, it also means that instead of the beginning of the base year, all costs are assumed to be discounted to the mid-point of the base year.
2. Second, make the necessary corrections to the discounting of all those cost components that cannot be assumed to be actually paid at the mid-point of the year.

By going through the various cost components, the following conclusions hold for step 2:

- All variable, fixed operation and surveillance costs can be assumed to be paid in the mid-point of each year, and no change is needed for them in the discounting.
- The lump-sum investment costs in Cases 1 *(NCAP_COST/D(T))* should be assumed to occur at the beginning of the investment year instead of the mid-point.
- All the lump-sum investment costs in Cases 2 *(NCAP_COST/ILED)* can be assumed to occur in the mid-point of each construction year. Therefore, no change is needed in the discounting of the annualized investment payments.
- Decommissioning costs in Cases 1 can be assumed to be paid in the mid-point of the year, because in these cases decommissioning is assumed to take exactly one year, and one may assume that, on the average, the costs occur at the mid-point.

- The lump-sum decommissioning costs in Cases 2 *(NCAP_DCOST/DLIFE)* can be assumed to occur at the mid-point of each year within the decommissioning lifetime. Therefore, no change is needed in the discounting of the annualized payments.

Consequently, the initial overall conclusion is that the only correction needed in the discounting of various cost components is related to the investment costs in Cases 1. If we assume that the Capital Recovery Factor used in the beginning-of-year discounting (CRF_{beg}) is still valid for mid-year discounting, we should simply shift the position of both the lump-sum investment and the annualized payments half a year backwards. In terms of discounting, this means that in Cases 1 the annualized investment payments should be multiplied by the factor $(1+d(y))^{0.5}$, where $d(y)$ is the **general discount rate**. Perhaps the simplest way to apply this correction in the objective function is to make the adjustment to the Capital Recovery Factor. Thus, for Cases 1 we could define a 'CRF corrected for mid-year discounting' ($CRF_{1,mid}$) as follows:

$$CRF_{1,mid} = CRF_{beg} \times (1+d(T(y)))^{0.5}$$

However, one could additionally argue that the Capital Recovery Factor CRF_{beg} is no longer valid for mid-year discounting. The annualized investment payments can also be assumed to represent a continuous stream of costs, which should thus be assumed to be paid at the mid-point of each year. The shortcoming of the original CRF_{beg} can be seen by calculating its value for an investment with an economic lifetime of just one year. The value of CRF_{beg} is in this case exactly 1, although it seems obvious that some interest should be involved as well. Assuming that the single payment represents a continuous stream of costs, the payment can be assumed to occur at the mid-point of the year, and would thus include interest for half-year's time.

Accordingly, we should correct the definition of the *CRF proper* by assuming that the annualized payments occur *half a year forward* in time with respect to the lump-sum investment, which means that we must increase the nominal size of the payments by the corresponding interest for the half-year's time. Combining these corrections together, the general discount rate $d(y)$ should be simply replaced by the **technology-specific discount rate** $d_S(T(y))$ in the expression above, because in addition to the nominal change in the *CRF*, the time of the annualized payments has been restored back to original. However, to maintain consistency between Cases 1 and 2, the same basic correction to the *CRF proper* should be applied to all cases. Therefore, the total adjustments needed when taking into account the correction to the *CRF proper* are the following:

$$CRF_{mid}^{proper} = CRF_{beg} \times (1+d_S(T(y)))^{0.5} \quad \text{(XI.1)}$$

$$\begin{aligned} CRF_{1,mid} &= CRF_{mid}^{proper} \times (1+d(T(y)))^{0.5} \times (1+d(T(y)))^{-0.5} = \\ &\quad CRF_{beg} \times (1+d_S(T(y)))^{0.5} \end{aligned} \quad \text{(XI.2)}$$

$$\begin{aligned} CRF_{2,mid} &= CRF_{mid}^{proper} \times (1+d(T(y)))^{-0.5} = \\ &\quad CRF_{beg} \times (1+d(T(y)))^{-0.5} \times (1+d_S(T(y)))^{0.5} \end{aligned} \quad \text{(XI.3)}$$

Consequently, in both cases the annualized investment payments are then assumed to occur at the mid-point of each fiscal year starting at the time of the lump-sum investment, and the annual payments are equivalent to the lump-sum investment when discounted back to that point by the technology-specific discount rate. The implementation of the optional corrections for mid-year discounting corresponds to equations (XI.1 to XI.3). To be consistent, the expression (XI.3) for $CRF_{2,mid}$ should also be used for decommissioning costs.

6.3 Constraints

The constraints available in standard TIMES are shown in Table 23 below, and later fully described in the following subsections. The constraints related to the Climate Module (CLI), Damage Cost Functions (DAM) and Endogenous Technology Learning (ETL) are shown and described in three separate chapters (Appendices A, B and C respectively).

Table 24. List of TIMES equations

Equation Name	Short description
BND_ELAST	Upper bound on each of the step variables used to linearize the demand function when elastic demand feature is used
EQ(l)_ACTBND	Bound on the activity of a process
EQE_ACTEFF	Equality relationship that defines the activity efficiency of a process
EQ_ACTFLO	Equality relationship that defines the activity of a process in terms of its flow variables
EQ_ACTPL	Defines the efficiency deterioration of a process at partial loads
EQ_ACTRAMP	Defines bounds on the ramping of process activity, in proportion to its online capacity, in either direction (LO/UP)
EQL_ACTUPC	Sets a lower limit on the successive on-line / off-line hours of capacity
EQE_ACTUPS	Expresses that the change in process on-line capacity between successive timeslices must be equal to the capacity started-up – shut-down
EQL_ACTUPS	Expresses that the sum of process started-up capacity over a cycle must be at least equal to the max. amount of capacity put off-line in the cycle
EQ(l)_ASHAR	Establishes advanced share constraints between process flows
EQ(l)_BLND	Special blending constraints used to specify the composition of refined oil products
EQ_BNDCST	Establishes a variable representing the cumulative amount of process costs, taxes and/or subsidies over a time interval, for defining a bound
EQ(l)_BNDNET	Bound on the net amount (production minus consumption) of a commodity
EQ(l)_BNDPRD	Bound on the total production of a commodity
EQ(l)_CAFLAC	Relates the flows in the primary group of a process to its available capacity; may be rigid (=) or flexible (≤)
EQ(l)_CAPACT	Relates the activity of a process to its available capacity; may be rigid (=) or flexible (≤,≥)
EQL_CAPFLO	Relates a flow not in the primary group of a process to its available capacity; only an upper bound for the flow ≤ is supported
EQ_CAPLOAD	Relates the activity of a process to its available on-line capacity in each timeslice; only for processes with flexible availability (≤,≥)
EQ(l)_CPT	Calculates the current capacity of a process in terms of all past and current investments in that process
EQ(l)_COMBAL	Balance equation of a commodity
EQE_COMPRD	Definition of the total production of a commodity
EQ_CUMFLO	Bound on the cumulative flow or activity of a process over a time interval
EQ_CUMNET	Bound on the cumulative production of a commodity over a time interval
EQ_CUMPRD	Bound on the cumulative net quantity of a commodity over a time interval
EQ_CUMRET	Establishes a variable representing the cumulative amount of retired capacity of a process

Equation Name	Short description
EQ_DSCNCAP and EQ_DSCONE	These two constraints ensure that some investments may only be made in certain discrete sizes
EQ_DSCRET	Ensures that early capacity retirements may only be made in multiples of a certain discrete block size
EQ(l)_FLOBND	Bound on the sum over a commodity group, of the commodity flows of a process
EQ(l)_FLOFR	Relationship between a flow in one timeslice and the annual flow, for a given process
EQ(l)_FLMRK	Expresses for a given commodity that the amount produced/consumed by a process is tied to the total amount produced/consumed of that commodity
EQ_IRE	Expresses that imports of a commodity by region r must be equal to all exports by other regions to region r
EQ_IREBND	Bound on exchange of a commodity between two regions
EQ_XBND	Bound on total exchanges of a commodity by one region
EQ(l)_INSHR	For a given process, expresses that the inflow of a commodity is tied to the total inflows of all commodities in a certain group
EQ(l)_OUTSHR	For a given process, expresses that the outflow of a commodity is tied to the total outflows of all commodities in a certain group
EQ_PEAK	Expresses that capacity available must exceed demand of a selected commodity in any time slice by a certain margin
EQ_PTRANS	Establishes an equality relationship between (groups of) inputs and certain (groups of) outputs of a process
EQL_SCAP	Bounds the amount of capacity salvaged if early retirements are active.
EQ_STGAUX	Establishes an equality relationship between storage main flows or activity and an auxiliary storage flow
EQ_STGIPS	Ensures the storage of a commodity between two time periods
EQ_STGTSS	Ensures the storage of a commodity between two timeslices
EQ(l)_STGIN	Bounds the input into a storage process
EQ(l)_STGOUT	Bounds the output of a storage process
EQ_STSBAL	Defines balances between timeslice levels in a general timeslice storage
EQ(l)_UCRTP	Defines a dynamic bound on the growth / decay in the installed capacity, new capacity or activity of a process over successive periods
User Constraints of the LHS type	User defined constraints that have a user defined constant RHS
Timeslice-dynamic User Constraints	User defined constraints that involve only a single region **r** and period **t** but both timeslice **s** and the preceding timeslice **s–rs_stg**(r,s)
User Constraints of dynamic type (t,t+1)	User defined constraints that involve both period **t** and the succeeding period **t+1**
User Constraints of dynamic type (t–1,t)	User defined constraints that involve both period **t** and the preceding period **t–1**

6.3.1 Bound: BND_ELAST

Indices: region (r), year (t), commodity (c), time slice (s), linearization step (j), direction of elastic demand change (*l*)

Type: ≤

Related variables: VAR_ELAST

Related equations: EQ(l)_COMBAL, EQ_OBJELS, EQ_OBJ

Purpose: Upper Bounds on the step variables used to represent the demand when the elasticity is non-zero.

Remarks:

- These bounds are applied whenever a demand is price elastic, i.e. when the COM_ELAST (elasticity) and COM_VOC (total range) parameters are specified and not zero.
- If COM_ELAST and COM_VOC are specified, and COM_STEP (number of steps) is not, the latter defaults to 1 (single step discretization)
- Attributes COM_VOC and COM_STEP do not have a timeslice index. The user can still control elasticities in each time slice through COM_ELAST_s.

Bound:

$$BND\_ELAST_{r,t,c,s,j,l} \ni COM\_STEP_{r,c,l} \wedge (s \in \mathbf{com\_ts}_{\mathbf{r,c,s}})$$

$$VAR\_ELAST_{r,t,c,s,j,l} \leq \frac{COM\_PROJ_{r,t,c} \times COM\_FR_{r,t,c,s} \times COM\_VOC_{r,t,c,l}}{COM\_STEP_{r,c,l}}$$

6.3.2 Equation EQ(*l*)_ACTBND

Indices: **region (r), model year (t), process (p), time slice (s)**

Type: Any type, as determined by the index **bd** of ACT_BND:

- l = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- l = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- l = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Related variables: **VAR_ACT**

Related equations: **EQ_COMBAL, EQ_ACTFLO, EQ_PTRANS**

Purpose: This equation bounds the total activity of a process in a period independently of the vintage years of the installed capacities. The equation will either be generated when the activity bound is specified for a timeslice being at a timeslice level above the timeslice level of the process (**prc_tsl**), e.g. ACT_BND is specified for an ANNUAL timeslice but the process operates on a DAYNITE timeslice level, or irrespectively of the timeslices when the process is characterized as a vintaged one (**prc_vint**). If activity bounds are specified for timeslices below the process timeslice level (**prc_tsl**), the bounds will be aggregated to the process timeslice level by standard aggregation (see Section 3.1.2) and then directly applied to the activity variable for non-vintaged processes. The same is true for activity bounds specified at the process timeslice level of non-vintaged processes.

Remarks:

- The equation is required because for the two cases described above (bound specified for a timelslice above the process timeslice level or process is characterized as a vintaged one), no single variable exists which can be bounded directly.
- The bound is only directly applied to VAR_ACT for non-vintaged processes, when ACT_BND is applied at the level **prc_ts(r,p,s)**.

Interpretation of the results:

Primal: The level value of the equation describes the activity of the process in the considered period **t** and timeslice **s**.

Dual: The dual variable describes in the case of a lower (upper) bound the cost increase (decrease) caused by an increase of the activity bound by one unit.

Equation:

Activity must exist and process is available in period t

All timeslices at or above **prc_tsl**

$$EQ(l)\_ACTBND_{r,t,p,s} \quad \ni ACT\_BND_{r,t,p,s,bd} \wedge \mathbf{rtp\_vara}_{\mathbf{r,t,p}} \wedge \mathbf{rps\_prcts}_{\mathbf{r,p,s}} \wedge \left(p \in \mathbf{prc\_vint}_{\mathbf{r,p}} \vee s \notin \mathbf{prc\_ts}_{\mathbf{r,p,s}}\right)$$

either p is vintaged or the bound is applied for a n exact slice of p

$$\sum_{v \in \mathbf{rtp\_vintyr}} \sum_{s2 \in \mathbf{prc\_ts}} VAR\_ACT_{r,v,t,p,s2} \ \{=;\leq;\geq\} \ ACT\_BND_{r,t,p,s,l}$$

*s2: all timeslices on process timeslice level(***prc_ts***) that are descendents of s in the timeslice tree; determined by the internal set* **ts_map(r,s,s2)**.

6.3.3 Equation: EQE_ACTEFF

Indices: **region (r), vintage year (v), period (t), process (p), commodity group (cg), side (io), timeslice (s)**

Type: =

Related variables: **VAR_ACT, VAR_FLO**

Related equations: **EQ_PTRANS, EQ_ACTPL**

Purpose: This equation is generated when the process activity efficiency has been defined with the input attribute $ACT\_EFF_{r,v,p,cg,s}$ for a group of flows on the shadow side.

Remarks:

- The group cg in the equation may be either directly specified in ACT_EFF, or, if *ACT_EFF* is only specified for single commodity, determined as the commodity type, or, if *ACT_EFF* is specified for the reserved group name 'ACT', determined as the default shadow group of the process.
- The parameter $ACT\_EFF_{r,v,p,cg,s}$ can be specified using any of the following as the cg:
  - commodity groups; these define a common efficiency for all member commodities in the group that are on the shadow side of the process;
  - commodity types (NRG/MAT/ENV/DEM/FIN); as above, these define a common efficiency for all member commodities in the group that are on the shadow side of the process;
  - the predefined commodity group 'ACT'; this defines a common efficiency for all members of the default shadow group of the process;
  - single commodities on the shadow side without an associated group efficiency; these define commodity-specific efficiencies, and the shadow group will consist of all commodities of the same type; if no commodity efficiency is defined for some member in the group, the default efficiency 1 is assumed;
  - single commodities on the shadow side with an associated group efficiency; these define commodity-specific efficiencies as above, but are multiplied by the efficiency specified for the group; if no efficiency is defined for some member in the group, the group efficiency is applied directly to that member;
  - single commodities C that are members of the PCG of the process; these define commodity-specific multipliers for the process efficiency when producing the commodity C; if no efficiencies are additionally defined on the shadow side of the process, the whole standard shadow group of the process is assumed to be involved in the transformation (as when using ‘ACT’), with the default efficiency of 1 on the shadow side.
- The ACT_EFF parameter can also be shaped by using a FLO_FUNCX parameter of the following form: FLO_FUNCX(reg,datayear,p,CG,'ACT') = shape index. Here, the CG should correspond to the group of commodities on the shadow side involved in the EQE_ACTEFF equation (the group, commodity type, or 'ACT' that was either explicitly or implicitly used in the ACT_EFF parameters that should be shaped).

Equation:

$$EQE\_ACTEFF_{r,v,t,p,cg,io,s} \ni (\mathbf{rtp\_vintyr_{r,v,t,p}} \wedge \neg \mathbf{rp\_inout_{r,p,io}} \wedge ACT\_EFF_{r,v,p,cg,s})$$

$$\sum_{\substack{\mathbf{com\_gmap_{r,cg,c}} \\ \mathbf{rtcp\_varf_{r,t,p,c,s}}}} \begin{pmatrix} VAR\_FLO_{r,v,t,c,ts} \times \\ \begin{pmatrix} ACT\_EFF_{r,v,p,c,ts} & if\ ACT\_EFF_{r,v,p,c,ts}\ given \\ 1 & otherwise \end{pmatrix} \\ \times RTCS\_TSFR_{r,t,c,s,ts} \end{pmatrix}$$

$$=$$

$$\sum_{\substack{\mathbf{rpc\_pg_{r,p,c}} \\ \mathbf{prc\_ts_{r,p,ts}}}} \begin{pmatrix} \begin{pmatrix} VAR\_ACT_{r,v,t,p,ts} & if\ RP\_PGACT_{r,p} \\ \dfrac{VAR\_FLO_{r,v,t,p,c,ts}}{PRC\_ACTFLO_{r,v,p,c}} & otherwise \end{pmatrix} \times \\ \begin{pmatrix} 1/ACT\_EFF_{r,v,p,cg,ts} & if\ ACT\_EFF_{r,v,p,cg,ts}\ given \\ 1 & otherwise \end{pmatrix} \times \\ \begin{pmatrix} 1/ACT\_EFF_{r,v,p,c,ts} & if\ ACT\_EFF_{r,v,p,c,ts}\ given \\ 1 & otherwise \end{pmatrix} \\ \times RTCS\_TSFR_{r,t,c,s,ts} \end{pmatrix} +$$

$$\sum_{\mathbf{prc\_ts_{r,p,ts}}} \begin{pmatrix} VAR\_UPS_{r,v,t,p,ts,'FX'} \times & \\ ACT\_LOSPL_{r,v,p,'FX'} & if\ ACT\_LOSPL_{r,v,p,'FX'}\ given \\ \times RS\_FR_{r,s,ts} & \end{pmatrix}$$

6.3.4 Equation: EQ_ACTFLO

Indices: **region (r), vintage year (v), milestone year (t), process (p), time slice (s)**

Type: =

Related variables: **VAR_ACT, VAR_FLO, VAR_IRE**

Related equations: **EQ_COMBAL, EQ_CAPACT, EQ_PTRANS**

Purpose: This equation defines the VAR_ACT activity variable in terms of the "primary flows" of a process. The primary flows are defined by the user through the **prc_actunt** set attribute.

Remarks:

- The internal set **rtp_vintyr** ensures that (v,t) expressions are generated for the vintaged processes and (t,t) for the non-vintaged ones.
- The constraint defines the activity of a process. The activity of a process is limited in the equation EQ(l)_CAPACT by the available capacity.
- **rtp_vara(r,t,p)** controls the valid periods in which the process can operate.
- **rp_aire(r,p)** controls which sides of an import/export process should define activity
- If the activity of a process is defined by a single flow, the flow variable is replaced by the activity variable in case that the reduction algorithm is activated. Then, in all equations where the flow occurs, the activity variable is used instead. In this case the equation EQ_ACTFLO is not generated.

Equation:

$$EQ\_ACTFLO_{r,v,t,p,s} \quad \ni \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \wedge \mathbf{prc\_ts}_{\mathbf{r,p,s}} \wedge \mathbf{rtp\_vara}_{\mathbf{r,t,p}}$$

IF NOT **rpc_ire**

The process is not an inter-regional process

$$VAR\_ACT_{v,t} = \sum_{c \in \mathbf{prc\_actunt}} \frac{VAR\_FLO_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}}$$

IF **rpc_ire**

The process is an inter-regional trade process.

$$VAR\_ACT_{t,v} = \sum_{c \in \mathbf{prc\_actunt}} \frac{\sum_{ie \in \mathbf{rp\_aire}} VAR\_IRE_{r,v,t,p,c,s,ie}}{PRC\_ACTFLO_{r,v,p,c}}$$

6.3.5 Equation: EQ_ACTPL

Indices: **region (r), vintage year (v), period (t), process (p), time slice (s)**

Type: =

Related variables: **VAR_ACT, VAR_UPS**

Related equations: **EQE_ACTEFF**

Purpose: This equation defines the variable proportional to the efficiency loss under partial loads, if endogenous partial load efficiencies are modeled for a process, or a corresponding cost penalty under partial loads.

Remarks:

- Endogenous partial load efficiencies can only be modeled for processes that have their efficiency modelled by the ACT_EFF parameter.
- The input parameter ACT_LOSPL(r,y,p,'FX') defines the proportional increase in specific fuel consumption at the minimum operating level, when modelling partial load efficiencies endogenously (for process p, vintage y, region r).
- The input parameter ACT_LOSPL(r,y,p,'LO') defines the minimum operating level used for the partial load efficiency function; default value is taken from ACT_UPS(r,y,p, 'ANNUAL','FX'), but if neither is specified, is set to 0.1.
- The input parameter ACT_LOSPL(r,y,p,'UP') defines the fraction of the feasible load range above the minimum operating level, below which the efficiency losses are assumed to occur; default value = 0.6.
- It is recommended that the minimum operating level is defined by the ACT_MINLD(r,v,p) parameter, which is then used as the default value for ACT_LOSPL(r,y,p,'LO'). However, if desired, the minimum level to be assumed can also be defined by explicitly specifying ACT_LOSPL('LO').
- When the ACT_CSTPL input parameter is defined instead of (or as a supplement to) ACT_LOSPL, the cost coefficient is applied in the objective function directly to the $VAR\_UPS_{r,v,t,p,s,'FX'}$ variable as defined by the EQ_ACTPL equation.

Notation:

- $AF\_MIN_{r,v,p,s}$ minimum operating level of online capacity of process ***p***, vintage ***v*** in timeslice ***s***, as defined by ACT_MINLD (default) or ACT_LOSPL('LO');
- $PL\_LDL_{p,v}$ the load level below which partial load efficiency losses start to occur for process ***p***, vintage ***v***;
- *SUP(s)* is the set of timeslices above timeslice ***s*** in the timeslice tree, but including also ***s*** itself;
- *UPS(p)* is the set of timeslices with start-ups/shut-downs allowed for process *p*.

Equations:

$$EQ\_ACTPL_{r,v,t,p,s,} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_LOSPL_{r,v,p,'FX'} > 0))$$

$$VAR\_UPS_{r,v,t,p,s,'FX'} \geq$$

$$\left(\begin{array}{l} COEF\_CPT_{r,v,t,p} \left(VAR\_NCAP_{r,v,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \right) \times \\ PL\_LDL_{r,v,p} \cdot PRC\_CAPACT_{r,p} \cdot G\_YRFR_s - VAR\_ACT_{r,v,t,p,s} \end{array} \right) \times$$

$$\frac{AF\_MIN_{r,v,p,ANNUAL}}{PL\_LDL_{r,v,p} - AF\_MIN_{r,v,p,ANNUAL}}$$

6.3.6 Equation: EQ_ACTRAMP

Indices: **region (r), vintage year (v), period (t), process (p), time slice (s), bound (bd)**

Type: =

Related variables: **VAR_ACT, VAR_NCAP, VAR_UPS**

Related equations: **EQE_CAPLOAD**

Purpose: This equation defines maximum ramp-up and ramp-down rates for a standard process. The maximum ramp-rates are specified with the input parameter *ACT_UPS*(r,v,p,s,'UP') and *ACT_UPS*(r,v,p,s,'LO'), as fractions of the nominal on-line capacity per hour.

Remarks:

- The amount of on-line capacity is the full available capacity, unless start-ups / shut-downs have been enabled by using the parameter *ACT_MINLD*.

Notation:

- *SUP(s)* is the set of timeslices above timeslice ***s*** in the timeslice tree, but including also ***s*** itself
- *UPS(p)* is the set of timeslices with start-ups/shut-downs allowed for process *p*.

Equations:

$$EQ\_ACTRAMP_{r,v,t,p,s,'UP'} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,'UP'} > 0))$$

$$\left(\frac{VAR\_ACT_{r,v,t,p,s}}{G\_YRFR_{r,s}} - \frac{VAR\_ACT_{r,v,t,p,s-1}}{G\_YRFR_{r,s-1}} - \left(VAR\_UPS_{r,v,t,p,s,'UP'} - VAR\_UPS_{r,v,t,p,s,'LO'}\right) \cdot ACT\_UPS_{r,v,p,s,'FX'}\right) \times$$

$$\frac{2 \cdot RS\_STGPRD_{r,s}}{8760 \times \left(G\_YRFR_{r,s} + G\_YRFR_{r,s-1}\right)} \leq \left(VAR\_NCAP_{r,v,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'}\right) \times$$

$$COEF\_CPT_{r,v,t,p} \times PRC\_CAPACT_{r,p} \times ACT\_UPS_{r,v,p,s,'UP'}$$

$$EQ\_ACTRAMP_{r,v,t,p,s,'LO'} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,'LO'} > 0))$$

$$\left(\frac{VAR\_ACT_{r,v,t,p,s-1}}{G\_YRFR_{r,s-1}} - \frac{VAR\_ACT_{r,v,t,p,s}}{G\_YRFR_{r,s}} - \left(VAR\_UPS_{r,v,t,p,s,'LO'} - VAR\_UPS_{r,v,t,p,s,'UP'}\right) \cdot ACT\_UPS_{r,v,p,s,'FX'}\right) \times$$

$$\frac{2 \cdot RS\_STGPRD_{r,s}}{8760 \times \left(G\_YRFR_{r,s} + G\_YRFR_{r,s-1}\right)} \leq \left(VAR\_NCAP_{r,v,p} - \sum_{ts \in SUP(s-1) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'}\right) \times$$

$$COEF\_CPT_{r,v,t,p} \times PRC\_CAPACT_{r,p} \times ACT\_UPS_{r,v,p,s,'LO'}$$

6.3.7 Equation: EQL_ACTUPC

Indices: **region (r), vintage year (v), period (t), process (p), timeslice level (tsl), lim_type (l), time slice (s)**

Type: $\leq$

Related variables: **VAR_UPS, VAR_NCAP, VAR_RCAP**

Related equations: **EQE_ACTUPS, EQ_CAPLOAD**

Purpose: This equation has two purposes, according to the lim_type (l):

1. It defines a lower limit for consecutive on-line / off-line hours of process capacity, such that capacity started-up cannot be immediately shut down again, or capacity shut-down cannot be immediately started up again. This purpose is served when lim_type=LO/UP.
2. It defines a maximum number of start-up cycles for the process capacity within the timeslice cycle under the parent timeslice.

Remarks:

- The minimum on-line / off-line hours are defined by using the input attribute $ACT\_TIME_{r,v,p,bd}$, where bd = LO/UP. The maximum number of start-up cycles is defined by using the input attribute $ACT\_TIME_{r,v,p,'N'}$.

Notation:

- *SUP(s)* is the set of timeslices above timeslice ***s*** in the timeslice tree, but including also ***s*** itself,
- *UPS(p)* is the set of timeslices with start-ups/shut-downs allowed for process *p*,
- *P(s)* and *C(s)* refer to the parent timeslice of ***s*** and the set of child timeslices of ***s***, respectively.

Equations:

Case A: Lower limit for consecutive on-line / off-line hours

$$EQ\_ACTUPC_{r,v,t,p,tsl,'UP',s} \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge \mathbf{ts\_group}_{r,tsl,s} \wedge (ACT\_TIME_{r,t,p,'UP'} > 0))$$

$$\sum_{ts \in C(P(s))} VAR\_UPS_{r,v,t,p,ts'UP'} \times (\mathrm{mod}(Hour(s) - Hour(ts), 24) < ACT\_TIME_{r,v,p,'UP'}) \leq$$

$$\left(VAR\_NCAP_{r,v,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \right)$$

$$EQ\_ACTUPC_{r,v,t,p,tsl,'LO',s} \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge \mathbf{ts\_group}_{r,tsl,s} \wedge (ACT\_TIME_{r,t,p,'LO'} > 0))$$

$$\sum_{ts \in C(P(s))} VAR\_UPS_{r,v,t,p,ts,'LO'} \times (\mathrm{mod}(Hour(s) - Hour(ts),24) < ACT\_TIME_{r,v,p,'LO'}) \leq \left(VAR\_UPS_{r,v,t,p,s,'N'}\right)$$

Case B: Maximum number of start-up cycles within parent timeslice cycle

$$EQ\_ACTUPC_{r,v,t,p,tsl,'N',s} \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \left(s \in \bigcup_{ts} \left\{ P(ts) \mid \mathbf{prc\_ts}_{r,p,ts} \right\} \right) \wedge \mathbf{ts\_group}_{r,tsl,s} \wedge (ACT\_UPS_{r,v,p,'ANNUAL','N'} > 0))$$

$$\sum_{ts \in C(s)} VAR\_UPS_{r,v,t,p,ts,'UP'} \leq \left(VAR\_NCAP_{r,v,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \right) \times ACT\_UPS_{r,v,p,'ANNUAL','N'}$$

6.3.8 Equation: EQE_ACTUPS

Indices: **region (r), vintage (v), period (t), process (p), timeslice level (tsl), timeslice (s)**

Type: =

Related variables: **VAR_UPS**

Related equations: **EQL_ACTUPS, EQ_CAPLOAD**

Purpose: This equation establishes the relation between start-ups/shut-downs and the change in the amount of on-line capacity between successive timeslices. It is generated only when start-up costs have been defined for a standard process with *ACT_CSTUP*.

Notation:

- $UPS^{+}(r,p,tsl)$ is the set of timeslice levels with start-ups/shut-down costs defined for process p.

Equation:

$$EQE\_ACTUPS_{r,v,t,p,tsl,s} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge UPS^{+}_{r,p,tsl} \wedge \mathbf{ts\_group}_{r,tsl,s})$$

$$VAR\_UPS_{r,v,t,p,s,'UP'} - VAR\_UPS_{r,v,t,p,s,'LO'}$$

$$=$$

$$VAR\_UPS_{r,v,t,p,s-1,'N'} - VAR\_UPS_{r,v,t,p,s,'N'}$$

6.3.9 Equation: EQL_ACTUPS

Indices: **region (r), vintage year (v), period (t), process (p), timeslice level (tsl), lim_type (l), time slice (s)**

Type: $\leq$

Related variables: **VAR_UPS**

Related equations: **EQE_ACTUPS, EQ_CAPLOAD**

Purpose: This equation ensures that startup costs are being consistently applied when start-up costs have been defined on multiple timeslice levels.

Notation:

- $UPS^{+}(r,p,tsl)$ is the set of timeslice levels with start-ups/shut-down costs defined for process **p**.
- $P(r,s)$ refers to the parent timeslice of ***s*** in region **r**.

Equations:

Case A: lim_type='N'

$$EQL\_ACTUPS_{r,v,t,p,tsl,'N',s} \ni \left(\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge UPS^{+}_{r,p,tsl} \wedge \mathbf{ts\_group}_{r,tsl,s}\right)$$

$$VAR\_UPS_{r,v,t,p,s,'N'} \leq VAR\_UPS_{p,v,t,P(s),'FX'}$$

Case B: lim_type='FX'

$$EQL\_ACTUPS_{r,v,t,p,tsl,'FX',s} \ni \begin{pmatrix} \mathbf{rtp\_vintyr}_{r,v,t,p} \wedge UPS^{+}_{r,p,tsl} \wedge \\ \mathbf{ts\_group}_{r,tsl,s} \wedge s \in \left\{ \bigcup_{sl} P(sl) \mid sl \in UPS^{+}(p) \right\} \end{pmatrix}$$

$$VAR\_UPS_{r,v,t,p,s,'FX'} \leq \sum_{ts \in C(s)} VAR\_UPS_{r,v,t,p,ts,'UP'}$$

6.3.10 Equation: EQ(*l*)_ASHAR

Indices: **region (r), vintage year (v), period (t), process (p), time slice (s)**

Type: As determined by the bd index of the input parameter FLO_SHAR:

- *l* = 'G' for **bd** = 'LO' yields ≥,
- *l* = 'L' for **bd** = 'UP' yields ≤,
- *l* = 'E' for **bd** = 'FX' yields =.

Related variables: **VAR_FLO, VAR_ACT, VAR_SOUT**

Related equations: **EQ(*l*)_INSHR, EQ(*l*)_OUTSHR**

Purpose: A share equation between process flows/activity is generated a process (**p**) in region (**r**) for time period (**t**) and each time-slice (**s**). The equation is similar to the equations EQ(l)_INSHR and EQ(l)_OUTSHR, but is only generated when the input parameter $FLO\_SHAR_{r,v,p,c,cg,s,bd}$ is specified in a non-standard way, for a commodity **c** and group **cg** such that **c** is not a member of group **cg**, or such that **c**=**cg**.

Remarks:

- Internally, the non-standard *FLO_SHAR*s are converted into the corresponding *FLO_ASHAR* parameters.
- In general, the constraint is generated on the level of the process flow variable for **c**, unless **c**='ACT', which will result in an annual level constraint.
- When **c**='ACT', the equation defines a bound on the amount of activity in proportion to the flows in the group **cg**, on the ANNUAL level.
- When **cg**='ACT', the equation defines a bound on the amount of flow of **c** in proportion to the activity, on the level of the process flow variable for **c**.
- When **c**=**cg**, and **c** is a member of the default shadow group, the share equation is generated for the flow of **c** in the total flow of commodities in the SPG, and either on the group level or on the WEEKLY level, whichever is higher. This feature makes it easy to define e.g. daily share constraints for a DAYNITE level process, such as fuel shares for plug-in hybrid cars.
- When the process is a storage process, the only valid share specification is $FLO\_SHAR_{r,v,p,c,'ACT',s,bd}$, where c is the discharge commodity of a timeslice storage. This generates a constraint between the output flow and the storage activity, which can be useful e.g. for preventing the use of the storage for a by-pass operation. The **cg** is set automatically to the SPG when the *FLO_SHAR* is converted into *FLO_ASHAR.*

$$EQ(l)\_ASHAR_{r,v,t,p,c,cg,s} \quad \ni \left(\begin{array}{l} \mathbf{rtp\_vintyr_{r,v,t,p}} \wedge \left(\mathbf{rpcs\_var_{r,p,c,s}} \vee \left((c='ACT') \wedge \mathbf{annual_s} \right) \right) \\ \wedge \sum_{ts\_map_{r,s,ts}} FLO\_ASHAR_{r,v,p,c,cg,ts,bd} \end{array} \right)$$

Case A: Standard processes:

$$\sum_{\mathbf{rps\_s2_{r,p,sl}}} FLO\_ASHAR_{r,v,p,c,cg,sl,bd} \times RS\_FR_{r,s,sl} \times$$

$$\left(\begin{array}{l} \sum_{com \in cg} \sum_{\mathbf{rtpcs\_varf_{r,t,p,com,ts}}} VAR\_FLO_{r,v,t,p,com,ts} \times RTCS\_TSFR_{r,t,p,com,sl,ts} + \\ \sum_{com \in \left\{ \begin{array}{c} \mathbf{rpc\_pg_{r,p,com}} \\ |cg='ACT' \end{array} \right\}} \sum_{\mathbf{prc\_ts_{r,p,ts}}} \frac{VAR\_FLO_{r,v,t,p,com,ts}}{PRC\_ACTFLO_{r,v,p,com}} \times RTCS\_TSFR_{r,t,p,com,sl,ts} \end{array} \right)$$

$$\{=;\leq;\geq\}$$

$$\sum_{\mathbf{rtpcs\_varf_{r,t,p,c,ts}}} VAR\_FLO_{r,v,t,p,c,ts} \times RTCS\_TSFR_{r,t,p,c,s,ts} +$$

$$\sum_{com \in \left\{ \begin{array}{c} \mathbf{rpc\_pg_{r,p,com}} \\ |c='ACT' \end{array} \right\}} \sum_{\mathbf{prc\_ts_{r,p,ts}}} \frac{VAR\_FLO_{r,v,t,p,com,ts}}{PRC\_ACTFLO_{r,v,p,com}} \times RTCS\_TSFR_{r,t,p,com,s,ts}$$

Case B: Storage processes:

$$\sum_{\mathbf{rps\_s2_{r,p,sl}}} FLO\_ASHAR_{r,v,p,c,'ACT',sl,bd} \times RS\_FR_{r,s,sl} \times$$

$$\left(\sum_{com \in \left\{ \begin{array}{c} \mathbf{rpc\_pg_{r,p,com}} \\ |cg='ACT' \end{array} \right\}} \sum_{\mathbf{prc\_ts_{r,p,ts}}} VAR\_FLO_{r,v,t,p,com,ts} \times RTCS\_TSFR_{r,t,p,com,sl,ts} \right)$$

$$\{=;\leq;\geq\}$$

$$\sum_{\substack{\mathbf{rpc\_stg_{r,p,c}} \\ \mathbf{rpcs\_var_{r,p,c,ts}}}} \frac{VAR\_SOUT_{r,v,t,p,c,ts}}{PRC\_ACTFLO_{r,v,p,c}} \times RTCS\_TSFR_{r,t,p,c,s,ts}$$

6.3.11 Equation: EQ(*l*)_BLND

Indices: **region (r), year (t), refinery product (ble), specification (spe)**

Type: Any type, as determined by the value of the input parameter BL_TYPE(r,ble,spe):

- *l* = 'L' for a value of 1 yields $\leq$.
- *l* = 'G' for a value of 2 yields $\geq$.
- *l* = 'E' for a value of 3 yields $=$.

Related variables: **VAR_BLND**

Related equations: **EQ_COMBAL**

Purpose: The blending equations ensure that the characteristics of petroleum products (e.g. sulfur content, density, octane number, etc.) lie within specified limits, if desired.

Remarks:

- Parameter BL_COM contains the values of the blending specifications **spe** for the blending streams **opr**.
- Parameter BL_SPEC contains the value of the specification **spe** of the blending product **ble**.
- The blending variables VAR_BLND are expressed in volume units. If the characteristics of the blending streams **opr** and the product **ble** are not given in volume units (indicated by input parameter REFUNIT), the user has to provide a conversion parameter CONVERT which contains the density and energy content (by weight or by volume) of each blending stream. The conversion parameters are used to derive the coefficients RU_CVT of the blending streams in the blending equation.

Equation:

$$EQ(l)\_BLND_{r,t,ble,spe} \qquad \ni bl\_type_{r,ble,spe}$$

$$\sum_{opr \in ble\_opr_{r,ble,opr}} BL\_COM_{r,ble,opr,spe} \cdot RU\_CVT_{r,ble,spe,opr} \cdot VAR\_BLND_{r,t,ble,opr}$$

$$\{\leq;=;\geq\}$$

$$\sum_{opr \in ble\_opr_{r,ble,opr}} BL\_SPEC_{r,ble,opr,spe} \cdot RU\_CVT_{r,ble,spe,opr} \cdot VAR\_BLND_{r,t,ble,opr}$$

6.3.12 Equation: EQ_BNDCST

Indices: **region (r), year1 (y1), period (t), year2 (y2), cost aggregation (costagg), currency (cur)**

Type: =

Related variables: **VAR_CUMCST**

Related equations: **EQ_OBJ**

Purpose: This equation is generated when a bound is specified on regional costs, taxes and/or subsidies, either cumulative over a year range (using $REG\_CUMCST_{r,y1,y2,agg,,cur,bd}$) or in given milestone years (using $REG\_BNDCST_{r,y,agg,cur,bd}$). It sets the level of the variable $VAR\_CUMCST_{r,y1,y2,costagg,cur}$ equal to the cost expression, to be bounded accordingly.

Remarks:

- The available cost aggergations that can be bounded are listed in the table below.
- All the cost components related to investments are expressed in terms of annualized capital costs, i.e. as annuities paid in the year(s) in question. These components thus include interest during both construction and payback time.
- In all combined cost aggregations, subsidies are treated as negative costs when summed up with other cost/taxes, but when bounded alone they are treated as positive.

Cost aggregation ID	Description
INV	investment costs (annuities)
INVTAX	investment taxes (annuities)
INVSUB	investment subsidies (annuities)
INVTAXSUB	investment taxes-subsidies (annuities)
INVALL	= INV+INVTAXSUB (annuities)
FOM	fixed OM costs
FOMTAX	fixed operating taxes
FOMSUB	fixed operating subsidies
FOMTAXSUB	fixed operating taxes-subsidies
FOMALL	= FOM+FOMTAXSUB
FIX	= INV+FOM
FIXTAX	= INVTAX+FOMTAX
FIXSUB	= INVSUB+FOMSUB
FIXTAXSUB	= FIXTAX-FIXSUB
FIXALL	= FIX+FIXTAXSUB
COMTAX	commodity taxes
COMSUB	commodity subsidies
COMTAXSUB	commodity taxes-subsidies
FLOTAX	process commodity flow taxes
FLOSUB	process commodity flow subsidies
FLOTAXSUB	process commodity flow taxes-subsidies
ALLTAX	= FIXTAX+COMTAX+FLOTAX
ALLSUB	= FIXSUB+COMSUB+FLOSUB
ALLTAXSUB	= ALLTAX-ALLSUB

Notation:

- INVTAX(r,y) = the tax portion of the (virtual) variable INVTAXSUB
- INVSUB(r,y) = the subsidy portion of the (virtual) variable INVTAXSUB
- FIXTAX(r,y) = the tax portion of the (virtual) variable FIXTAXSUB
- FIXSUB(r,y) = the subsidy portion of the (virtual) variable FIXTAXSUB
- COMTAX(r,y) = the commodity tax portion of the (virtual) variable VARTAXSUB
- COMSUB(r,y) = the commodity subsidy portion of the (virtual) variable VARTAXSUB
- FLOTAX(r,y) = the flow tax portion of the (virtual) variable VARTAXSUB
- FLOSUB(r,y) = the flow subsidy portion of the (virtual) variable VARTAXSUB
- $\mathbf{cost\_map_{agg,costagg}}$ = mapping coefficient between all cost aggregations and the component aggregations to be summed up (value = 0 / 1 / –1).

Remark: See the Section on the objective function for details on the expressions for the (virtual) cost variables mentioned above.

Equation:

$$EQ\_BNDCST_{r,y1,t,y2,agg,cur} \quad \ni \left(\left(\sum_{bd} REG\_CUMCST_{r,y1,y2,agg,cur,bd} <> 0 \right) \wedge \left(t = \underset{\{tt|M(tt-1)<y2\}}{\arg\max} (M(tt)) \right) \right)$$

$$\begin{aligned}
VAR_CUMCST_{r,y1,y2,agg,cur} = \\
&\sum_{y1 \le y \le y2} INVCOST(r,y) \times \left(\mathbf{cost_map_{agg,'INV',type}}\right) \\
&\sum_{y1 \le y \le y2} INVTAX(r,y) \times \left(\mathbf{cost_map_{agg,'INVTAX'}}\right) \\
&\sum_{y1 \le y \le y2} INVSUB(r,y) \times \left(\mathbf{cost_map_{agg,'INVSUB'}}\right) \\
&\sum_{y1 \le y \le y2} FIXCOST(r,y) \times \left(\mathbf{cost_map_{agg,'FOM'}}\right) \\
&\sum_{y1 \le y \le y2} FIXTAX(r,y) \times \left(\mathbf{cost_map_{agg,'FOMTAX'}}\right) \\
&\sum_{y1 \le y \le y2} FIXSUB(r,y) \times \left(\mathbf{cost_map_{agg,'FOMSUB'}}\right) \\
&\sum_{y1 \le y \le y2} COMTAX(r,y) \times \left(\mathbf{cost_map_{agg,'COMTAX'}}\right) \\
&\sum_{y1 \le y \le y2} COMSUB(r,y) \times \left(\mathbf{cost_map_{agg,'COMSUB'}}\right) \\
&\sum_{y1 \le y \le y2} FLOTAX(r,y) \times \left(\mathbf{cost_map_{agg,'FLOTAX'}}\right) \\
&\sum_{y1 \le y \le y2} FLOSUB(r,y) \times \left(\mathbf{cost_map_{agg,'FLOSUB'}}\right)
\end{aligned}$$

6.3.13 Equation: EQ(*l*)_BNDNET/PRD

Indices: region (r), period (t), commodity (c), timeslice (s)

Type: Any type, as determined by the bound index **bd** of COM_BNDNET/PRD:

- *l* = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- *l* = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- *l* = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Purpose: If the bound on the net or gross production of a commodity is specified for a timeslice being above the timeslice level of the commodity, the equation described here is generated. The bound on the net or gross production of a commodity is directly applied to the variable (VAR_COMNET, VAR_COMPRD), if the bound parameter is specified for a commodity timeslice (**com_ts**).

Remarks:

- The internal set **rcs_comts** used in the equation contains all timeslices at or above the timeslice level being defined for the commodity.
- The internal set **rtcs_varc** used in the summation part of the equation contains all timeslices (out of **com_ts**) and periods for which the commodity is available.
- The internal set **ts_map(r,s,ts)** used in the summation part of the equation contains for a given timeslice (**s**) all timeslices (**ts**) being at or below **s** in the timeslice tree.

Interpretation of the results:

Primal: Value of the net production of a commodity (production minus consumption)

Dual: marginal cost of increasing the bound by one unit

Equation

$$EQ(l)\_BND(NET/PRD)_{r,t,c,s} \ni \left\{ \mathbf{rcs\_comts}_{\mathbf{r,c,s}} \wedge (NOT\ \mathbf{com\_ts}_{\mathbf{r,c,s}}) \wedge COM\_BND(NET/PRD)_{r,t,c,s,bd} \right\}$$

$$\sum_{ts \in \mathbf{rtcs\_varc}_{\mathbf{r,t,c,ts}} \cap \mathbf{ts\_map}_{\mathbf{r,s,ts}}} VAR\_COM(NET/PRD)_{r,t,c,ts}$$

$$(\leq / \geq / =)$$

Sign according to the *l* equation index (must coincide with the **bd** index in parameter COM_BNDNET/PRD).

$$COM\_BND(NET/PRD)_{r,t,c,s,bd}$$

6.3.14 Equation: EQ(*l*)_CAFLAC

Indices: **region (r), vintage year (v), period (t), process (p), time slice (s)**

Type: As determined by the bd index of the standard availability parameter:

- *l* = 'L' for **bd** = 'UP' yields $\le$,
- *l* = 'E' for **bd** = 'FX' yields $=$.

Related variables: **VAR_NCAP, VAR_FLO, VAR_IRE, VAR_SIN, VAR_SOUT**

Related equations: **EQ(*l*)_CAPACT, EQL_CAPFLO**

Purpose: This equation relates the flows in the primary group of a process to its available existing capacity in period **t**. The existing capacity consists of investments made in the current and previous periods (VAR_NCAP) and investment decisions that have been made exogenously (*NCAP_PASTI/PRC_RESID*). The availability of the existing capacity in a specific period **t** and timeslice **s** can be specified by the input attribute $NCAP\_AFC_{r,v,p,cg,tsl}$ / $NCAP\_AFCS_{r,v,p,cg,s}$, where cg can be a single commodity in the PG, thereby making the process availability factor dependent on the output mix.

Remarks:

- The cg index in the input attributes $NCAP\_AFC_{r,v,p,cg,tsl}$ / $NCAP\_AFCS_{r,v,p,cg,s}$ can be either a single commodity in the PG, the reserved group name 'ACT' denoting the PG itself (for other than storage processes, see below), or a commodity type of the PG. Any $NCAP\_AFCS_{r,v,p,cg,s}$ specified overrides an $NCAP\_AFC_{r,v,p,cg,tsl}$ specified on the same level, and any value specified for a single commodity overrides a value specified for a group containing the commodity.
- For storage and trade processes, which both can have the same commodity IN and OUT of the process, defining an *NCAP_AFC/NCAP_AFCS* both for the commodity type and for the single commodity itself results in the commodity availability being applied to the output flow while the group availability is applied to the input flow.
- For storage processes, defining an *NCAP_AFC/NCAP_AFCS* on the reserved group name 'ACT' defines a separate availability factor constraint (EQL_CAPFLO) for the storage level, and not for the flows in the PG.
- For trade processes, $\mathbf{prc\_map}_{\mathbf{r,'DISTR',p}}$ can be also used for removing exports from contributing to the availability equation, if the process is bi-directional.

Special notation used for the equation formulation:

- $SX_{r,v,p,c,s}$ denotes an adjustment coefficient for storage inputs:

$$SX_{r,v,p,c,s} = \begin{cases} 0 & \text{if } \mathbf{top_{OUT}}(c) \wedge \mathbf{top_{IN}}(c) \wedge \neg NCAP\_AFCS_{r,v,p,cg,s} \\ \left(\dfrac{NCAP\_AFCS_{r,v,p,c,s}}{\sum\limits_{\mathbf{rp\_pg_{r,p,cg}}} NCAP\_AFCS_{r,v,p,cg,s}} \right) & \begin{array}{l} \text{if } \mathbf{top_{OUT}}(c) \wedge \mathbf{top_{IN}}(c) \wedge \\ NCAP\_AFCS_{r,v,p,c,s} \wedge \\ NCAP\_AFCS_{r,v,p,cg,s} \end{array} \\ 1 & \text{otherwise} \end{cases}$$

- $IX_{r,v,p,c,s}$ denotes an adjustment coefficient for trade process exports:

$$IX_{r,v,p,c,s} = \begin{Bmatrix} \left(\dfrac{NCAP\_AFCS_{r,v,p,c,s}}{\sum\limits_{\mathbf{rp\_pg_{r,p,cg}}} NCAP\_AFCS_{r,v,p,cg,s}} \right) & \begin{matrix} if\ \mathbf{imp}(c) \wedge \mathbf{exp}(c) \wedge \\ NCAP\_AFCS_{r,v,p,c,s} \wedge \\ NCAP\_AFCS_{r,v,p,cg,s} \end{matrix} \\ 1 & otherwise \end{Bmatrix}$$

Equation:

$$EQ(l)\_CAFLAC_{r,v,t,p,s} \quad \ni \left(\begin{matrix} \mathbf{rtp\_vintyr_{r,v,t,p}} \wedge NCAP\_AFCS_{r,v,p,s} \wedge \\ \left(NCAP\_AF_{r,v,p,s,bd} \vee NCAP\_AFS_{r,v,p,s,bd} \vee NCAP\_AFA_{r,t,p,bd} \right) \end{matrix} \right)$$

$$\sum_{\substack{\mathbf{rpc\_pg_{r,p,c}} \\ \mathbf{prc\_ts_{r,p,ts}}}} \left(\begin{matrix} \dfrac{1}{PRC\_ACTFLO_{r,v,p,c}} \times \\ \left(\begin{matrix} 1/NCAP\_AFCS_{r,v,p,c,s} & if\ NCAP\_AFCS_{r,v,p,c,s}\ given \\ \sum\limits_{\mathbf{rp\_pg_{r,p,cg}}} \dfrac{1}{NCAP\_AFCS_{r,v,p,cg,s}} & elseif\ NCAP\_AFCS_{r,v,p,cg,s}\ given \\ 1 & otherwise \end{matrix} \right) \times \\ \left(\begin{matrix} VAR\_FLO_{r,v,t,p,c,ts} & if\ \mathbf{rp\_std_{r,p}} \\ \left(\begin{matrix} VAR\_SOUT_{r,v,t,p,c,ts} + \\ VAR\_SIN_{r,v,t,p,c,ts} \times SX_{r,v,p,c,ts} \end{matrix} \right) & if\ \mathbf{rp\_stg_{r,p}} \\ \left(\begin{matrix} VAR\_IRE_{r,v,t,p,c,ts,'IMP'} + \\ VAR\_IRE_{r,v,t,p,c,ts,'EXP'} \times IX_{r,v,p,c,ts} \end{matrix} \right) & if\ \mathbf{rp\_ire_{r,p}} \end{matrix} \right) \\ \times RTCS\_TSFR_{r,t,c,s,ts} \end{matrix} \right) +$$

$$\{\leq, =\}$$

$$\sum_{\{v2|\mathbf{rtp\_cptyr_{r,v2,t,p}}\}} \left(\begin{matrix} COEF\_AF_{r,v2,t,p,s} \times COEF\_CPT_{r,v2,t,p} \times \\ \left(\begin{matrix} VAR\_NCAP_{r,v2,p} + NCAP\_PASTI_{r,v2,p} - \\ \left(VAR\_SCAP_{r,v2,t,p} \quad if\ PRC\_RCAP_{r,p} \right) \end{matrix} \right) \\ \times PRC\_CAPACT_{r,p} \end{matrix} \right) \quad if\ \neg\mathbf{prc\_vint}_{r,p}$$

$$+ \quad \left(\begin{matrix} COEF\_AF_{r,v,t,p,s} \times COEF\_CPT_{r,v,t,p} \times \\ \left(\begin{matrix} VAR\_NCAP_{r,v,p} + NCAP\_PASTI_{r,v,p} - \\ \left(VAR\_SCAP_{r,v,t,p} \quad if\ PRC\_RCAP_{r,p} \right) \end{matrix} \right) \\ \times PRC\_CAPACT_{r,p} \end{matrix} \right) \quad if\ \mathbf{prc\_vint}_{r,p}$$

6.3.15 Equation: EQ(*l*)_CAPACT

Indices: region (r), vintage year (v), period (t), process (p), time slice (s)

Type: Determined by the bound index **bd** of NCAP_AF, NCAP_AFS or NCAP_AFA:

- *l* = 'E' for **bd** = 'FX' (fixed bound) yields =.
- *l* = 'L' for **bd** = 'UP' (upper bound) yields ≤.
- *l* = 'G' for **bd** = 'LO' (lower bound) yields ≥.

Related variables: VAR_ACT, VAR_NCAP, VAR_FLO

Related equations: EQ_ACTFLO, EQ_COMBAL, EQ_INSHR, EQ_OUTSHR, EQ_PTRANS

Purpose: The capacity-activity equation relates the activity of a process to its available existing capacity in period **t**. The existing capacity consists of investments made in the current and previous periods (VAR_NCAP) and investment decisions that have been made exogenously (NCAP_PASTI/PRC_RESID). The availability of the existing capacity in a specific period **t** and timeslice **s** is specified by the availability factor. Three availability factors exist:

- NCAP_AF(r,v,p,s,bd):
  Availability factor specified for a specific period and timeslice. If this availability factor is not specified for the process timeslices (**prc_ts**), the availabilities are aggregated/inherited according to the timeslice tree. Thus, for a process operating on the DAYNITE level it is sufficient to specify only one availability for the ANNUAL timeslice, which is then inherited to the DAYNITE timeslices.
- NCAP_AFS(r,v,p,s,bd):
  Availability factor specified for a specific period and timeslice. In contrast to NCAP_AF, this availability is not inherited/aggregated along the timeslice tree. If this availability is specified for a seasonal timeslice for a process operating on the DAYNITE level, the capacity-activity constraint is generated for the seasonal timeslice and sums over the DAYNITE activities. This gives the process flexibility how to operate within the seasonal timeslice as long as the overall seasonal availability restriction is fulfilled.
- NCAP_AFA(r,v,p,bd):
  Annual availability factor similar to NCAP_AFS being specified for the ANNUAL timeslice with the difference that NCAP_AFA is always applied in such a way as if the process is non-vintage dependent, even if it is specified as a vintaged one (**prc_vint**). Thus the annual availability factor is especially useful to calibrate the activity of a process in the first period(s) to the statistics irrespectively of its vintage structure and the vintage dependent activities (NCAP_AFS), which can be specified in addition to NCAP_AFA.

If the process is defined as a vintaged one (**prc_vint**), for each vintage year (**v**) of the existing capacity stock in period (**t**) a separate capacity-activity constraint will be generated (exception NCAP_AFA), while for a non-vintaged process one capacity-activity constraint is generated that sums over all vintage years. In the latter case the vintage index of the equation (EQ(l)_CAPACT(r,v,t,p,s)) always equals the period index (**v** = **t**).

Remarks:

- For all process timeslices (**prc_ts**), NCAP_AF(r,t,p,s,'UP') is by default set to 1, unless NCAP_AF('UP') or NCAP_AF('FX') has been specified by the user. Thus, it is ensured that the activity of a process can never exceed its capacity. If for example only NCAP_AFA is specified by the modeler as annual availability for a process with a DAYNITE timeslice resolution, in addition to the annual activity-capacity constraint activity-capacity constraints with an availability of 100% are generated for the DAYNITE process timeslices.
- An average value of the availability factors (NCAP_AF/S) over the years in the period is used when an age-dependent 'Shape' is specified for them.
- **rtp_cptyr** identifies the capacities installed in period **v** still available in period **t**. This set takes into account that investments may be turned-off for certain periods (by PRC_NOFF). The condition is as under:

v,t such that B(v) ≥ B(t) – (COEF_RPTI*TLIFE) – ILED + 1 *and* B(v) ≤ E(t) – ILED

- **prc_vint** is a set of processes for which attributes are changing over time and vintaging is required.
- Entries in **rtp_vintyr** are controlled by the same logic as applied to COEF_CPT combined with the vintaging consideration. Note **v** = **t** when no vintaging is required, or vintaging is turned off for a particular processes, where the sum over the previous investments is used instead of individual variables.
- $COEF\_AF_{r,v,t,p,s,bd}$ will be read off a pre-processed table, after application of SHAPE and MULTI to the user provided availabilities (NCAP_AF/A/S).
- COEF_AF is calculated in the following manner:
    1) aggregate if possible (pp_lvlbd.mod), otherwise inherit (in ppmain.mod)
    2) apply SHAPE and MULTI to NCAP_AF/S
- For storage processes, the capacity describes the volume of the storage and the activity the storage content. For storage processes between timeslices (**prc_tgtss**, **prc_nstts**) parameter RS_STGPRD is used instead of G_YRFR. RS_STGPRD(r,s) equals the number of storage periods on the timeslice level of timeslice **s** in the whole year multiplied with the duration of its parent timeslice **ts**. Thus, the storage level VAR_ACT (and indirectly the storage in- and output flows VAR_SIN and VAR_SOUT) are scaled-up for the entire year.

  Consequently, the value of RS_STGPRD(r,s) is:
    - 1 for a seasonal storage,
    - 365/7*G_YRFR(r,ts) for a weekly storage, where **ts** is the parent node of **s**,
    - 365*G_YRFR(r,ts) for a daynite storage, where **ts** is the parent node of **s**.

Interpretation of the results:

Primal: In case of an inequality constraint and no past investments (i.e. RHS is zero), the primal value describes the difference between the activity level and the maximum possible activity due to the installed capacity in the considered period and timeslice. If the primal value is negative, it means that the capacity is not fully utilized. In case of past investments, the RHS is not zero[42], but has a positive value and corresponds to the possible activity due to the past investments. If the primal value equals the RHS value, the capacity is fully utilized. If not the difference (RHS minus primal value), where the primal value may also be negative, describes the possible unused activity production.

Dual: The dual value is in case of an inequality constraint a negative number, when the constraint is binding. It describes the cost reduction caused by an additional capacity unit and can thus be interpreted as the value of the capacity. For a power plant for example it can be viewed as the part of the electricity price that can be used for covering the fixed operating and investment costs of the capacity (multiplied by the corresponding coefficient in the dual equation of the electricity flow variable). If NCAP_AFS or NCAP_AFA are applied for timeslices above the process timeslice level, in addition capacity-activity constraints (with a default value for NCAP_AF of 1 as upper bound) are generated for the process timeslices. The dual value of the constraints related to NCAP_AFS or NCAP_AFA serve as benchmark value of the capacity between the process timeslices. If for example NCAP_AFA is given for a power plant with a DAYNITE timeslice resolution (e.g. WD, WN, SD, SN), the NCAP_AF related capacity constraints with an availability of 1 are usually binding only in one process timeslice level, e.g. WD. Now the dual variable of NCAP_AFA can be seen as rent that must be covered in other process timeslices (WN, SD, SN) by the then prevailing electricity price, so that the model would decide to shift the scarce annual capacity from WD to another timeslice.

[42] GAMS moves all constants (e.g. past investments) on the RHS and the variables on the LHS of the equation. In the listing file the primal value of the equation can be found in the solution report under the LEVEL column. The RHS value is given under the column UPPER column in case of a <= inequality and in the LOWER column for a >= inequality. For an equality LOWER, LEVEL and UPPER value are the same.

Equation:

$$EQ(l)\_CAPACT_{r,v,t,p,s} \ni \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \wedge \mathbf{prc\_ts}_{\mathbf{r,p,s}} \wedge \mathbf{rtp\_vara}_{\mathbf{r,t,p}} \wedge \left(NCAP\_AF_{r,t,p,s} \vee NCAP\_AFS_{r,t,p,s} \vee NCAP\_AFA_{r,t,p}\right)$$

$$\left\{\begin{array}{ll} \displaystyle\sum_{ts\in\left(\mathbf{prc\_ts}_{\mathbf{r,p,ts}}\cap\mathbf{ts\_map}_{\mathbf{r,s,ts}}\right)} VAR\_ACT_{r,v,t,p,ts} & if\ \neg\mathbf{prc\_map}_{\mathbf{r,'STG'p}} \\ \displaystyle\sum_{ts\in\left(\mathbf{prc\_ts}_{\mathbf{r,p,ts}}\right)} \frac{VAR\_ACT_{r,v,t,p,ts}}{RS\_STGPRD_{r,ts}} \times RS\_FR_{r,ts,s} & if\ \mathbf{prc\_map}_{\mathbf{r,'STG'p}} \end{array}\right\}$$

$$\{\leq;=;\geq\}$$

$$\begin{array}{l} \text{Case1: Non-vintaged process(v = t):} \\ \displaystyle\sum_{\{v2|\mathbf{rtp\_cptyr}_{\mathbf{r,v2,t,p}}\}} \begin{pmatrix} COEF\_AF_{r,v2,t,p,s,bd} \times COEF\_CPT_{r,v2,t,p} \times \\ \begin{pmatrix} VAR\_NCAP_{r,v2,p} + NCAP\_PASTI_{r,v2,p} - \\ \left(VAR\_SCAP_{r,v2,t,p} \quad if\ PRC\_RCAP_{r,p}\right) \end{pmatrix} \\ \times PRC\_CAPACT_{r,p} \end{pmatrix} \quad if\ \neg\mathbf{prc\_vint}_{r,p} \\ \text{Case2: Vintaged process(v = vintage):} \\ + \quad \begin{pmatrix} COEF\_AF_{r,v,t,p,s,bd} \times COEF\_CPT_{r,v,t,p} \times \\ \begin{pmatrix} VAR\_NCAP_{r,v,p} + NCAP\_PASTI_{r,v,p} - \\ \left(VAR\_SCAP_{r,v,t,p} \quad if\ PRC\_RCAP_{r,p}\right) \end{pmatrix} \\ \times PRC\_CAPACT_{r,p} \end{pmatrix} \quad if\ \mathbf{prc\_vint}_{r,p} \end{array}$$

$$\times\left[G\_YRFR_{r,s} \times (p \notin \mathbf{prc\_map}_{\mathbf{r,'STG',p}}) + 1 \times (p \in \mathbf{prc\_map}_{\mathbf{r,'STG',p}})\right]$$

$COEF\_CPT_{r,v,t,p}$:

$$
\begin{aligned}
&\textit{if}\quad v=t\\
&= Max\begin{pmatrix}\dfrac{D(t)-NCAP\_ILED}{D(t)}\\ 0\end{pmatrix}
\end{aligned}
$$

*If **v** has been a long time period, and **t** is close enough to encounter a capacity created at the end of **v**.*

$$
\begin{aligned}
&\textit{else}\\
&\textit{if}\ \ t\geq v \wedge D(v) > IL+TL \wedge B(t) < E(v)+TL\\
&= Max\begin{pmatrix}\dfrac{Min\big(B(v)+IL+COEF\_RPTI_{r,v,p}\times TL, E(t)+1\big)-B(t)}{D(t)}\\ 0\end{pmatrix}
\end{aligned}
$$

*Number of years of existence within period **t**, divided by the period duration*

$$
\begin{aligned}
&\textit{else}\\
&= Max\begin{pmatrix}\dfrac{Min\big(B(v)+IL+TL, E(t)+1\big)-Max\big(B(v)+IL, B(t)\big)}{D(t)}\\ 0\end{pmatrix}\\
&\textit{endif}\\
&\textit{endif}
\end{aligned}
$$

This step blocks out the investments that have already retired, which may be evaluated with a negative remaining life

Where,

$$COEF\_RPTI_{r,v,p} = \left\langle \frac{D(v)-IL}{TL} \right\rangle$$

Simply counts the number of investments in a long time period.

Expression $\langle a \rangle$ is equal to the smallest integer $\geq$ a.

where:
IL = $NCAP\_ILED_{r,v,p}$
TL = $NCAP\_TLIFE_{r,v,p}$
B(t) = 1^{st} year of the period containing **t**
E(t) = Last year of the period containing **t**
D(t) = Duration of the period containing **t**

6.3.16 Equation: EQL_CAPFLO

Indices: **region (r), vintage (v), period (t), process (p), commodity (c), time slice (s)**

Type: $\leq$

Related variables: **VAR_NCAP, VAR_SCAP, VAR_FLO, VAR_ACT, VAR_UPS**

Related equations: **EQ(*l*)_CAFLAC**

Purpose: The equation defines a maximum level for a process flow (standard processes) or activity (only for storage processes) in relation to its capacity, according to an *NCAP_AFC/NCAP_AFCS* parameter specified by the user.

Remarks:

- The equation is generated only for process flows not in the PG, as the PG flows are handled by EQ_CAFLAC. However, independent EQL_CAPFLO constraints may be requested also for the PG flows by setting $NCAP\_AFC_{r,'0',p,'ACT',tsl}$ =–1.
- When defined for storage activity, note that the capacity is assumed to represent an annual production capacity equivalent to the amount produced by full power during one full year/week/day for SEASON/WEEKLY/DAYNITE level storage processes, respectively. The availability factor should be adjusted to correspond to the actual storage capacity. For example, a capacity of 1 GW is equal to 24 GWh for a DAYNITE storage, and if the real daily storage capacity is, say 8 GWh / GW, the maximum availability factor should be 0.333.
- The equation formulation is shown below is for the vintaged case only; the non-vintaged case differs in the RHS in the same way as in *EQ(l)_CAPACT*.

Notation:

- *P(s)* denotes the parent timeslice of *s*.

Equation (vintaged case only):

$$EQ\_CAPFLO_{r,v,t,p,c,s} \quad \ni \left(\mathbf{rtp\_vintyr_{r,v,t,p}} \wedge NCAP\_AFC_{r,v,p,c,s} \wedge \neg \mathbf{rpc\_pg_{r,p,c}}\right)$$

$$\begin{pmatrix} \sum\limits_{\mathbf{rtpcs\_varf_{r,t,p,c,ts}}} VAR\_FLO_{r,v,t,p,c,s} & if \;\; c \neq \text{'ACT'} \\ \sum\limits_{\mathbf{prc\_ts_{r,p,ts}}} \dfrac{VAR\_ACT_{r,v,t,p,ts} \times RS\_FR_{r,ts,s} \times G\_YRFR_{r,s}}{G\_YRFR_{r,P(ts)}} & if \;\; c = \text{'ACT'} \end{pmatrix}$$

$$\leq$$

$$NCAP\_AFC_{r,v,t,p,c,s} \times \begin{pmatrix} VAR\_NCAP_{r,v,p} + NCAP\_PASTI_{r,v,p} - \\ \sum\limits_{\mathbf{prc\_rcap_{r,p}}} VAR\_SCAP_{r,v,t,p} - \sum\limits_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \end{pmatrix} \cdot$$

$$COEF\_CPT_{r,v,p,t} \times PRC\_CAPACT_{r,p} \times G\_YRFR_{r,s}$$

6.3.17 Equation: EQ_CAPLOAD

Indices: **region (r), vintage year (v), period (t), process (p), time slice (s), lim_type (l)**

Type: $\leq$

Related variables: **VAR_ACT, VAR_NCAP, VAR_UPS**

Related equations: **EQE_ACTUPS**

Purpose: This equation is used as a replacement for the standard EQ(l)_CAPACT equations for the process timeslices, whenever flexible minimum operating limits are defined for a standard process. It defines the maximum and minimum levels of activity in relation to the available capacity, taking also into account capacity that may have been shut-down during some timeslices. The difference to the standard EQ(l)_CAPACT equations is thus that EQ_CAPLOAD refers to the on-line capacity in each timeslice, while EQ(l)_CAPACT refers to the full available capacity.

Remarks:

- The flexible minimum operating limits are defined with the parameter ACT_MINLD(r,y,p). Any fixed lower bound availability factor at the process timeslice level is ignored when ACT_MINLD is defined.
- Star-ups/shut-downs of capacity are by default only allowed on the SEASON level, and without costs. More general dispatchability features can be activated by defining start-up costs, with the parameter ACT_CSTUP. Start-up costs can be optionally defined even on the SEASON level, if desired (see the Table below).
- Start-ups and shut-downs will always occur in pairs, and therefore any shut-down costs can be directly included in the ACT_CSTUP parameter. If start-ups on some level can be assumed without additional costs, it is advisable to leave ACT_CSTUP unspecified at that level. If the start-up costs are assumed zero on some timeslice level, they must be zero also on any higher levels.

	Input parameters specified					**Resulting start-up capability**		
			ACT_CSTUP(TSLVL)			on timeslice levels		
Case	ACT_MINLD	ACT_TIME(N)	SEASON	WEEKLY	DAYNITE	SEASON	WEEKLY	DAYNITE
0	**No**	**NA**	**NA**	**NA**	**NA**	**(S)**	**(S)**	**(S)**
1	Yes	No	–	–	–	S	–	–
2	Yes	Yes	–	–	–	S	S	–
3	Yes	*	Yes	–	–	SC	–	–
4	Yes	*	–	Yes	–	S	SC	–
5	Yes	*	–	–	Yes	S	S	SC
6	Yes	*	Yes	Yes	–	SC	SC	–
7	Yes	*	Yes	–	Yes	S	S	SC
8	Yes	*	–	Yes	Yes	S	SC	SC
9	Yes	*	Yes	Yes	Yes	SC	SC	SC

S = start-ups enabled without cost
SC = start-ups enabled with costs

Notation:

- $AF\_MAX_{r,v,p,t,s}$ maximum operating level of online capacity of process ***p***, vintage ***v***, in period ***t*** and timeslice ***s***, as defined by NCAP_AF('UP')
- $AF\_MIN_{r,v,p,s}$ minimum operating level of online capacity of process ***p***, vintage ***v*** in timeslice ***s***, as defined by ACT_MINLD
- *SUP(s)* is the set of timeslices above timeslice ***s*** in the timeslice tree, but including also ***s*** itself
- *UPS(p)* is the set of timeslices with start-ups/shut-downs allowed for process *p*;

Note: Only vintaged case shown below, see the RHS of EQ_CAPACT for the differences in the non-vintaged case.

Equations:

$$EQ\_CAPLOAD_{r,v,t,p,s,UP} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,s,'FX'} > 0))$$

$$VAR\_ACT_{r,v,t,p,s} \leq$$

$$AF\_MAX_{r,v,t,p,s} \times \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} - \\ \sum\limits_{\mathbf{prc\_rcap_{r,p}}} VAR\_SCAP_{r,v,t,p} - \sum\limits_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \end{array} \right) \cdot$$

$$COEF\_CPT_{r,v,p,t} \cdot PRC\_CAPACT_{r,p} \cdot G\_YRFR_{r,s}$$

$$EQ\_CAPLOAD_{r,v,t,p,s,LO} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,s,'FX'} > 0))$$

$$VAR\_ACT_{r,v,t,p,s} \geq$$

$$AF\_MIN_{r,v,p,s} \times \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} - NCAP\_PASTI_{r,v,p} - \\ \sum\limits_{\mathbf{prc\_rcap_{r,p}}} VAR\_SCAP_{r,v,t,p} - \sum\limits_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \end{array} \right) \cdot$$

$$COEF\_CPT_{r,v,p,t} \cdot PRC\_CAPACT_{r,p} \cdot G\_YRFR_{r,s}$$

6.3.18 Equation: EQ(*l*)_CPT

Indices: region (r), period (t), process (p)

Type: Any type, as determined either by the bound index **bd** of CAP_BND or the need to have a capacity variable (learning technology or capacity variable used in user constraint):

- *l* = 'G' for **bd** = 'LO' (lower bound) yields $\geq$, if no upper bound at the same time
- *l* = 'E' for **bd** = 'FX' (fixed bound), or for lower and upper capacity bound at the same time, or for learning technology or for capacity variable used in user constraint yields $=$.
- *l* = 'L' for **bd** = 'UP' (upper bound) yields $\leq$, if no lower bound at the same time.

Related variables: VAR_NCAP, VAR_CAP

Related equations: EQ(*l*)_CAPACT

Purpose: This equation adds up the investments (VAR_NCAP), which have been made in the current and previous periods and still exist in the current period, and past investments being made before the beginning of the model horizon and either assigns it to the capacity variable VAR_CAP or applies directly lower or upper capacity bounds to it.

Remarks:

- It is generated only for those milestone year & process combinations that have a corresponding CAP_BND specification, for processes where there is a user constraint involving a capacity variable, and for processes being a learning technology (**teg**).
- In case that only a lower or an upper capacity bound is specified, the capacity bounds are directly used as RHS constants. In the other cases, the capacity variable is used instead.
- The set **rtp_varp(r,t,p)** describes the cases where a capacity variable is needed:
  - Capacity variable is used in a user constraint,
  - Lower and upper capacity bound are specified for the same period. In this case it is more efficient to generate one capacity variable by one EQE_CPT equation and bound the variable instead of generating the two equations EQL_CPT and EQG_CPT.

Equation:

$$EQ(l)\_CPT_{r,t,p} \ni CAP\_BND_{r,t,p,bd} \vee \mathbf{teg_p} \vee \mathbf{rtp\_varp}_{\mathbf{r,t,p}}$$

$$\begin{aligned} & VAR\_CAP_{r,t,p} \times \left(\mathbf{rtp\_varp}_{\mathbf{r,t,p}} \vee CAP\_BND_{r,t,p,'FX'} \vee \mathbf{teg_p} \right) \\ & + CAP\_BND_{r,t,p,'LO'} \times \left[\left(NOT\ \mathbf{rtp\_varp}_{\mathbf{r,t,p}} \right) \wedge CAP\_BND_{r,t,p,'LO'} \right] \\ & + CAP\_BND_{r,t,p,'UP'} \times \left[\left(NOT\ \mathbf{rtp\_varp}_{\mathbf{r,t,p}} \right) \wedge CAP\_BND_{r,t,p,'UP'} \right] \end{aligned}$$

$$\{\leq ; = ; \geq\}$$

$$\sum_{v \in \mathbf{rtp\_cptyr}_{\mathbf{r,v,t,p}}} COEF\_CPT_{r,v,t,p} \times \begin{pmatrix} VAR\_NCAP_{r,v,p} \times (v \in MILESTONYR) \\ + NCAP\_PASTI_{r,v,p} \times (v \in PASTYEAR) \\ - VAR\_SCAP_{r,v,t,p} \times ((r,p) \in \mathbf{prc\_rcap}) \end{pmatrix}$$

where

$COEF\_CPT_{r,v,t,p}$ *as defined in equation* $EQ(l)\_CAPACT$

6.3.19 Equation: EQ(*l*)_COMBAL

Indices: region (r), period (t), commodity (c), timeslice (s)

Type: Determined by the user-supplied set **com_lim**. Defaults are:

- *l* = 'G' (**lim** = 'LO' in **com_lim**) for energy carriers (**com_tmap**(r,c,'NRG')), demands (**com_tmap**(r,c,'DEM')), and emissions (**com_tmap**(r,c,'ENV')); yields $\geq$ type of equation; production has to be greater or equal consumption if no upper bound at the same time
- *l* = 'E' (**lim** = 'FX' in **com_lim**) for materials (**com_tmap**(r,c,'MAT')) and financial commodities (**com_tmap**(r,c,'FIN')); yields = type of equation; production has to be equal consumption if no upper bound at the same time

Related variables: VAR_ACT, VAR_FLO, VAR_COMNET, VAR_COMPRD, VAR_IRE, VAR_NCAP, VAR_SIN/OUT, VAR_BLND, VAR_ELAST

Purpose: This equation ensures that at each period and time-slice, the total procurement of a commodity balances its total disposition. A commodity may be procured in several different ways: imported, produced by technologies (activity and capacity based), released at retirement of some investments. A commodity may be disposed of in several other ways: exported, consumed by technologies (activity or capacity based) or by a demand, or "sunk" at investment time of a process. The default type for the balance constraint of an energy carrier and for an emission is ≥, which allows procurement to exceed disposition. This may be important in order to avoid some infeasibilities dues to rigid processes with many outputs or inputs. The default sign is = for materials. Both defaults may be modified by the user by the set **com_lim**.

Remarks:

- The commodity balance is generated for the timeslices (**s**) according to the user defined sets **com_tsl** or **com_ts**.
- When there are one or more of the attributes BND/CST/SUB/TAX/CUM relating to production of the commodity, EQE_COMPRD is generated in addition to this equation. EQE_COMPRD simply creates a new variable (VAR_COMPRD) equal to the production part of the LHS of the balance constraint (see expression COMSUP below)
- Similarly, if there are relevant coefficients for the net production of the commodity, the expression VAR_COMNET is created, containing the net production, and used in the RHS (see below).
- Note that CAL_FLOFLO(r,t,p,c,s,io) table stores the complete expressions (***coefficients and variables***) giving the flow of each commodity.
- The investment related input flows are assumed to be spread uniformly throughout the commodity lead-time, NCAP_CLED, ending exactly at the end of NCAP_ILED (default value for NCAP_CLED is NCAP_ILED).
- Commodity output flows related to dismantling are assumed to occur uniformly over NCAP_DLIFE, and to start right after NCAP_DLAG (default value: NCAP_DLIFE =1).

- Net/gross production of other commodities can be aggregated to the production side of the commodity balance by using the COM_AGG attribute.
- Capacity-related input/output flows can be defined with NCAP_COM, which has the 'io' index. Examples exist for (physical) consumption as well as release, land use by hydro dams and methane emissions from them, respectively.

EQ_COMBAL reads schematically as follows:
Procurement – Disposition {≥ or = } COEF_FBRHS

where COEF_FBRHS is 0 for all balance equations, except for demand balances where it is equal to a positive parameter. In addition, COEF_FBRHS is equal to a variable when the equation is used to define the variables VAR_COMPRD or VAR_COMNET.

This is expressed mathematically as the following equation, whose coefficients will be further developed in what follows.

Interpretation of the results:

Primal: In case of an inequality constraint of the commodity balance, the primal value corresponds to the value which is obtained when all terms with variables are moved to the LHS of the equation and all constants, e.g. terms with the demand parameter COM_PROJ, are moved to the RHS side. The primal value equals the value of the LHS side. Thus, the commodity balance is binding when its primal value equals its RHS constant, it is non-binding, i.e., production exceeds consumption if the primal value is greater than the RHS constant[43].

Dual: The dual variable (shadow price) of the commodity balance describes the internal value of the commodity. If the commodity balance is binding, i.e., consumption equals production, the shadow price describes the cost change in the objective function induced by an increase of the commodity demand by one unit. Since the LHS of the commodity balance describes the difference between production and consumption, this additional demand may be covered by an increase in production or by a decrease in consumption. In the first case the shadow price is determined by activities on the supply side of the commodity, while in the latter case saving measures on the demand side of the commodity are setting the shadow price. Note that when a peaking constraint (EQ_PEAK) for the considered commodity exists, the price consumers must pay during peak hours depends not only on the shadow price of the commodity balance but also on the shadow of the peaking constraint (if the flow variable of the consuming technology has the same timeslice resolution as the commodity and the peaking parameters COM_PKFLX=0 and FLO_PKCOI=1, the price to consumers is simply the sum of the two shadow prices; in other cases the dual constraint of the flow variable should be inspected to identify the correct coefficients for the two shadow prices).

[43] The primal value and the RHS constant of an equation can be found in the GAMS listing file in solution report part. The LEVEL value column corresponds to the primal value, the LOWER level value equals the RHS of a constraint of type >= and the UPPER level value equals the RHS of a constraint of a type <=.

Equation:

$$EQ(l)\_COMBAL_{r,t,c,s} \ni \left[\mathbf{rcs\_combal}_{\mathbf{r,t,c,s,bd}}\right]$$

This internal set gives the periods at which the commodity is available (usually all periods, but the user can turn off periods by the set **com_off***), and the timeslices as defined by the user in* **com_tsl** *or* **com_ts***.*

$$COM_IE_{r,t,c} \times \left(
\begin{array}{l}
\sum\limits_{p \in \mathbf{top}_{\mathbf{r,p,c,'OUT'}}} \sum\limits_{v \in \mathbf{rtp_vintyr}_{\mathbf{r,v,t,p}}} CAL_FLOFLO_{r,v,t,p,c,s,'OUT'} \\
+ \sum\limits_{p \in \mathbf{rpc_ire}_{\mathbf{r,p,c,'IMP'}}} \sum\limits_{v \in \mathbf{rtp_vintyr}_{\mathbf{r,v,t,p}}} CAL_IRE_{r,v,t,p,c,s,'IMP'} + AUX_IRE_{r,t,c,s,'OUT'} \\
+ \sum\limits_{\substack{p \in \mathbf{rpc_stg}_{\mathbf{r,p,c}} \\ (p,v) \in \mathbf{rtp_vintyr}_{\mathbf{r,v,t,p}}}} \sum\limits_{s1 \in \mathbf{prc_ts}_{\mathbf{r,p,ts}}} \left(VAR_SOUT_{r,v,t,p,c,ts} \times \left(RS_FR_{r,s,ts} \right) \times STG_EFF_{r,v,p} \right) \\
+ \sum\limits_{opr \in \mathbf{ble_opr}_{\mathbf{r,c,opr}}} \left(BLE_BAL_{r,t,c,opr} \times VAR_BLND_{r,t,c,opr} \times RTCS_TSFR_{r,t,c,s,'ANNUAL'} \right) \\
+ \left[\sum\limits_{\substack{(p,v) \in \mathbf{rpc_capflo}_{\mathbf{r,v,p,c}} \\ if\,(\mathbf{rtp_cptyr}_{\mathbf{r,v,t,p}} \\ \wedge \\ NCAP_COM_{r,v,p,c,'OUT'})}} \left(\begin{array}{l} NCAP_COM_{r,v,p,c,'OUT'} \times COEF_CPT_{r,v,t,p} \times \\ \left(\begin{array}{l} VAR_NCAP_{r,tt(v),p} + NCAP_PASTI_{r,v,p} \\ VAR_SCAP_{r,tt(v),t,p} \ni (r,p) \in \mathbf{prc_rcap} \end{array} \right) \end{array} \right) \right] \times G_YRFR_{r,s} \\
+ \left[\sum\limits_{\substack{(p,v) \in \mathbf{rpc_capflo}_{\mathbf{r,v,p,c}} \\ if\, COEF_OCOM_{r,v,t,p,c}}} \left(\begin{array}{l} COEF_OCOM_{r,v,t,p,c} \times \\ \left(\begin{array}{l} VAR_NCAP_{r,tt(v),p} + NCAP_PASTI_{r,v,p} \\ VAR_SCAP_{r,tt(v),t,p} \ni (r,p) \in \mathbf{prc_rcap} \end{array} \right) \end{array} \right) \right] \times G_YRFR_{r,s} \\
+ \sum\limits_{\substack{(com,ts) \in \\ \mathbf{com_ts}_{\mathbf{r,com,ts}}}} COM_AGG_{r,t,com,c} \times \left(\begin{array}{c} VAR_COMNET_{r,t,com,ts} \\ \text{or, if } \mathbf{com_lim}_{\mathbf{com}} = \text{FX/N:} \\ VAR_COMPRD_{r,t,com,ts} \end{array} \right) \times RTCS_TSFR_{r,t,com,s,ts}
\end{array}
\right)$$

Output flow of ordinary processes

Import of the commodity

Storage output

Output of blending process; the parameter BLE_BAL converts the blending streams to energy units

Flow produced by Technology Capacity

Flow produced by Technology Investment/Dismantling

Flow produced by commodity aggregation

This entire expression is denoted: COMSUP

$$+ \sum_{j=1}^{COM\_STEP_{r,c,'LO'}} VAR\_ELAST_{r,t,c.s,j,'LO'} - \sum_{j=1}^{COM\_STEP_{r,c,'UP'}} VAR\_ELAST_{r,t,c,s,j,'UP'}$$

Net reduction in demand

(*continued on next page*)

$$
-\left(
\begin{array}{l}
\sum_{p\in \mathbf{top}_{\mathbf{r,p,c,'OUT'}}} \quad \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_FLOFLO_{r,v,t,p,c,s,'IN'} \\
+ \sum_{p\in \mathbf{rpc\_ire}_{\mathbf{r,p,c},'EXP'}} \quad \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_IRE_{r,v,t,p,c,s,'EXP'} + AUX\_IRE_{r,t,c,s,'IN'} \\
+ \sum_{\substack{p\in \mathbf{rpc\_stg}_{\mathbf{r,p,c}} \\ (p,v)\in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}}} \quad \sum_{ts\in \mathbf{prc\_ts}_{\mathbf{r,p,ts}}} \left(VAR\_SIN_{r,v,t,p,c,ts} \times \left(RS\_FR_{r,s,ts} \right) \right) \\
+ \sum_{ble\in \mathbf{ble\_opr}_{\mathbf{r,ble,c}}} \left(BLE\_BAL_{r,t,ble,c} \times VAR\_BLND_{r,t,ble,c} \times RTCS\_TSFR_{r,t,c,s,'ANNUAL'} \right) \\
+ \left[\sum_{\substack{(p,v)\in \mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}} \\ if\ (\mathbf{rtp\_cptyr}_{\mathbf{r,v,t,p}} \\ \wedge \\ NCAP\_COM_{r,v,p,c,'IN'})}} \left(\begin{array}{l} NCAP\_COM_{r,v,p,c,'IN'} \times COEF\_CPT_{r,v,t,p} \times \\ \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \\ VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{array} \right) \end{array} \right) \right] \times G\_YRFR_{r,s} \\
+ \left[\sum_{\substack{(p,v)\in \mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}} \\ if\ COEF\_ICOM_{r,v,t,p,c}}} \left(\begin{array}{l} COEF\_ICOM_{r,v,t,p,c} \times \\ \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \\ VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{array} \right) \end{array} \right) \right] \times G\_YRFR_{r,s}
\end{array}
\right)
$$

Input flow of ordinary processes

Export of the commodity

Input flow into storage processes

Output of blending process; the parameter BLE_BAL converts the blending streams to energy units

Input flow of technology capacity

Commodity consumed by Technology Investment/Dismantling

$$\{\geq ; =\} \quad COEF\_FBRHS$$

'=' sign if (***com_type*** = MAT or FIN) or if user–defined equation type by ***com_lim*** is given

We now show the detailed calculation of the Right-hand-side

$COEF\_FBRHS:$

$Do\,Case$

$Case \ni COM\_BNDNET \vee COM\_CUMNET$
$\vee COM\_CSTNET \vee COM\_SUBNET \vee COM\_TAXNET$

$COEF\_FBRHS = VAR\_COMNET$

$Case \ni COM\_BNDPRD \vee COM\_CUMPRD$
$\vee COM\_CSTPRD \vee COM\_SUBPRD \vee COM\_TAXPRD$

$COEF\_FBRHS = VAR\_COMPRD$

$Case\ COM\_PROJ$

$COEF\_FBRHS = COM\_PROJ \times COM\_FR$

$Otherwise$

$COEF\_FBRHS = 0$

$Endcase$

Flow Coefficients related to process activity (VAR_FLO)

The process has regular flow variables (VAR_FLO).

$$CAL\_FLOFLO_{r,v,t,p,c,s,io} \ni \mathbf{rp\_flo}_{\mathbf{r,p}} \wedge NOT\ \mathbf{rpc\_conly}_{\mathbf{r,t,p,c}}$$

$$= \sum_{s1 \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,s1}}} VAR\_FLO_{r,v,t,p,c,s1} \times RTCS\_TSFR_{r,t,c,s,s1}$$

RPC_CONLY contains commodities ONLY involved in NCAP_I/O/COM

with RTCS_TSFR defined in the following way:

The TS resolution of VAR_FLO is determined by the process-commodity combination, and not by the commodity alone (see EQ_PTRANS). The set **rtpcs_varf** *contains the valid periods (t) and timeslices (s1) for which the flow variable exists.*

$$RTCS\_TSFR(r,t,c,s,s1)$$

$$IF\ \mathbf{ts\_map}_{\mathbf{r,s,s1}}$$

$$=1$$

$$ELSE$$

$$= \frac{COM\_FR_{r,t,c,s}}{COM\_FR_{r,t,c,s1}} \text{ if c is a demand commodity and } COM\_FR \text{ is specified,}$$

$$= \frac{G\_YRFR_{r,t,c,s}}{G\_YRFR_{r,t,c,s1}} \text{ otherwise.}$$

The parameter RTCS_TSFR is used to match the timeslice resolution of flow variables (VAR_FLO/VAR_IRE) and commodities. RTCS_TSFR is the coefficient of the flow variable, which is producing or consuming commodity (**c**), in the commodity balance of **c**. If timeslice **s** corresponds to the commodity timeslice resolution of **c** and timeslice **s1** to the timeslice resolution of the flow variable two cases may occur:

1) The flow variables are on a finer timeslice level than the commodity balance (first case in the formula above, **ts_map**(r,s,s1) is true): in this case the flow variables with timeslices **s** being below ts in the timeslice tree are summed to give the aggregated flow within timeslice **s1**. RTCS_TSFR has the value 1.
2) The flow variables are on coarser timeslice level than the commodity balance: in this case the flow variable is split-up on the finer timeslice level of the commodity balance according to the ratio of the timeslice duration of **s** to **s1**: RTCS_TSFR has the value = COM_FR(r,s) / COM_FR(r,s1) for demand commodities and G_YRFR(r,s) / G_YRFR(r,s1) otherwise. When COM_FR is used, the demand load curve is moved to the demand process. Thus, it is possible to model demand processes on an ANNUAL level and ensure at the same time that the process follows the given load curve COM_FR.

Inter-regional Flow Coefficients

Internal set indicating that commodity **(c)** *is imported/exported* **(ie)** *via process* **(p)** *in/from region* **(r)**.

$$CAL\_IRE_{r,v,t,p,c,s,ie} \ni \mathbf{rpc\_ire}_{\mathbf{r,p,c,ie}} \wedge NOT\ \mathbf{rpc\_conly}_{\mathbf{r,t,p,c}}$$

$$= \sum_{s1 \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,s1}}} VAR\_IRE_{r,v,t,p,c,s1,ie} \times RTCS\_TSFR_{r,t,c,s,s1}$$

Adjusts the time-slice of IRE for COM_BAL

Computes the Auxiliary flows associated with an inter-regional process

$$AUX\_IRE_{r,t,c,s,io}$$

$$=$$

$$= \sum_{(p,com,ie) \in \left(\begin{array}{l} \mathbf{rpc\_ire}_{\mathbf{r,p,com,ie}} \\ \wedge IRE\_FLOSUM_{r,t,p,com,s,ie,c,io} \end{array} \right)} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \sum_{s1 \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,com,s1}}} \left(\begin{array}{l} IRE\_FLOSUM_{r,t,p,com,s1,ie,c,io} \times VAR\_IRE_{r,v,t,p,c,s1,ie} \\ if\ \ \mathbf{ts\_map}_{\mathbf{r,s,s1}} \\ \times 1 \\ else \\ \times \dfrac{G\_YRFR_{r,s}}{G\_YRFR_{r,s1}} \end{array} \right)$$

The timeslice **(s1)** *of the flow variable VAR_IRE is below* **(s)** *in the timeslice tree.*

Since the timeslice **(s1)** *of the flow variable VAR_IRE is above* **(s)** *in the timeslice tree, VAR_IRE is apportioned according to the timeslice durations.*

Investment Related Flow Coefficients

Intermediate Notation:

$BCF = B(v) + NCAP\_ILED - NCAP\_CLED$ Beginning year of commodity flow

$ECF = B(v) + NCAP\_ILED - 1$ Ending year of commodity flow

Note that these flows never need to be carried across 'long' periods, because the construction never exceeds the end of period v if v is 'long'

$COEF\_ICOM:$

$$if\ \ (v = t) \wedge (IL + TL < D(t))$$

$$= COEF\_RPTINV \times \frac{NCAP\_ICOM_v}{D(t)}$$

Case IV

$$where\ \ COEF\_RPTINV = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

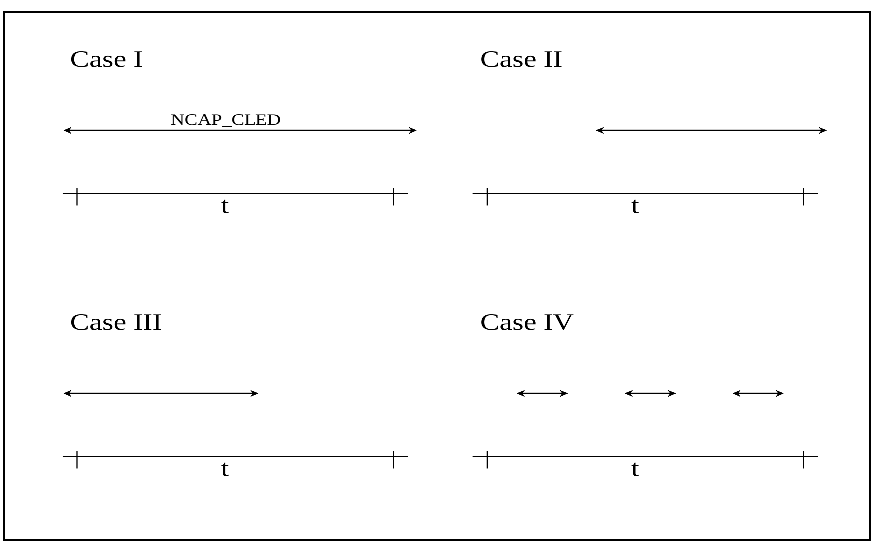
Counts the number of investments in a long period

$$else$$

$$= Max\left(\frac{1 + Min(ECF, E(t)) - Max(BCF, B(t))}{D(t)} \times \frac{NCAP\_ICOM_v}{NCAP\_CLED_v}, 0 \right)$$

Cases I, II, III

$$endif$$



Dismantling Related Flow Coefficients

Intermediate Notation:

$BCF = B(v) + NCAP\_ILED + NCAP\_TLIFE + NCAP\_DLAG$ Start year of commodity flow.

$ECF = B(v) + NCAP\_ILED + NCAP\_TLIFE + NCAP\_DLAG + NCAP\_DLIFE - 1$ End year of commodity flow.

Either the current period is 'long' or there was a long period that could have investments late enough to be dismantled in 't'.

$COEF\_OCOM:$

$$if \;\; t \geq v \wedge D(v) > IL + TL \wedge B(t) < E(v) + TL + DLAG + DLIFE$$

$$= \sum_{i=1}^{COEF\_RPTINV} \left(Max \left(\begin{array}{l} \left(\begin{array}{l} \dfrac{Min\big(B(v) + IL + (i \times TL) + DLAG + DLIFE - 1, E(t)\big)}{D(t)} \\ - \dfrac{Max\big(B(v) + IL + (i \times TL) + DLAG, B(t)\big)}{D(t)} \end{array} \right) \\ 0 \end{array} \right) \right) \times \frac{NCAP\_OCOM_v}{NCAP\_DLIFE_v}$$

else

$$= Max \left(\begin{array}{l} \dfrac{1 + Min(ECF, E(t)) - Max(BCF, B(t))}{D(t)} \times \dfrac{NCAP\_OCOM_v}{NCAP\_DLIFE_v} \\ 0 \end{array} \right)$$

endif

6.3.20 Equation: EQE_COMPRD

Indices: region (r), period (t), commodity (c), timeslice (s)

Type: =

Related variables: VAR_ACT, VAR_FLO, VAR_COMNET, VAR_COMPRD, VAR_IRE, VAR_NCAP, VAR_SOUT, VAR_BLND, VAR_ELAST

Related equations: EQ(l)_COMBAL, EQ(l)_BNDPRD, EQ(l)_CUMPRD, EQ_OBJVAR

Purpose: This equation generates a variable VAR_COMPRD equal to the total supply of the commodity, i.e. import + production (activity and capacity based) + investment-time outflow + dismantling related outflows, in each period and time slice. Note that this excludes demand reduction (in the case of a demand commodity).

Remarks:

- Enables the application of bounds to the annual or cumulative production of commodities. This is also needed to incorporate cost/sub/tax attributes on commodity production.

Equation:

$$EQE\_COMPRD_{r,t,c,s} \ni COM\_BNDPRD \vee COM\_CUMPRD$$
$$\vee COM\_CSTPRD \vee COM\_SUBPRD \vee COM\_TAXPRD$$

$$COMSUP = VAR\_COMPRD_{r,t,c,s}$$

This refers to the term marked "COM_SUP", on equation EQ_COMBAL

6.3.21 Equation: EQ_CUMFLO

Indices: **region (r), process (p), commodity (c), year1 (y1), year2 (y2)**

Type: =

Related variables: **VAR_ACT, VAR_FLO, VAR_CUMFLO**

Related equations: **EQ_CUMNET, EQ_CUMPRD**

Purpose: This equation is generated whenever the input parameter $FLO\_CUM_{r,p,c,y1,y2}$ or $ACT\_CUM_{r,p,y1,y2}$ has been specified, for bounding the cumulative amount of process flow or activity. It is also generated when the input parameter *UC_CUMFLO* or *UC_CUMACT* has been specified. It sets the variable $VAR\_CUMFLO_{r,p,c,y1,y2}$ equal to the cumulative flow/activity expression, to be bounded accordingly or to be referred to in a user constraint.

Remarks:

- The internal set $\mathbf{rpc\_cumflo_{r,p,c,y1,y2}}$ is set according to any user-defined $FLO\_CUM_{r,p,c,y1,y2}$, $ACT\_CUM_{r,p,y1,y2}$, *UC_CUMFLO or UC_CUMACT*, with the reserved commodity name 'ACT' used for *ACT_CUM* and *UC_CUMACT*.

Equation:

$$EQ\_CUMFLO_{r,p,c,y1,t,y2} \quad \ni \left(\mathbf{rpc_{r,p,c}} \wedge \mathbf{rpc\_cumflo_{r,p,c,y1,y2}}\right)$$

$if \;\; c \neq \text{'ACT'}:$

$$\sum_{t=T(y1)}^{t=T(y2)} \sum_{\substack{s \in \mathbf{rtpcs\_varf_{r,t,p,c,s}} \\ \mathbf{v} \in \mathbf{rtp\_vintyr_{r,v,t,p}}}} \left[Min\{E(t),y2\} - Max\{B(t),y1\} + 1\right] \times VAR\_FLO_{r,v,t,p,c,s} = VAR\_CUMFLO_{r,p,c,y1,y2}$$

$if \;\; c = \text{'ACT'}:$

$$\sum_{t=T(y1)}^{t=T(y2)} \sum_{\substack{s \in \mathbf{prc\_}ts_{\mathbf{r,p,s}} \\ \mathbf{v} \in \mathbf{rtp\_vintyr_{r,v,t,p}}}} \left[Min\{E(t),y2\} - Max\{B(t),y1\} + 1\right] \times VAR\_ACT_{r,v,t,p,s} = VAR\_CUMFLO_{r,p,c,y1,y2}$$

Bounds:

$VAR\_CUMFLO.LO_{r,p,'ACT',y1,y2} = ACT\_CUM_{r,y1,y2,'LO'}$

$VAR\_CUMFLO.UP_{r,p,'ACT',y1,y2} = ACT\_CUM_{r,y1,y2,'UP'}$

$VAR\_CUMFLO.FX_{r,p,'ACT',y1,y2} = ACT\_CUM_{r,y1,y2,'FX'}$

$VAR\_CUMFLO.LO_{r,p,c,y1,y2} = FLO\_CUM_{r,p,c,y1,y2,'LO'}$

$VAR\_CUMFLO.UP_{r,p,c,y1,y2} = FLO\_CUM_{r,p,c,y1,y2,'UP'}$

$VAR\_CUMFLO.FX_{r,p,c,y1,y2} = FLO\_CUM_{r,p,c,y1,y2,'FX'}$

6.3.22 Equation: EQ_CUMNET/PRD

Indices: **region (r), year1 (y1), year2 (y2), commodity (c)**

Type: =

Related variables: **VAR_COMNET/VAR_COMPRD, VAR_CUMCOM**

Related equations: **EQ(*l*)_COMBAL, EQE_COMPRD**

Purpose: This equation defines a variable representing the cumulative amount of net release or total gross production of a commodity, primarily for bounding the variable according to the bound parameter COM_CUMNET/PRD. The constraint concerns net release/production over an arbitrary number of consecutive years between the year (y1) and year (y2) as given in the parameter COM_CUMNET/PRD.

Remarks:

- It is possible to have multiple cumulative bounds of any type.
- The total time span for calculating the cumulative production need not consist of an exact number of periods.
- The cumulative bounds are expressed annually only.
- The sign of the bound is indicated by the **l** equation index.

Interpretation of the results:

Primal: The primal value describes the cumulative net release/the cumulative production of commodity **c** between the years **y1** and **y2**.

Dual: The dual value of the constraint describes the change in the objective function if the bound parameter is increased by one unit. The increase of an upper bound yields a reduction of the total costs (dual value is negative), since the system wants to use more of this commodity. The increase of a lower bound yields an increase of the total costs (dual value is positive), since the system has to be forced to use more of an uncompetitive commodity (the commodity itself or the technologies utilizing it maybe too expensive). The dual value of a cumulative production constraint can also be interpreted as a tax/subsidy that is applied between the years **y1** and **y2** to reach the same cumulative productions as specified in the bound (the tax/subsidy has to be adjusted by the discount rate).

Equation:

$$EQ(l)\_CUMNET_{r,y1,y2,c} \ni COM\_CUMNET_{r,y1,y2,c,l}$$

$$\sum_{t=T(y1)}^{t=T(y2)} \sum_{s \in \mathbf{rtcs\_varc}_{\mathbf{r,t,c,s}}} \left[Min\{E(t),y2\} - Max\{B(t),y1\} + 1\right] \times VAR\_COMNET_{r,t,c,s} = VAR\_CUMCOM_{r,c,'NET',y1,y2}$$

The internal set **rtcs_varc** *gives the periods at which the commodity is available (usually all periods, but the user can turn off periods by the set* **com_off***), and the timeslices as defined by the user in* **com_tsl** *or* **com_ts***.*

$$EQ(l)\_CUMPRD_{r,y1,y2,c,s} \ni COM\_CUMPRD_{r,y1,y2,c,l}$$

$$\sum_{t=T(y1)}^{t=T(y2)} \sum_{s \in \mathbf{rtcs\_varc}_{\mathbf{r,t,c,s}}} \left[Min\{E(t),y2\} - Max\{B(t),y1\} + 1\right] \times VAR\_COMPRD_{r,t,c,s} = VAR\_CUMCOM_{r,c,'PRD',y1,y2}$$

Bounds:

$VAR\_CUMCOM.LO_{r,c,'NET',y1,y2} = COM\_CUMNET_{r,y1,y2,c,'LO'}$

$VAR\_CUMCOM.UP_{r,c,'NET',y1,y2} = COM\_CUMNET_{r,y1,y2,c,'UP'}$

$VAR\_CUMCOM.FX_{r,c,'NET',y1,y2} = COM\_CUMNET_{r,y1,y2,c,'FX'}$

$VAR\_CUMCOM.LO_{r,c,'PRD',y1,y2} = COM\_CUMPRD_{r,y1,y2,c,'LO'}$

$VAR\_CUMCOM.UP_{r,c,'PRD',y1,y2} = COM\_CUMPRD_{r,y1,y2,c,'UP'}$

$VAR\_CUMCOM.FX_{r,c,'PRD',y1,y2} = COM\_CUMPRD_{r,y1,y2,c,'FX'}$

6.3.23 Equation: EQ_CUMRET

Indices: **region (r), vintage year (v), period (t), process (p)**

Type: =

Related variables: **VAR_RCAP, VAR_SCAP**

Related equations: **EQ_DSCRET**, **EQ_SCAP**

Purpose: This equation defines the relation between the early retirements of capacity occurring in each period t and the cumulative retirements over all periods tt ≤ t, by vintage. Its main purpose is to define the early retirements of capacity by each period, in order to be able to bound them directly with the attribute RCAP_BND.

Equation:

$$EQ\_CUMRET_{r,v,t,p} \quad \ni (\mathbf{rtp\_cptyr}_{r,v,t,p} \wedge \mathbf{prc\_rcap}_{r,p})$$

$$VAR\_SCAP_{r,v,t,p} = VAR\_RCAP_{r,v,t,p} + \sum_{t-1 \in \{tt | \mathbf{rtp\_cptyr}_{r,v,tt,p}\}} VAR\_SCAP_{r,v,t-1,p}$$

6.3.24 Equation EQ_DSCNCAP

Indices: region (r), milestone year (t), process (p)

Type: =

Related variables: VAR_DNCAP, VAR_SNCAP, VAR_NCAP

Related equations: EQ_DSCONE

Purpose: The investment variable of the technology p in period t and region r can take only specific unit sizes given by the parameter NCAP_DISC. This equation defines the investment variable to be equal to the sum over the different unit sizes each multiplied by the corresponding decision variable VAR_DNCAP. However, the sister equation EQ_DSCONE restricts this sum to a single term only (i.e. a single unit – of a specific size – is allowed to be invested in at period **t**). Alternatively, if NCAP_SEMI is defined, the equation defines the investment variable to be equal to the semi-continuous variable VAR_SNCAP.

Remarks:

- The set **unit** contains the names of capacity blocks/units that can be added, the set contain integer numbers going from ‘0’ to ‘100’. The unit name ‘0’ is used to describe the decision that no capacity should be added.
- The set **prc_dscnap(r,p)** contains the processes **p** (in region **r**) for which the discrete capacity formulation should be used
- The parameter **NCAP_DISC(r,t,p,u)** is the allowed capacity size of unit u; e.g. the size of unit ‘1’ could be 50 MW, unit ‘2’ 100 MW and unit ‘3’ 500 MW. The size of unit ‘0’ is automatically set to zero (EPS). If all unit sizes are taken equal, the formulation allows the repeated investment of a basic unit (as many as 100 times, in integer numbers).
- The parameter **NCAP_SEMI**(r,t,p) can alternatively be used for defining the investment variable VAR_NCAP semi-continuous, with the lower bound defined by *NCAP_SEMI*, and upper bound by *NCAP_BND*. If *NCAP_BND* is not defined, upper bound is assumed equal to the lower bound.
- VAR_DNCAP(r,t,p,u) is a binary decision variable describing whether the capacity unit of technology **p** should be added in period **t** or not. Some solvers for mixed-integer problems, as CPLEX or XPRESS, allow the definition of variables as so-called SOS1 sets (special ordered sets) in order to improve the solution process. An SOS1 set is defined as a set of variables of which only one variable can take a non-zero value. VAR_DNCAP is currently defined as an SOS1 variable. Not all solvers support this option, in these cases the variable type should be changed to a binary variable in the file **mod_vars.dsc**.

Equation:

$$EQ\_DSCNCAP_{r,t,p} \quad \ni \left(\mathbf{rp\_dscncap}_{\mathbf{r,p}} \wedge \mathbf{rtp}_{\mathbf{r,t,p}}\right)$$

$$VAR\_NCAP_{r,t,p} = \sum_{u \in \mathbf{unit}} \left(VAR\_DNCAP_{r,t,p,u} \times NCAP\_DSC_{r,t,p,u}\right) + \left(VAR\_SNCAP_{r,t,p} \quad if \;\; NCAP\_SEMI_{r,t,p} \; given\right)$$

6.3.25 Equation: EQ_DSCONE

Indices: **region (r), milestoneyear (t), process (p)**

Type: =

Related variables: VAR_DNCAP, VAR_NCAP

Related equations: EQ_DSCNCAP

Purpose: The equation ensures that only one of the multiple unit sizes allowed for technology **p** (described by NCAP_DSC(r,t,p,u)) can be added in period t.

Equation

$$EQ\_DSCONE_{r,t,p} \ni \left(\mathbf{rp\_dscncap}_{\mathbf{r,p}} \wedge \mathbf{rtp}_{\mathbf{r,t,p}}\right)$$

$$\sum_{u \in \mathbf{unit}} VAR\_DNCAP_{r,t,p,u} = 1$$

$$\textit{Note that } VAR\_DNCAP \textit{ must be declared as a binary variable (taking values 0 or 1 only)}$$

6.3.26 Equation: EQ_DSCRET

Indices: **region (r), vintage year (v), period (t), process (p)**

Type: =

Related variables: **VAR_NCAP, VAR_SCAP, VAR_DRCAP**

Related equations: **EQ_CUMRET**

Purpose: This equation defines the cumulative early retirement variable *VAR_SCAP* to be a multiple of a user-defined block size, specified by *RCAP_BLK*. The amount of capacity retired early can thus only take discrete values $n \times RCAP\_BLK,\ n=0,1,2,3,\ldots$.

Remarks:

- Because the residual capacity can be defined rather freely by PRC_RESID, a forced component (*RTFORC*) is added into the cumulative retirements for processes having existing capacities defined with PRC_RESID, corresponding to the trajectory given by PRC_RESID.
- Because it should always be possible to retire the remaining residual capacity in full (regardless of the block size specified), that amount is added as a second alternative block size, which can only be retired in a multiple of 1.

Equation:

$$EQ\_DSCRET_{r,v,t,p} \quad \ni (\mathbf{rtp\_cptyr}_{r,v,t,p} \wedge RCAP\_BLK_{r,v,p})$$

$$\begin{aligned} VAR\_SCAP_{r,v,t,p} - RTFORC_{r,v,t,p} = \\ & RCAP\_BLK_{r,v,p} \times VAR\_DRCAP_{r,v,t,p,2} + \\ & (NCAP\_PASTI_{r,v,p} - RTFORC_{r,v,t,p}) \times VAR\_DRCAP_{r,v,t,p,1} \end{aligned}$$

6.3.27 Equation: EQ(*l*)_FLOBND

Indices: region (r), period (t), process (p), commodity group (cg), timeslice (s)

Type: Any type, as determined by the bound index **bd** of FLO_BND:

- l = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- l = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- l = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Purpose: Bound on the sum of process flows in a given commodity group (**cg**) for a particular process (**p**) in period (**t**) and timeslice (**s**).

Remarks:

- The constraint bounds the flows in a specific period (**t**) irrespectively of the vintage years of the process capacity.
- The bound can be defined for a single commodity or a group of commodities linked to the process (**p**). In the latter case, a commodity group (**cg**) must be defined by the user (through **com_gmap**).
- The constraint is generated if one of the following conditions is true:
    - Process (**p**) is vintaged, or
    - The sum of several process flows given by the commodity group (**cg**), and not only a single process flow, should be bounded, or
    - The timeslice resolution of the flow variables are below the timeslice (**s**) of the bound parameter.

  In other cases, the bound can be directly applied to the corresponding flow variable, so that no extra equation is needed.
- The timeslice level (**s**) of the bound must be at or higher than the timeslice level of the process flows (**rtpcs_varf**).
- If *FLO_BND* is defined for a trade process, the constraint bounds the sum of imports and exports when **cg** is a single commodity, but net imports if **cg** is a true commodity group (i.e. is itself not a commodity).

Interpretation of the results:

Primal: If the primal value equals the bound parameter, the constraint is binding.

Dual: The dual value describes for a lower/upper bound the cost increase/decrease in the objective function, if the bound is increased by one unit. It may also be interpreted as subsidy/tax needed to reach the given bound value.

Notation used in formulation:

- $XS_{cg,ie}$ denotes a sign coefficient for trade flows, such that $XS_{cg,'IMP'} = 1$ for all cg, $XS_{c,'EXP'} = 1$ for all c, and $XS_{cg,'EXP'} = -1$ for all other (true) cg.

Equation:

The process is vintaged.

*The timeslice resolution (**ts**) of the process flow(s) is below the timeslice resolution (**s**) of the bound.*

The bound is applied to a true commodity group, no to a commodity.

$$EQ(l)\_FLOBND_{r,t,p,cg,s} \quad \ni \left\{ \begin{array}{l} \mathbf{rtp}_{\mathbf{r,t,p}} \wedge FLO\_BND_{r,t,p,cg,s,bd} \wedge \\ \left(\mathbf{prc\_vint}_{\mathbf{r,p}} \vee \sum\limits_{c \in \mathbf{com\_gmap}_{\mathbf{r,cg,c}}} \sum\limits_{ts \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \mathbf{rs\_below}_{\mathbf{r,s,ts}} \vee \neg \mathbf{com}_{r,cg} \right) \end{array} \right\}$$

$$\sum_{c \in \mathbf{com\_gmap}_{\mathbf{r,cg,c}}} \sum_{ts \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left(\begin{array}{ll} VAR\_FLO_{r,v,t,p,c,ts} & \text{if } \mathbf{rp\_flo}_{\mathbf{r,p}} \\ \sum\limits_{ie} VAR\_IRE_{r,v,t,p,c,ts,ie} \times XS_{cg,ie} & \text{if } \mathbf{rp\_ire}_{\mathbf{r,p}} \end{array} \right)$$

$$(\leq / \geq / =) \qquad FLO\_BND_{r,t,p,cg,s,bd}$$

where the equation sign is indicated by equation index **l** based on the bound type **bd**.

6.3.28 Equation: EQ(*l*)_FLOFR

Indices: region (r), period (t), process (p), commodity (c), timeslice (s)

Type: Any type, as determined by the bound index **bd** of FLO_FR:

- *l* = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- *l* = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- *l* = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Purpose: **1)** Relationship in period (**t**) between the total annual flow and the flow in a particular timeslice (**s**) for a specific process (**p**). This is the standard usage of the *FLO_FR* parameter, which may be used even for defining a full load curve for a process flow.
2) Relationship in period (**t**) between the the flow level in a particular flow timeslice (**s**) and the average level under all timeslices under its parent timeslice for a specific process (**p**). This variant will only be used when *FLO_FR* is levelized to the flow timeslices (**rpcs_var**), which is triggered by defining any *FLO_FR* value for that process flow at the ANNUAL level.

Remarks:

The sign of the equation determines whether the flow in a given timeslice is rigidly (=) or flexibly (≥ ; ≤) linked to the annual flow (or the parent flow level). The constraint bounds the flows irrespectively of the vintage years of the process capacity.

Equation:

Case A: Standard EQ(*l*)_FLOFR: fraction of flow in total ANNUAL flow

$$EQ(l)\_FLOFR_{r,t,p,c,s} \quad \ni \left\{ \sum_{ts \in \mathbf{rpcs\_var}_{\mathbf{r,p,c,ts}}} \mathbf{ts\_map}_{\mathbf{r,s,ts}} \wedge FLO\_FR_{r,t,p,c,s,bd} \right\}$$

*The timeslices of the process flow (**ts**) have to be below the timeslice (**s**) of the bound.*

$$\sum_{ts \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left(VAR\_FLO_{r,v,t,p,c,ts} \times RTCS\_TSFR_{r,t,c,s,ts} \right)$$

$$(\leq / \geq / =)$$

$$\sum_{ts \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left[VAR\_FLO_{r,v,t,p,c,ts} \times FLO\_FR_{r,t,p,c,s,bd} \right]$$

See under EQ(l)_COMBAL for the definition of the internal parameter RTCS_TSFR .

where the equation sign is indicated by equation index **l**.

Case B: Levelized EQ(*l*)_FLOFR: flow level in proportion to average level under parent

$$EQ(l)\_FLOFR_{r,t,p,c,s} \quad \ni \left\{\mathbf{rtpcs\_var}_{\mathbf{r,t,p,c,s}} \wedge FLO\_FR_{r,t,p,c,s,bd}\right\}$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left(\frac{VAR\_FLO_{r,v,t,p,c,s}}{G\_YRFR_{r,s}}\right)$$

$$(\leq / \geq / =)$$

$$\sum_{ts \in \mathbf{rs\_below1}_{\mathbf{r,ts,s}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left(\frac{\sum_{sl \in \mathbf{rs\_below1}_{\mathbf{r,ts,sl}}} VAR\_FLO_{r,v,t,p,c,sl}}{G\_YRFR_{r,ts}}\right) \times FLO\_FR_{r,t,p,c,s,bd}$$

where the equation sign is indicated by equation index **l**.

6.3.29 Equation: EQ(*l*)_FLMRK

Indices: **region (r), period (t), process (p), commodity (c), time-slice (s)**

Type: Any type, as determined by the bound index **bd** of FLO_MARK/PRC_MARK:

- *l* = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- *l* = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- *l* = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Related variables: VAR_FLO, VAR_IRE, VAR_SIN/SOUT, VAR_COMPRD

Related equations: **EQ(*l*)_COMBAL, EQE_COMPRD**

Purpose: Relationship to facilitate constraints on the market share of process (p) in the total production of commodity (c). Indicates that the flow of commodity (c) from/to process (p) is bounded by the given fraction of the total production of commodity (c). The time-slice level of the constraint is that of the commodity (c) when using FLO_MARK, and ANNUAL when using PRC_MARK. The same given fraction is applied to all timeslices.

Variables involved:

- **VAR_FLO(r,v,t,p,com,s)** – the average flow to/from a process built in period **v**, during time-slice **s**, during each year of period **t**. The variable for an input flow appears on the consumption side of the balance equation without any coefficients, and the variable for an output flow on the production side multiplied with the commodity efficiency (COM_IE).
- **VAR_IRE(r,v,t,p,com,s,ie)** – the average flow to/from an exchange process built in period **v**, during time-slice **s**, during each year of period **t**. The export variable appears on the consumption side of the balance equation without any coefficients, and the import variable on the production side multiplied by the commodity efficiency (COM_IE).
- **VAR_SIN/SOUT(r,v,t,p,c,s)** – flows entering/leaving a storage process **p** storing a commodity **c**. The variable for charging appears on the consumption side of the balance equation without any coefficients; the import variable on the production side multiplied by both the storage efficiency and commodity efficiency (SGT_EFF, COM_IE).
- **VAR_COMPRD(r,t,com,s)** – variable equal to the total import + production (activity and capacity based) in each period and time slice. This balance is defined by the equation EQE_COMPRD, which is automatically generated for all commodities used in the FLO_MARK or PRC_MARK parameters.

Parameters:

- **FLO_MARK(r,t,p,c,l)** – Market share of single process in total production of commodity **c**.
- **PRC_MARK(r,t,p,grp,c,l)** – Market share of a group **grp** of processes in total production of commodity **c**.

Remarks:

1. All the FLO_MARK parameters are internally converted to PRC_MARK parameters by the model generator, using the process name of the FLO_MARK parameter as the process group index (**grp**) in PRC_MARK. Therefore, below references to the parameters are mostly given in terms of PRC_MARK only.
2. Market-share constraints can be specified for standard processes, as well as for exchange and storage processes. For standard processes, the PRC_MARK parameter value can be unambiguously applied to the process flow, and the value should normally be non-negative. However, because exchange and storage processes may have both input and output flows of the same commodity, for these processes the sign of the parameter value determines whether it is applied to the input or output flow, by using the following simple conventional rules:
    - Value ≥ 0: Constraint is applied to the output flow (imports or storage discharge)
    - Value ≤ 0: Constraint is applied to the negative of input flow (exports or storage charge)
    - Value=EPS: Constraint is applied to the net output flow (output–input flow)

   These simple rules provide reasonable flexibility for specifying market share bounds also for exchange and storage processes, in addition to ordinary processes. Although these rules preclude individually bounding the input or output flow to zero, this could always be accomplished by using the IRE_BND, STG_OUTBND, and STG_INBND parameters when necessary.
3. The default timeslice level of the constraint is the commodity timeslice level for the constraints defined by using FLO_MARK by the user, and ANNUAL level for those defined by using the PRC_MARK parameter. For overriding the default, see remark 4 below.
4. The commodity used in the parameter does not actually need to be in the topology, but it should contain some commodity that does exist in the process topology. This feature can be utilized for defining market-share equations at any desired timeslice level. For example, if ELC is a DAYNITE level commodity, the user could define a dummy commodity ELC_ANN that includes ELC as a group member (through COM_GMAP membership), and use the ELC_ANN commodity in the PRC_MARK parameter instead of ELC. The constraint would then be defined at the timeslice level of the ELC_ANN commodity, which is ANNUAL if not explicitly defined.
5. In the equation formulation below, the set $\mathbf{mrk\_ts}_{\mathbf{r,grp,c,s}}$ denotes the timeslices assigned to the constraints associated with group **grp** and commodity **c** in region **r**, as explained in remarks 3 and 4 above.
6. Zero market shares are either removed (for bound type 'LO') or converted into flow bounds (bound types 'UP' and 'FX'), because the formulation employs inverse values.

Examples:

- Define an upper market share bound of 5% for technology WIND1 in total ELC production in the 2010 period.
- Define an upper market share of 25% for diesel export (through exchange process DSLXHG) of total DSL production in the 2010 period. Note that because the bound is for exports, in this case the parameter value should be negative and the bound type LO instead of UP.

```
PARAMETER FLO_MARK /
        REG.2010.WIND1.ELC.UP    0.05
        REG.2010.DSLXHG.DSL.LO        –0.25
    /;
```

Interpretation of the results:

Primal: If the primal value is zero, the constraint is binding. If the primal value is positive for a lower PRC_MARK bound or negative for an upper bound, the constraint is non-binding.

Dual: The dual value describes for example for a lower bound, the subsidy needed to guarantee the market share of the technology being forced into the market. The subsidy is needed, since the production of the technology is too expensive compared to other competing technologies. The value of the subsidy, which the technology receives, is equal to (1–PRC_MARK)*(dual variable). This subsidy has to be paid by the other technologies producing the same commodity. Thus, the costs of these technologies are increased by the amount PRC_MARK*(dual variable). The constraint can therefore be interpreted as a quota system for the production of a specific technology, e.g. a certificate system for electricity by a wind technology: each non-wind producer has to buy certificates according to the quota. The price of the certificates equals the dual value of the constraint.

Equation:

$$EQ(l)\_FLMRK_{r,t,grp,c,s} \forall (r,t,grp,c,s) \in \left(\left\{\mathbf{rtpc_{r,t,p,c}} \mid PRC\_MARK_{r,t,p,grp,c,s,l} \neq 0\right\} \cap \mathbf{mrk\_ts_{r,grp,c,s}}\right)$$

$$\sum_{\substack{(com,v,ts)\in \\ RPC_p \cap COM\_GMAP_c \\ \cap RTP\_VINTYR_{p,t} \\ \cap RPCS\_VAR_p}} \left\{ \left[VAR\_FLO_{r,v,t,p,com,ts} \times \begin{bmatrix} COM\_IE_{r,com,ts} & if\ output \\ 1 & if\ input \end{bmatrix} + \right.\right.$$

$$\begin{pmatrix} VAR\_IRE_{r,v,t,p,com,ts,imp} \\ VAR\_SOUT_{r,v,t,p,com,ts} \times STG\_EFF_{r,v,p} \end{pmatrix} \times \begin{bmatrix} COM\_IE_{r,com,ts} & if\ PRC\_MARK_{r,t,p,grp,c,s,l} \geq 0 \\ 0 & if\ PRC\_MARK_{r,t,p,grp,c,s,l} < 0 \end{bmatrix} -$$

$$\left.\left. \begin{pmatrix} VAR\_IRE_{r,v,t,p,com,ts,exp} \\ VAR\_SIN_{r,v,t,p,com,ts} \end{pmatrix} \times \begin{bmatrix} 1 & if\ PRC\_MARK_{r,t,p,grp,c,s,l} \leq 0 \\ 0 & if\ PRC\_MARK_{r,t,p,grp,c,s,l} > 0 \end{bmatrix} \right] \times \left(\frac{RS\_FR_{r,s,ts}}{PRC\_MARK_{r,t,p,grp,c,s,l}} \right) \right\}$$

$$\{=;\leq;\geq\}$$

$$\sum_{\substack{com\in \\ RPC \cap COM\_GMAP_c}} \sum_{\substack{ts\in \\ RHS\_COMPRD_{t,com}}} \left\{ VAR\_COMPRD_{r,t,com,ts} \times \left(RS\_FR_{r,s,ts} \right) \right\}$$

6.3.30 Equations related to exchanges (EQ_IRE, EQ_IREBND, EQ_XBND)

The three equations in this section concern trade between regions. Since these equations involve (directly or indirectly) more than one region, we start their presentation by a complete description of the modeling approach used, which, as we shall see, involves various schemes for representing different types of trade. The description already given in Chapter 4 is also relevant to these equations.

Variables

- VAR_IRE(r, v, t, p, c, s, ie)

Description: The total amount of traded commodity (**c**) imported/exported (**ie**) to/from region (**r**), through process (**p**) vintage (**v**) in each time period (**t**)

Purpose: The trade variables facilitate trade of commodities between exporting and importing regions

Bounds: The amount of commodity imported to a region from each exporting region can be directly constrained by the IRE_BND parameter.

Remarks:

- Note that there is a one-to-one correspondence between the VAR_IRE variables and the top_ire entries (one variable for the supply region/commodity and one variable for the demand region/commodity for each instance of top_ire).
- In market-based trade, the VAR_IRE variables for the market region describe the net imports to, and exports from, the market region, not the total market volume.
- There is no variable for the total volume of the commodity market in market-based trade. The total volume can only be addressed by means of UC_IRE parameters (summing over all imports to or exports from the market).
- In market-based trade, only the amount of commodity imported to a region from the market, or exported from the region to the market, can be constrained by the IRE_BND parameter. The imports and exports thus cannot be attributed to a specific supply or demand region on the other side of the trade.
- The amount of commodity exported from / imported to a region may also be limited by various user constraints. However, unless the trade is modeled with bilateral processes, such bounds can only apply to the total exports from or imports to a region, and cannot apply to e.g. imports from a specific region.

There are only three trade equations, namely a generic trade balance equation EQ_IRE, and two bounds, EQ(l)_IREBND and EQ(l)_XBND. The generic balance equation, EQ_IRE, can be further divided into two flavors:

A. Balance equations for bilateral and other unidirectional trade into a single destination region (Cases 1 and 2).

B. Balance equations for multidirectional trade from single export region and multi-lateral market-based trade (Cases 3 and 4).

6.3.30.1 Equation EQ_IRE

Indices: region (r), year (t), process (p), commodity (c), timeslice (s)

Type: =

Related variables: VAR_IRE

Related equations: EQ(l)_IREBND, EQ(l)_XBND, EQ(l)_COMBAL, EQ_ACTFLO

Purpose: This equation defines the balance between the imports of each traded commodity (**c**) into region (**r**) and the corresponding exports through each exchange process (**p**) in each time period (**t**) and timeslice (**s**) of the process.

Units: Units of commodity traded. Normally PJ for energy, Mton or kton for materials or emissions.

Remarks:

- Flows into individual regions may be limited by the IRE_BND and IRE_XBND parameters.
- The equation has two flavors: The first one is for bilateral and unidirectional trade with a single destination region, and the second is for market-based trade and multidirectional trade from a single source region.

6.3.30.1.1 Case A. Bi-lateral or multilateral unidirectional trade to a single import region

Equation:

$$EQ\_IRE_{r,t,p,c,s} \ni \{r,t,p,c,s \in (\mathbf{rtp}_{\mathbf{r,t,p}} \wedge \mathbf{rpcs\_var}_{\mathbf{r,p,c,s}} \wedge \mathbf{rpc\_eqire}_{\mathbf{r,p,c}})\}:$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} VAR\_IRE_{r,v,t,p,c,s,'IMP'} =$$

$$\sum_{(r2,c2) \in \mathbf{top\_ire}_{\mathbf{r2,c2,r,c,p}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r2,v,t,p}}} \sum_{s2 \in IRE\_TSCVT_{r2,s2,r,s}} \sum_{ts \in \left(\begin{array}{l}\mathbf{rtpcs\_varf}_{\mathbf{r2,t,p,c2,ts}} \\ \cap\ \mathbf{rs\_tree}_{\mathbf{r,s2,ts}}\end{array}\right)} \left(\begin{array}{l} VAR\_IRE_{r2,v,t,p,c2,ts,'EXP'} \times IRE\_FLO_{r2,v,p,c2,r,c,s} \times \\ IRE\_TSCVT_{r2,s2,r,s} \times IRE\_CCVT_{r2,c2,r,c} \times \\ RTCS\_TSFR_{r2,t,c,s2,ts} \end{array} \right)$$

This is the efficiency of process **p** *for the pair of regions and commodity* **c**.

s2 *is the timeslice in region* **r2** *that corresponds to timeslice* **s** *in region* **r**. *The conversion table IRE_TSCVT contains the conversion coefficients.*

The timeslices (**ts**) *of the export flow in region* **r2** *are described by the set* **rtpcs_varf**.

Coefficients for mapping timeslices **ts** *of VAR_IRE with the timeslices* **s2**. *See EQ(l)_COMBAL for the definition of RTCS_TSFR.*

This converts the units.

Remarks:

- The IRE_TSCVT conversion coefficients are in practice provided only for some pairs of mapped timeslices between **r2** and **r**. Therefore, the timeslice conversion is actually done in two stages: First, the timeslices of the VAR_IRE variables are converted to the mapped timeslices, and then the mapped timeslices in **r2** to those in **r** as follows:
  - The mapping coefficients IRE_TSCVT do not have to be provided by the user if the timeslice definitions in both regions are identical.
  - If the timeslice definitions are different, the user provides the mapping coefficients IRE_TSCVT to convert the timeslice **s2** in region **r2** to the timeslice **s** in region **r**. Since the timeslice level of **s2** may be different from the timeslice level **ts** of the exchange variable in region **r2**, the parameter RTCS_TSFR is used to match **ts** and **s2**.
- Note that the equation is generated for each period in **rtp** only, not for each vintage in **rtp_vintyr** as in the original code. This is because **prc_vint** is region-specific. If **prc_vint** is set to YES in one region and to NO in another, that would create serious sync problems, if the equation were generated for each vintage in **rtp_vintyr**. In addition, differences in e.g. NCAP_PASTI, NCAP_TLIFE, and NCAP_AF could create sync problems, even if **prc_vint** would be set to YES in all regions.

6.3.30.1.2 Case B. Multidirectional and market-based trade between regions.

Equation:

$$EQ\_IRE_{r,t,p,c,s} \quad \ni \left\{ r,t,p,c,s \in (\mathbf{rtp}_{\mathbf{r,t,p}} \wedge \mathbf{rpcs\_var}_{\mathbf{r,p,c,s}} \wedge \mathbf{rpc\_eqire}_{\mathbf{r,p,c}}) \right\}:$$

$$\sum_{\substack{(r2,c1,c2)\in \\ (\mathbf{top\_ire}_{\mathbf{r,c1,r2,c2,p}} \cap \mathbf{top\_ire}_{\mathbf{r,c1,r,c,p}} \cap \mathbf{rpc\_market}_{\mathbf{r,p,c1}})}} \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r2,v,t,p}}} \sum_{s2\in IRE\_TSCVT_{r2,s2,r,s}} \sum_{ts\in\left(\substack{\mathbf{rtpcs\_varf}_{\mathbf{r2,t,p,c2,ts}} \\ \cap\, \mathbf{rs\_tree}_{\mathbf{r,s2,ts}}}\right)} \begin{pmatrix} VAR\_IRE_{r2,v,t,p,c2,s2,'IMP'} \times IRE\_CCVT_{r,c1,r,c} \\ \times IRE\_CCVT_{r2,c2,r,c1} \times IRE\_TSCVT_{r2,s2,r,s} \\ \times RTCS\_TSFR_{r2,t,c,s2,ts} \end{pmatrix} \times$$

$$=$$

$$\sum_{(r2,c2)\in \mathbf{top\_ire}_{\mathbf{r2,c2,r,c,p}}} \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r2,v,t,p}}} \sum_{s2\in IRE\_TSCVT_{r2,s2,r,s}} \sum_{ts\in\left(\substack{\mathbf{rtpcs\_varf}_{\mathbf{r2,t,p,c2,ts}} \\ \cap\, \mathbf{rs\_tree}_{\mathbf{r,s2,ts}}}\right)} \begin{pmatrix} VAR\_IRE_{r2,v,t,p,c2,ts,'EXP'} \times IRE\_FLO_{r2,v,p,c2,r,c,s} \times \\ IRE\_TSCVT_{r2,s2,r,s} \times IRE\_CCVT_{r2,c2,r,c} \times \\ RTCS\_TSFR_{r2,t,c,s2,ts} \end{pmatrix}$$

Remarks:

- The IRE_TSCVT conversion coefficients are in practice provided only for some pairs of mapped timeslices between ***r2*** and ***r***. Therefore, the timeslice conversion is actually done in two stages: First, the timeslices of the VAR_IRE variables are converted to the mapped timeslices, and then the mapped timeslices in ***r2*** to those in ***r***.
- In the case of market-based trading, **prc_aoff** can be used to switch off the entire commodity market for periods that fall within a range of years. It is also possible to specify multiple entries of **prc_aoff**, if, for example trading should be possible only between selected years.
- The **top_ire** entry between the export and import commodity in the market region itself is automatically defined by the TIMES model generator when necessary, i.e. there is no need to provide it by the user.

6.3.30.2 Equation: EQ(*l*)_IREBND

Indices: region (r), year (t), commodity (c), timeslice (s), region2 (all_r), import/export (ie)

Type: Any type, as determined by the bound index **bd** of IRE_BND:

- l = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- l = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- l = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Related variables: VAR_IRE

Related equations: EQ_IRE, EQ(l)_XBND, EQ(l)_COMBAL

Description*:* Sets a bound for the amount of commodity (c) imported/exported (ie) to/from region (r), from/to another region (all_r) in time period (t) and timeslice (s).

Purpose: The equation is optional and can be used to set a bound for a pair-wise inter-regional exchange. The generation of the equation is triggered by the user-specified parameter IRE_BND.

Units: Units of commodity traded. Normally PJ for energy, Mton or kton for materials or emissions.

Type: Set according to the 'l' index in IRE_BND.

Remarks:

- Total trade flows into/from individual regions may be limited by using the IRE_XBND parameter.

Interpretation of the results:

Primal: If the primal value equals the bound parameter, the constraint is binding.

Dual: The dual value describes for a lower/upper bound the cost increase/decrease in the objective function, if the bound is increased by one unit. It may also be interpreted as subsidy/tax needed to reach the given bound value.

Equation:

Case A. Imports from an external region or market region

$$EQ(l)\_IREBND_{r,t,c,s,all\_r,ie} \quad \forall \left\{ \begin{array}{l} r,t,c,s,all\_r,ie:(RCS\_COMTS_{r,c,s} \wedge \\ (\exists p: RPC\_IE_{r,p,c,ie}) \wedge IRE\_BND_{r,t,c,s,all\_r,ie}) \end{array} \right\}:$$

$$\sum_{p:(\exists c2:TOP\_IRE_{all\_r,c2,r,c,p})} \sum_{v \in RTP\_VINTYR_{r,v,t,p}} \sum_{s2} VAR\_IRE_{r,v,t,p,c,s2,exp} \times$$

$$\left(\begin{array}{ll} 1 & if\ s2 \in TS\_MAP(r,s,s2) \\ \dfrac{FR(s)}{FR(s2)} & if\ s2 \in RS\_BELOW(r,s2,s) \end{array} \right)$$

$$\{\le;=;\ge\}\ IRE\_BND_{r,t,c,s,all\_r,ie}$$

Case B. Imports from an internal non-market region

$$EQ(l)\_IREBND_{r,t,c,s,all\_r,ie} \quad \forall \left\{ \begin{array}{l} r,t,c,s,all\_r,ie:(RCS\_COMTS_{r,c,s} \wedge \\ (\exists p \in RPC\_IE_{r,p,c,ie}) \wedge IRE\_BND_{r,t,c,s,all\_r,ie}) \end{array} \right\}:$$

$$\sum_{\substack{(c2,p) \in TOP\_IRE_{all\_r,c2,r,c,p} \\ s1 \in RPCS\_VAR_{r,p,c,s1}}} \sum_{v \in RTP\_VINTYR_{all\_r,v,t,p}} \sum_{s2} VAR\_IRE_{all\_r,v,t,p,c2,s2,exp} \times IRE\_FLO_{all\_r,v,p,c2,r,c,s1} \times$$

$$IRE\_CCVT_{all\_r,c2,r,c} \times IRE\_TSCVT_{all\_r,s2,r,s1} \times \left(\begin{array}{ll} 1 & if\ s1 \in TS\_MAP(r,s,s1) \\ \dfrac{FR(s)}{FR(s1)} & if\ s1 \in RS\_BELOW(r,s1,s) \end{array} \right)$$

$$\{\le;=;\ge\}\ IRE\_BND_{r,t,c,s,all\_r,ie}$$

Case C. Exports from a non-market region to an internal or external region

$$EQ(l)\_IREBND_{r,t,c,s,all\_r,ie} \quad \forall \left\{ \begin{array}{l} r,t,c,s,all\_r,ie:(RCS\_COMTS_{r,c,s} \wedge \\ (\exists p: RPC\_IE_{r,p,c,ie}) \wedge IRE\_BND_{r,t,c,s,all\_r,ie}) \end{array} \right\}:$$

$$\sum_{p:(\exists c2:TOP\_IRE_{r,c,all\_r,c2,p})} \sum_{v \in RTP\_VINTYR_{r,v,t,p}} \sum_{s2} VAR\_IRE_{r,v,t,p,c,s2,exp} \times$$

$$\left(\begin{array}{ll} 1 & if\ s2 \in TS\_MAP(r,s,s2) \\ \dfrac{FR(s)}{FR(s2)} & if\ s2 \in RS\_BELOW(r,s2,s) \end{array} \right)$$

$$\{\le;=;\ge\}\ IRE\_BND_{r,t,c,s,all\_r,ie}$$

Case D. Exports from a market region to an internal region

$$EQ(l)\_IREBND_{r,t,c,s,all\_r,ie} \quad \forall \left\{ \begin{array}{l} r,t,c,s,all\_r,ie:(RCS\_COMTS_{r,c,s} \wedge \\ (\exists p: RPC\_IE_{r,p,c,ie}) \wedge IRE\_BND_{r,t,c,s,all\_r,ie}) \end{array} \right\}:$$

$$\sum_{(c2,p) \in TOP\_IRE_{r,c,all\_r,c2,p}} \sum_{v \in RTP\_VINTYR_{all\_r,v,t,p}} \sum_{s2} VAR\_IRE_{all\_r,v,t,p,c2,s2,exp} \times$$

$$IRE\_CCVT_{all\_r,c2,r,c} \times IRE\_TSCVT_{all\_r,s2,r,s} \times \left(\begin{array}{ll} 1 & if\ s2 \in TS\_MAP(r,s,s2) \\ \dfrac{FR(s)}{FR(s2)} & if\ s2 \in RS\_BELOW(r,s2,s) \end{array} \right)$$

$$\{\le;=;\ge\}\ IRE\_BND_{r,t,c,s,all\_r,ie}$$

Remarks:

- The IRE_TSCVT conversion coefficients are in practice provided only for some pairs of mapped timeslices between **all_r** and **r**. Therefore, the timeslice conversion is actually done in two stages: First, the timeslices of the VAR_IRE variables are converted to the mapped timeslices, and then the mapped timeslices in **all_r** to those in **r**.

6.3.30.3 Equation: EQ(l)_XBND

Indices: region (r), year (t), commodity (c), timeslice (s), imp/exp (ie)

Type: Any type, as determined by the bound index **bd** of IRE_XBND:

- l = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- l = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- l = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Related variables: VAR_IRE

Related equations: EQ(*l*)_IRE, EQ(*l*)_IREBND, EQ(*l*)_COMBAL

Description: Bound on the total amount of traded commodity (c) imported/exported (ie) to/from region (all_r) in a period (t) and timeslice (s).

Purpose: This equation bounds inter-regional or exogenous exchanges in a particular region, across all other regions.

Units: Units of commodity traded. Normally PJ for energy, Mton or kton for materials or emissions.

Remarks: Flows into/from individual regions may be limited by the IRE_BND parameter.

Interpretation of the results:

Primal: If the primal value equals the bound parameter, the constraint is binding.

Dual: The dual value describes for a lower/upper bound the cost increase/decrease in the objective function, if the bound is increased by one unit. It may also be interpreted as subsidy/tax needed to reach the given bound value.

Equation:

$$EQ(l)\_XBND_{all\_r,t,c,s,ie} \ni IRE\_XBND_{all\_r,t,c,s,ie,bd}$$

all_r is an internal

$$\sum_{p \in \mathbf{rpc\_ire}_{\mathbf{all\_r,p,c,ie}}} \sum_{s2 \in \left(\mathbf{rtpcs\_varf}_{\mathbf{all\_r,t,p,c,s2}} \cap \mathbf{rs\_tree}_{\mathbf{all\_r,s,s2}}\right)} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{all\_r,v,t,p}}} \left[\begin{array}{ll} VAR\_IRE_{all\_r,v,t,p,c,s2,ie} \times & \\ 1 & if\ s2 \in \mathbf{ts\_map}_{\mathbf{all\_r,s,s2}} \\ \dfrac{G\_YRFR(s)}{G\_YRFR(s2)} & if\ s2 \in \mathbf{rs\_below}_{\mathbf{all\_r,s2,s}} \end{array} \right]$$

$$\{=;\le;\ge\}\ IRE\_XBND_{all\_r,t,c,s,ie,bd}$$

all_r is an external

$$\sum_{p \in \mathbf{rpc\_ire}_{\mathbf{r,p,com,impex}}} \sum_{(ts,s2) \in \left(\mathbf{rs\_tree}_{\mathbf{r,ts,s2}} \cap \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,com,s2}} \cap IRE\_TSCVT_{r,ts,all\_r,s}\right)} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left[\begin{array}{l} VAR\_IRE_{r,v,t,p,com,s2,impexp} \times IRE\_CCVT_{r,com,all\_r,c} \\ \times IRE\_TSCVT(r,ts,all\_r,s) \end{array} \right]$$

All regions r with impex ≠ ie
Having import/export from/to all_r

$$\{=;\le;\ge\}\ IRE\_XBND_{all\_r,t,c,s,ie,bd}$$

6.3.31 Equations: EQ(*l*)_INSHR, EQ(*l*)_OUTSHR

Indices: region (r), year (t), process (p), commodity (c), commodity group (cg), time-slice (s)

Type: Any type, as determined by the bound index **bd** of FLO_SHAR:

- *l* = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- *l* = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- *l* = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Related variables: VAR_FLO, VAR_ACT

Related equations: EQ(*l*)_COMBAL, EQ_PTRANS, EQ_ACTFLO

Purpose: A market/product allocation constraint equation is generated for each process (**p**) for each time period (**t**) and each time-slice (**s**) in each region (if desired). It ensures that the share of an inflow/outflow of a commodity (**c**) is lower/higher/equal a certain percentage of the total consumption/production of this process for a specified commodity group (**cg**).

Quality Control Checks:

$$\sum_{c \in cg} FLO\_SHAR_{r,v,p,c,cg,s,'LO'} \forall (FLO\_SHAR_{r,v,p,c,cg,s,'LO'} \ni l = "\geq") \leq 1 - \sum_{c \in cg} FLO\_SHAR_{r,v,p,c,cg,s,'FX'}$$

$$\sum_{c \in cg} FLO\_SHAR_{r,v,p,c,cg,s,'UP'} \forall (FLO\_SHAR_{r,v,p,c,cg,s,'UP'} \ni l = "\leq") \geq 1 - \sum_{c \in cg} FLO\_SHAR_{r,t,p,c,cg,s,'FX'}$$

$$\forall FLO\_SHAR > 0$$

Remarks:

- Exchanging top(r,p,c,’IN’)=Input vs. top(r,p,c’OUT’) = Output in this equation yields EQ(l)_OUTSHR since **c** is only member of one **cg**.
- The period index of the parameter FLO_SHAR is related to the vintage period (**v**) of the process, i.e., if the process is vintaged (**prc_vint**), a constraint will be generated for each period (**t**) the installation made in the vintage period (**v**) still exists (these period pairs are internally provided by the set **rtp_vintyr**).

Interpretation of the results:

Primal: If the primal value is zero, the constraint is binding. If the primal value is positive for a lower FLO_SHAR bound or negative for an upper bound, the constraint is non-binding.

Dual: The dual value describes, for a lower bound, the subsidy needed to guarantee that the flow is at the given lower bound. The subsidy is needed, since for an output flow the shadow price of the produced commodity is too low to cover the production costs of the flow variable (for an input flow the opposite is true, the commodity is too expensive to be used in the process). The value of the subsidy that the flow receives is equal to (1-FLO_SHAR)*(dual variable). This subsidy has to be paid by the other flows forming the denominator in FLO_SHAR constraint, thus, the costs for these flows are increased by the amount FLO_SHAR*(dual variable). In a similar way, an upper bound FLO_SHAR can be interpreted as a tax being added to the costs of a flow.

Equation:

$$EQ(l)\_IN/OUTSHR_{r,v,t,p,c,cg,s} \quad \ni (c \in cg) \wedge \left(t \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}\right) \wedge \mathbf{top}_{\mathbf{r,p,c,'IN'/'OUT'}} \wedge$$

$$(s \in \mathbf{rps\_s1}_{\mathbf{r,p,s}}) \wedge FLO\_SHAR_{r,v,p,c,cg,s,bd}$$

$$FLO\_SHAR_{r,v,p,c,cg,s,bd} \times \sum_{com \in cg} \sum_{s2 \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,com,s2}}} \left[VAR\_FLO_{r,v,t,p,com,s2} \times RTCS\_TSFR_{r,t,p,com,s,s2}\right]$$

$$\{=;\leq;\geq\}$$

$$\sum_{s2 \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,s2}}} VAR\_FLO_{r,v,t,p,c,s2} \times RTCS\_TSFR_{r,t,p,c,s,s2}$$

See EQ(l)_COMBAL for the definition of RTCS_TSFR.

The set **rps_s1** *contains the timeslices of the timeslice level, which is defined to be the finest timeslice level of the process (***prc_tsl***)and all commodities (***com_tsl***) linked to the process.*

6.3.32 Equation: EQ_PEAK

Indices: region (r), period (t), commodity group (cg), time-slice (s)

Type: $\geq$

Related variables: VAR_ACT, VAR_NCAP, VAR_FLO

Related equations: EQ(*l*)_COMBAL, EQ(*l*)_CAPACT

Purpose: The commodity peaking constraint ensures that the capacity installed is enough to meet the highest demand in any timeslice, taking into consideration both adjustments to the average demands tracked by the model and a reserve margin requiring excess capacity to be installed.

Remarks:

- In the description below, the production and consumption components resemble those of the EQ(l)_COMBAL commodity balance equation, but with a peak contribution/co-incident factor applied to the terms. These factors are process dependent and as such are actually applied within the referenced expression during the summing operation.

Sets and parameters involved:

- **com_peak(r,cg)** is a flag that a peaking constraint is desired. It is optional if **com_pkts(r,cg,s)** is provided
- **com_pkts(r,cg,s)** are the explicit time slices for which peaking constraints are to be constructed. A post-optimization QC check will be done to ensure that the timeslice with highest demand is in said list. Default is all **com_ts(r,c,s)**.
- COM_PKRSV(r,t,c) is the peak reserve margin. Default 0.
- COM_PKFLX(r,t,c,s) is the difference (fluctuation) between the average calculated demand and the actual shape of the peak. Default 0
- FLO_PKCOI(r,t,p,c,s) is a factor that permits increasing the average demand calculated by the model to handle the situation where peak usage is typically higher due to coincidental usage at peak moment (e.g., air condition). Default 1 for each process consuming **c**. User can prevent a process from contributing to the calculation of the peak by specifying = 0
- NCAP_PKCNT(r,t,p,s) is the amount of capacity (activity) to contribute to the peak. Default 1 for each process producing commodity **c**. User can prevent a process from contributing to the peak by specifying = EPS
- **prc_pkaf(r,p)** switch to set NCAP_PKCNT=NCAP_AF/1 as default. Default: no
- **prc_pkno(r,p)** switch to disable process **p** from contributing to the peak by its capacity, and to disable also assigning the default value of NCAP_PKCNT for the process.
- **rpc_pkc(r,p,c)** is an internal set defined to contain those processes (p) and peaking commodities (c) that will be assumed to contribute to the peak by their capacity. Derived by the preprocessor from all those process producing commodity **c**, which either have **c** as their primary group PG or have **prc_pkaf** defined, but are in neither case included in the set **prc_pkno**.

Interpretation of the results:

Primal: When the equation is binding, the primal value of the equation is equal to the RHS constant of the equation, i.e. corresponds to the maximum output from the existing capacity in the peak timeslice, adjusted with the peak reserve requirement. When the equation is non-binding, the primal level also includes the amount of output capacity exceeding the capacity requirements during the timeslice.

Dual: The dual value of the peaking equation describes the premium consumers have to pay in addition to the commodity price (dual variable of EQ(l)_COMBAL) during the peak timeslice. The premium equals (1+COM_PKFLX)*FLO_PKCOI*RTCS_TSFR*(dual variable).

Equation:

$$EQ\_PEAK_{r,t,cg,s} \quad \ni \mathbf{com\_peak}_{\mathbf{r,cg}} \wedge s \in \mathbf{com\_pkts}_{\mathbf{r,cg,s}}$$

$$\left\{\begin{array}{l}
\sum_{c \in cg} 1/(1+COM_PKRSV_{r,t,c}) \times COM_IE_{r,t,c} \times \sum_{p \in \left(\mathbf{top}_{\mathbf{r,p,c,'OUT'}} \cup \mathbf{rpc_ire}_{\mathbf{r,p,c,'IMP'}}\right)} \\
\left[\begin{array}{l}
if \ \left(\mathbf{rpc_pkc}_{\mathbf{r,p,c}} \wedge \mathbf{prc_cap}_{\mathbf{r,p}}\right) \\
\left(G_YRFR_{r,s} \times \sum_{v \in \mathbf{rtp_cptyr}_{\mathbf{r,v,t,p}}} \left[\begin{array}{l}
NCAP_PKCNT_{r,v,p,s} \times COEF_CPT_{r,v,t,p} \\
\times \left(\begin{array}{l} VAR_NCAP_{r,tt(v),p} + NCAP_PASTI_{r,v,p} - \\ VAR_SCAP_{r,v,t,p} \times (\ni \mathbf{prc_rcap}_{\mathbf{r,p}}) \end{array}\right) \\
PRC_CAPACT_{r,p} \times PRC_ACTFLO_{r,v,p,c}
\end{array}\right] \right) \\
else \\
\sum_{v \in \mathbf{rtp_vintyr}_{\mathbf{r,v,t,p}}} CAL_FLOFLO_{r,v,t,p,c,s,'OUT'} \times NCAP_PKCNT_{r,v,p,s} \\
+ \\
\sum_{\substack{(p,c) \in \\ \mathbf{rpc_stg}_{\mathbf{r,p,c}}}} \sum_{\substack{\mathbf{rtp_vintyr}_{\mathbf{r,v,t,p}} \\ \mathbf{rpcs_var}_{\mathbf{r,p,c,ts}}}} VAR_SOUT_{r,v,t,p,c,ts} \times RS_FR_{r,s,ts} \times NCAP_PKCNT_{r,v,p,s} \\
+ \\
\sum_{v \in \mathbf{rtp_vintyr}_{\mathbf{r,v,t,p}}} CAL_IRE_{r,v,t,p,c,s,'IMP'} \times NCAP_PKCNT_{r,v,p,s} \\
- \\
\sum_{v \in \mathbf{rtp_vintyr}_{\mathbf{r,v,t,p}}} CAL_IRE_{r,v,t,p,c,s,'EXP'} \times NCAP_PKCNT_{r,v,p,s} \qquad if \ \mathbf{prc_pkno}_{\mathbf{r,p}}
\end{array}\right]
\end{array}\right\}$$

$$\geq$$

$$\sum_{c \in cg} (1 + COM\_PKFLX_{r,t,c,s}) \times$$

$$\left[\sum_{p \in \left(\mathbf{top}_{\mathbf{r,p,c,'IN'}} \cup \mathbf{rpc\_ire}_{\mathbf{r,p,c,'EXP'}}\right) \wedge \left(NOT\ \mathbf{prc\_pkno}_{\mathbf{r,p}}\right)} \left\{ \begin{array}{l} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_FLOFLO_{r,v,t,p,c,s,'OUT'} \times FLO\_PKCOI_{r,t,p,c,s} \\ + \\ \sum_{p \in \mathbf{rpc\_ire}_{\mathbf{r,p,c,'EXP'}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_IRE_{r,v,t,p,c,s,'EXP'} \times FLO\_PKCOI_{r,t,p,c,s} \\ + \\ + \left[\sum_{(p,v) \in \mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}}} \left(\begin{array}{l} NCAP\_COM_{r,v,p,c,'IN'} \times COEF\_CPT_{r,v,t,p} \times \\ \left(VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \right) \\ \left(VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \right) \end{array} \right) \right] \times G\_YRFR_{r,s} \\ \\ + \left[\sum_{(p,v) \in \mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}}} \left(\begin{array}{l} COEF\_ICOM_{r,v,t,p,c} \times \\ \left(VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \right) \\ \left(VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \right) \end{array} \right) \right] \times G\_YRFR_{r,s} \end{array} \right\} + \right.$$

$$\sum_{\substack{(p,c) \in \\ \mathbf{rpc\_stg}_{\mathbf{r,p,c}}}} \sum_{\substack{\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \\ \mathbf{rpcs\_var}_{\mathbf{r,p,c,ts}}}} VAR\_SIN_{r,v,t,p,c,ts} \times RS\_FR_{r,s,ts} +$$

$$\left. \sum_{c \in \mathbf{dem}_{\mathbf{r,c}}} \left(\begin{array}{l} COM\_PROJ_{r,t,c} \times COM\_FR_{r,t,c,s} - \\ \sum_{j=1}^{COM\_STEP_{r,c,'LO'}} VAR\_ELAST_{r,t,c,s,j,'LO'} + \sum_{j=1}^{COM\_STEP_{r,c,'UP'}} VAR\_ELAST_{r,t,c,s,j,'UP'} \end{array} \right) \right]$$

6.3.33 Equation: EQ_PTRANS

Indices: region (r), year (y), process (p), commodity group1 (cg1), commodity group2 (cg2), time-slice (s)

Type: **=**

Related variables: **VAR_FLO, VAR_ACT**

Related equations: **EQ(*l*)_COMBAL, EQ(*l*)_INSHR, EQ(*l*)_OUTSHR, EQ_ACTFLO**

Purpose: Allows specifying an equality relationship between certain inputs and certain outputs of a process e.g. efficiencies at the flow level, or the modeling of emissions that are tied to the inputs. It is generated for each process for each time period and each time-slice in each region.

Remarks:

- Internal set **rps_s1(r,p,s)**: The finer of (set of time slices of the most finely divided member of the commodities within the shadow primary group (commodities being not part of primary commodity group and on the process side opposite to the primary commodity group) and the process timeslice level (**prc_tsl**)).
- The flow variables of the commodities within the primary commodity group are modelled on the process level (**prc_tsl**). All other flow variables on the timeslice level of **rps_s1**.
- The internal parameter COEF_PTRAN(r,v,t,p,cg1,c,cg2) is the coefficient of the flow variables of commodity **c** belonging to the commodity group **cg2**. While FLO_FUNC(r,v,p,cg1,cg2,s) establishes a relationship between the two commodity groups **cg1** and **cg2**, FLO_SUM(r,v,p,cg1,c,cg2,s) can be in addition specified as multiplier for the flow variables of **c** in **cg2**.

  COEF_PTRAN is derived from the user specified FLO_FUNC and FLO_SUM parameters based on the following rules:
  - If FLO_FUNC is given between **cg1** and **cg2** but no FLO_SUM for the commodities **c** in **cg2**, it is assumed that the FLO_SUMs are 1.
  - If FLO_SUM is specified but no FLO_FUNC, the missing FLO_FUNC is set to 1.
  - If FLO_SUM(r,v,p,cg1,c,cg2) and FLO_FUNC(r,v,p,cg2,cg1,s) are specified, the reciprocal of FLO_FUNC is taken to calculate COEF_PTRAN.
- FLO_SUMs can only be specified for the flows within one commodity group **cg1** or **cg2** of EQ_PTRANS between these two commodity groups, but not for both commodity groups at the same time.
- By specifying a SHAPE curve through the parameter FLO_FUNCX(r,v,p,cg1,cg2) the efficiencies FLO_FUNC and FLO_SUM can be described as function of the age of the installation. The internal parameter RTP_FFCX contains the average SHAPE multiplier for the relevant years in a period (those years in which the installed capacity exists).

Interpretation of the results:

Primal: The primal value of the transformation is usually zero.

Dual: Due to the flexibility of the transformation equation the interpretation of its dual value depends on the specific case. For a simple case, a process with one input flow **c1** and one output flow **c2** being linked by an efficiency FLO_FUNC(c1,c2), the dual variable, which is being defined as the cost change when the RHS is increased by one unit, can be interpreted as cost change when the efficiency of the process is increased by 1/VAR_FLO(r,v,t,p,c1,s):

$$VAR\_FLO_{r,v,t,p,c2,s} - FLO\_FUNC_{r,v,t,p,c1,c2,s} \times VAR\_FLO_{r,v,t,p,c1,s} = 1$$

$$VAR\_FLO_{r,v,t,p,c2,s} - FLO\_FUNC_{r,v,t,p,c1,c2,s} \times VAR\_FLO_{r,v,t,p,c1,s} - 1 = 0$$

$$VAR\_FLO_{r,v,t,p,c2,s} - \left(FLO\_FUNC_{r,v,t,p,c1,c2,s} + \frac{1}{VAR\_FLO_{r,v,t,p,c1,s}} \right) \times VAR\_FLO_{r,v,t,p,c1,s} = 0$$

.

Equation:

$$EQ\_PTRANS_{r,v,t,p,cg1,cg2,s1} \ni (r,v,t,p) \in \left(\mathbf{rp\_flo_{r,p}} \cap \mathbf{rtp\_vintyr_{r,v,t,p}}\right) \wedge s1 \in \mathbf{rps\_s1_{r,p,s1}}$$

$$\wedge \left(s2 \in \mathbf{ts\_map_{r,s2,s1}} \wedge \begin{pmatrix} FLO\_SUM_{r,t,p,cg1,c,cg2,s2} \quad \vee \\ FLO\_FUNC_{r,t,p,cg1,cg2,s2} \wedge NOT\left(FLO\_SUM_{r,t,p,cg1,c,cg2,s2} \vee FLO\_SUM_{r,t,p,cg2,c,cg1,s2}\right) \end{pmatrix} \right)$$

$$\sum_{c \in cg2} \sum_{s \in \left(\mathbf{ts\_map_{r,s,s1}} \cap \mathbf{rtpcs\_varf_{r,t,p,c,s}}\right)} VAR\_FLO_{r,v,t,p,c,s} \times RTCS\_TSFR_{r,t,c,s1,s} \quad =$$

$$\sum_{c \in cg1} \sum_{s \in \left(\mathbf{ts\_map_{r,s,s1}} \cap \mathbf{rtpcs\_varf_{r,t,p,c,s}}\right)} \left(COEF\_PTRAN_{r,v,t,p,cg1,c,cg2,s} \times VAR\_FLO_{r,v,t,p,c,s} \times RTCS\_TSFR_{r,t,c,s1,s} \right)$$

$$\times \left(1 + RTP\_FFCX_{r,v,t,p,cg1,cg2} \times \left(if\ \mathbf{prc\_vint_{r,p}} \right)\right)$$

$$COEF\_PTRAN_{r,v,t,p,cg1,c,cg2,ts} \quad ts \in \mathbf{rpcs\_varc}_{\mathbf{r,p,c,ts}}$$

$$=$$

$$\sum_{s \in \mathbf{prc\_ts}_{\mathbf{r,p,s}}} \left(1 \times \left(if \;\; \mathbf{ts\_map}_{\mathbf{r,ts,s}} \right) + \frac{G\_YRFR_{r,ts}}{G\_YRFR_{r,s}} \times \left(if \;\; \mathbf{rs\_below}_{\mathbf{r,s,ts}} \right) \right)$$

$$\frac{FLO\_FUNC_{r,v,t,p,cg1,cg2,s}}{FLO\_FUNC_{r,v,t,p,cg2,cg1,s} \times \left(if \;\; FLO\_SUM_{r,v,t,p,cg1,c,cg2,s} \right)} \times \begin{pmatrix} if \;\; FLO\_FUNC_{r,v,t,p,cg1,cg2,s} \\ \vee FLO\_FUNC_{r,v,t,p,cg2,cg1,s} \end{pmatrix} \times$$

$$\left(1 \times \left(if \;\; NOT \; FLO\_SUM_{r,v,t,p,cg1,c,cg2,s} \right) + FLO\_SUM_{r,v,t,p,cg1,c,cg2,s} \right)$$

Calculation of SHAPE parameter RTP_FFCX

Case A: Lifetime minus construction time is longer than the construction period

$$PRC\_YMIN_{r,v,p} = B_v + NCAP\_ILED_{r,v,p}$$

$$PRC\_YMAX_{r,v,p} = PRC\_YMIN_{r,v,p} + NCAP\_TLIFE_{r,v,p} - 1$$

$$RTP\_FFCX_{r,v,t,p,cg1,c,cg2} \quad \ni FLO\_FUNCX_{r,v,p,cg1,cg2}$$

$$=$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \frac{\displaystyle\sum_{y \in \left(\mathbf{periodyr}_{\mathbf{t,y}} \wedge \left[y \le MAX(B(t), PRC\_YMAX_{r,v,p}) \right] \right)} SHAPE\left(FLO\_FUNCX_{r,v,p,cg1,cg2}, 1 + MIN\left(y, PRC\_YMAX_{r,v,p} \right) - PRC\_YMIN_{r,v,p} \right)}{MAX\left[1, MIN(E(t), PRC\_YMAX_{r,v,p}) - MAX(B(t), PRC\_YMIN_{r,v,p}) + 1 \right]} - 1$$

Case B: Lifetime minus construction time is shorter than the construction period => Investment is repeated in construction period

$$PRC\_YMAX_{r,v,p} = NCAP\_TLIFE_{r,v,p} - 1$$

$$RTP\_FFCX_{r,v,t,p,cg1,c,cg2} \quad \ni FLO\_FUNCX_{r,v,p,cg1,cg2}$$

$$=$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \frac{SHAPE\left(FLO\_FUNCX_{r,v,p,cg1,cg2}, PRC\_YMAX_{r,v,p} \right)}{PRC\_YMAX_{r,v,p}} - 1$$

6.3.34 Equations: EQL_SCAP

Indices: region (r), vintage (v), process (p), indicator (ips)

Type: ≤

Related variables: VAR_ACT, VAR_NCAP, VAR_SCAP

Related equations: **EQ_CUMRET**

Purpose: Establishes an upper bound for the cumulative retirements and salvaged capacity by process vintage, as well as for the cumulative process activity. The equation is only generated when early retirements are allowed for the process, or if the process is vintaged and a maximum operating life is specified with *NCAP_OLIFE*.

Notation:

- $RVPRL_{r,v,p}$ is defined as the time (in years) between the vintage year (v) and the last period **t** of availability for that vintage: $RVPRL_{r,v,p}$ = M(t)–M(v).

Equation:

$$EQL\_SCAP_{r,v,p,ips} \quad \ni \left(\mathbf{rtp}_{r,v,p} \wedge \left(\begin{array}{l} \left(\mathbf{prc\_rcap}_{\mathbf{r,p}} \wedge \left(\neg \mathbf{prc\_vint}_{\mathbf{r,p}} \vee \mathbf{obj\_sums}_{\mathbf{r,v,p}}\right)\right) \\ \vee \left(\mathbf{prc\_vint}_{\mathbf{r,p}} \wedge NCAP\_OLIFE_{r,v,p}\right) \end{array} \right) \right)$$

$$\left(\begin{array}{ll} \sum\limits_{t=v+RVPRL_{r,v,p}} VAR\_SCAP_{r,v,t,p} & if \;\; ips = \text{'N'} \\ \sum\limits_{\mathbf{rtp\_cptyr}_{\mathbf{r,v,t,p}}} \; \sum\limits_{\mathbf{prc\_ts}_{\mathbf{r,p,s}}} \dfrac{VAR\_ACT_{r,v,t,p,s} \times D_t}{PRC\_CAPACT_{r,p} \times NCAP\_OLIFE_{r,v,p}} & otherwise \end{array} \right)$$

$$\leq$$

$$VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} - \sum_{\substack{\mathbf{obj\_sums}_{\mathbf{r,v,p}} \\ \mathbf{prc\_rcap}_{\mathbf{r,p}}}} VAR\_SCAP_{r,v,\text{'0'},p}$$

6.3.35 Equations: EQ_STGAUX

Indices: region (r), vintage (v), period (t), process (p), commodity (c), timeslice (s)

Type: =

Related variables: VAR_ACT, VAR_FLO, VAR_SIN, VAR_SOUT

Related equations: **EQ_STGTSS, EQ_STGIPS**

Purpose: Establishes the relations between the main storage flows / activity and auxiliary storage flows.

Equation:

$$EQ\_STGAUX_{r,v,t,p,c,s} \quad \ni \left(\mathbf{rtp\_vintyr_{r,v,t,p}} \wedge \mathbf{rpcs\_var_{r,p,c,s}} \wedge \mathbf{prc\_map_{r,'STG',p}} \wedge \neg \mathbf{rpc\_stg_{r,p,c}}\right)$$

$$VAR\_FLO_{r,v,t,p,c,s} =$$

$$PRC\_ACTFLO_{r,v,p,c} \times \left(\left(\begin{array}{l} VAR\_ACT_{r,v,t-1,p,s} \times \dfrac{G\_YRFR_{r,s}}{RS\_STGPRD_{r,s}} - \\ \left(\sum\limits_{c \in \left\{ \substack{\mathbf{top}_{IN} \\ \mathbf{prc\_stgips}} \right\}} \dfrac{VAR\_SIN_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} - \sum\limits_{c \in \left\{ \substack{\mathbf{top}_{OUT} \\ \mathbf{prc\_stgips}} \right\}} \dfrac{VAR\_SOUT_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} \right) \times \\ \left(\sum\limits_{y \in \left\{ \substack{\mathbf{periodyr_{t,y}} \\ y \geq M(t)} \right\}} \left(1 - STG\_LOSS_{r,v,p,s}\right)^{(E(t)-y+0.5)} \right) \end{array} \right) \\ \times \left(\left(1 - STG\_LOSS_{r,v,p,s}\right)^{(M(t)-E(t)-Mod(D(t)/2,1))} \right)^{\left(1 \, if \, \mathbf{prc\_map_{r,'STK',p}}\right)} \right)$$

$$+ \sum_{com \in \mathbf{top}_{IN}} VAR\_SIN_{r,v,t,p,com,s} \times \left(COEF\_PTRAN_{r,v,p,com,com,c,s} \right)$$

$$+ \sum_{com \in \mathbf{top}_{OUT}} VAR\_SOUT_{r,v,t,p,com,s} \times \left(\frac{1}{COEF\_PTRAN_{r,v,p,c,c,com,s}} \; if \; COEF\_PTRAN_{r,v,p,c,c,com,s} > 0 \right)$$

6.3.36 Equation: EQ_STGTSS/IPS

Indices: **region (r), vintage year (v), period (t), process (p), time-slice (s)**

Type: "="

Related variables: VAR_FLO, VAR_ACT

Related equations: EQ(*l*)_COMBAL, EQ(*l*)_CAPACT, EQ(*l*)_STGIN/OUT

Purpose

- The model allows two kinds of storage: inter-period storage (IPS), and storage across time-slices (or time-slice storage TSS). A special type of the TSS storage is a night-storage device, which may have an input commodity different from its output commodity. The input and output commodity of a night-storage device are given by the topology set **top**.
- Storage processes are special, as they have the same commodity as input and output. Also, all other processes transform energy within their time-slices and time periods. Since topology (with the exception of night-storage devices) does not determine in/out, different variables have to be used for this purpose. Similarly, since the transformation is special, EQ_PTRANS is replaced by new equations for the two types of storage.

Sets:

- **prc_stgips(r,p,c)**: The set of inter-period storage processes. They are forced to operate annually.
- **prc_stgtss(r,p,c)**: The set of time-slice storage processes. A storage process can operate only at one particular time slice level.
- **prc_nstts(r,p,s):** The set contains the allowed charging timeslices for a night-storage device.

Variables:

- **VAR_SIN(r,v,t,p,c,s)** – the average **in** flow to a process built in period **v**, during time-slice **s**, during each year of period **t**. This variable would appear on the consumption side of the balance equation, without any coefficients.
- **VAR_SOUT(r,v,t,p,c,s)** – the average **out** flow from a process built in period **v**, during time-slice **s**, during each year of period **t**. This variable would appear on the supply side of the balance equation, multiplied by *STG_EFF* and *COM_IE*.
- **VAR_ACT(r,v,t,p,s)** – the energy stored in a storage process at the beginning of time-slice **s** (for a timeslice storage) or end of period **t** (for an inter-period storage). Note that this is a special interpretation of 'activity', to represent 'storage level.' Therefore, EQ_ACTFLO will not be generated for storage processes.
- In EQ_STGIPS only annual flows are allowed; the timeslice s index is set to ANNUAL in this case.

Equations:

- **EQ_STGTSS(r,t,p,s)** – transforms input to output for the timeslice storage processes.
- **EQ_STGIPS(r,t,p)** – transforms input to output for the interperiod storage processes.

Parameters:

- **STG_LOSS(r,v,p,s)** – annual energy loss from a storage technology, per unit of (average) energy stored.
- **STG_CHRG(r,t,p,s)** – exogenous charging of a storage technology. For timeslice storage this parameter can be specified for each period, while for interperiod storage this parameter can only be specified for the first period, to describe the initial content of the storage.

6.3.36.1 EQ_STGTSS: Storage between timeslices (including night-storage devices):

Equation:

$$EQ\_STGTSS_{r,v,t,p,s} \quad \forall (r,v,t,p,s) \in \left(\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \wedge \mathbf{rps\_stg}_{\mathbf{r,p,s}} \wedge \mathbf{prc\_map}_{\mathbf{r,'STG',p}}\right)$$

$$VAR\_ACT_{r,v,t,p,s} =$$

$$\left[\begin{array}{l}
VAR_ACT_{r,v,t,p,s-1} + \\
\sum\limits_{c \in \mathbf{rpc_stg}} \\
\quad \left(\begin{array}{l}
\text{if p is a night-storage device:} \\
\quad \dfrac{VAR_SIN_{r,v,t,p,c,s-1}}{PRC_ACTFLO_{r,v,p,c}} \times \left(if \;\; s-1 \in \mathbf{prc_nstts}_{\mathbf{r,p,s-1}} \wedge \mathbf{top}_{\mathbf{r,p,c,'IN'}}\right) - \\
\quad \dfrac{VAR_SOUT_{r,v,t,p,c,s-1}}{PRC_ACTFLO_{r,v,p,c}} \times \left(if \;\; s-1 \notin \mathbf{prc_nstts}_{\mathbf{r,p,s-1}} \wedge \mathbf{top}_{\mathbf{r,p,c,'OUT'}}\right) \\
\text{if p is not a night-storage device} \\
\quad + \sum\limits_{top_{r,p,c,'IN'}} \dfrac{VAR_SIN_{r,v,t,p,c,s-1}}{PRC_ACTFLO_{r,v,p,c}} - \sum\limits_{top_{r,p,c,'OUT'}} \dfrac{VAR_SOUT_{r,v,t,p,c,s-1}}{PRC_ACTFLO_{r,v,p,c}}
\end{array}\right) \\
- \sum\limits_{ts \in \{sl | \mathbf{prc_ts}_{r,p,sl} \cap \mathbf{rs_below}_{r,sl,s}\}} VAR_SOUT_{r,v,p,'ACT',ts} \times RS_FR_{r,s-1,ts} \\
- \left[\left(\dfrac{VAR_ACT_{r,v,t,p,s} + VAR_ACT_{r,v,t,p,s-1}}{2}\right)\right] \times STG_LOSS_{r,v,p,s} \times \dfrac{G_YRFR_{r,s}}{RS_STGPRD_{r,s}}
\end{array}\right]$$

$$+ STG\_CHRG_{r,t,p,s-1}$$

6.3.36.2 EQ_STGIPS: Storage between periods

Equation:

$$EQ\_STGIPS_{r,v,t,p} \quad \forall (r,v,t,p) \in \left(\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \cap \mathbf{prc\_map}_{\mathbf{r,'STK',p}}\right)$$

$$VAR\_ACT_{r,v,t,p,'ANNUAL'} =$$

$$\sum_{\substack{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t-1,p}} \\ p \in \mathbf{prc\_vint}}} \left[\begin{array}{l} VAR\_ACT_{r,v,t-1,p,'ANNUAL'} \\ \times (1 - STG\_LOSS_{r,v,p,'ANNUAL'})^{D(t)} \end{array}\right] + \sum_{p \notin \mathbf{prc\_vint}} \left[\begin{array}{l} VAR\_ACT_{r,t-1,t-1,p,'ANNUAL'} \\ \times (1 - STG\_LOSS_{r,v,p,'ANNUAL'})^{D(t)} \end{array}\right]$$

$$+ \left[\sum_{\substack{y \in \mathbf{periodyr}_{\mathbf{t,y}} \\ c \in \mathbf{rpc\_stg}_{r,p,c}}} \left(\begin{array}{l} \displaystyle\sum_{c \in \mathbf{top}_{IN}} \frac{VAR\_SIN_{r,v,t,p,c,'ANNUAL'}}{PRC\_ACTFLO_{r,v,p,c}} - \\ \displaystyle\sum_{c \in \mathbf{top}_{OUT}} \frac{VAR\_SOUT_{r,v,t,p,c,'ANNUAL'}}{PRC\_ACTFLO_{r,v,p,c}} \end{array}\right) \times (1 - STG\_LOSS_{r,v,p,'ANNUAL'})^{(E(t)-y+0.5)}\right]$$

$$+ \quad STG\_CHRG_{r,t,p,'ANNUAL'} (when\, ORD(t) = 1)$$

6.3.37 Equations: EQ(*l*)_STGIN / EQ(*l*)_STGOUT

Indices: region (r), period (t), process (p), commodity (c), timeslice (s)

Type: Any type, as determined by the bound index **bd** of STGIN/OUT_BND:

- *l* = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- *l* = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- *l* = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Related variables: VAR_SIN, VAR_SOUT

Related equations: EQ_STGTSS, EQ_STGIPS

Purpose: Bound on the input/output flow of a storage process of commodity (**c**) for a particular process (**p**) in period (**t**) and timeslice (**s**).

Remarks:

- The constraint bounds the flows in a specific period (**t**) irrespectively of the vintage years of the process capacity.
- The constraint is generated if one of the following conditions is true:
  - Process (**p**) is vintaged, or
  - the timeslice resolution of the flow variables (VAR_SIN/OUT) are below the timeslice (**s**) of the bound parameter.

  In other cases, the bound can be directly applied to the flow variable (VAR_SIN/SOUT), so that no extra equation is needed.
- The timeslice level (**s**) of the bound must be at or higher than the timeslice level at which the storage operates.

Interpretation of the results:

Primal: If the primal value equals the bound parameter, the constraint is binding.

Dual: The dual value describes for a lower/upper bound the cost increase/decrease in the objective function, if the bound is increased by one unit. It may also be interpreted as subsidy/tax needed to reach the given bound value.

Equation:

$$EQ(l)\_STGIN/OUT_{r,t,p,c,s} \quad \ni \left\{ \begin{array}{l} (r,t,p,c) \in \mathbf{rtpc}_{\mathbf{r,t,p,c}} \wedge STGIN/OUT\_BND_{r,t,p,c,s,bd} \wedge s \in \mathbf{rps\_prcts}_{\mathbf{r,p,s}} \wedge \\ \left(\mathbf{prc\_vint}_{\mathbf{r,p}} \vee \left(NOT\ \mathbf{prc\_ts}_{\mathbf{r,p,s}}\right)\right) \end{array} \right\}$$

All timeslices **s** *at or above the timeslice level of the process (***prc_tsl***).*

$$\sum_{ts \in \left(\mathbf{prc\_ts}_{\mathbf{r,p,ts}} \wedge \mathbf{ts\_map}_{\mathbf{r,s,ts}}\right)} \; \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} VAR\_SIN/SOUT_{r,v,t,p,c,ts}$$

$$(\leq / \geq / =) \qquad STGIN/OUT\_BND_{r,t,p,c,s,bd}$$

where the equation sign is indicated by equation index **l** based on the bound type **bd**.

6.3.38 Equations: EQ_STSBAL

Indices: region (r), vintage (v), period (t), process (p), timeslice (s)

Type: =

Related variables: VAR_ACT, VAR_FLO, VAR_SIN, VAR_SOUT

Related equations: **EQ_STGTSS**

Purpose: Establishes the balance between different levels of a general timeslice storage.

Equation:

$$EQ\_STSBAL_{r,v,t,p,s} \quad \forall (r,v,t,p,s) \in \left(\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \cap \mathbf{prc\_ts}_{\mathbf{r,p,s}} \cap \neg \mathbf{rps\_stg}_{\mathbf{r,p,s}}\right)$$

$$\sum_{\mathbf{rs\_below}_{\mathbf{r,'ANNUAL',s}}} VAR\_ACT_{r,v,t,p,s} =$$

$$\sum_{\mathbf{rs\_below}_{\mathbf{r,'ANNUAL',s-1}}} \left(\begin{array}{l} VAR\_ACT_{r,v,t,p,s-1} + VAR\_SOUT_{r,v,t,p,'ACT',s-1} - \\ \displaystyle\sum_{ts \in \{sl | \mathbf{prc\_ts}_{r,p,sl} \wedge \mathbf{rs\_below1}_{\mathbf{r,sl,s}}\}} VAR\_SOUT_{r,v,t,p,'ACT',ts} \times RS\_FR_{r,s-1,ts} - \\ \left(\dfrac{VAR\_ACT_{r,v,t,p,s} + VAR\_ACT_{r,v,t,p,s-1}}{} \right) \times STG\_LOSS_{r,v,p,s} \times \dfrac{G\_YRFR_{r,s}}{RS\_STGPRD_{r,s}} \end{array} \right) +$$

$$+ \sum_{s \in \{sl | \mathbf{annual}(sl)\}} \left[\begin{array}{l} \displaystyle\sum_{c \in \left\{\substack{\mathbf{stgips} \cap \\ \mathbf{top}_{\mathbf{'IN'}}}\right\}} \frac{VAR\_SIN_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} - \sum_{c \in \left\{\substack{\mathbf{stgips} \cap \\ \mathbf{top}_{\mathbf{'OUT'}}}\right\}} \frac{VAR\_SOUT_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} - \\ VAR\_SOUT_{r,v,t,p,'ACT',s} \end{array} \right]$$

6.3.39 Equations: EQ(*l*)_UCRTP

Indices: name (uc_n), region (r), period (t), process (p), type (uc_grptype), bound (bd)

Type: Any type, as determined by the bound index **bd** of **uc_dynbnd**:

- *l* = 'N' for **bd** = 'LO' (lower bound) yields $\geq$.
- *l* = 'N' for **bd** = 'UP' (upper bound) yields $\leq$.
- *l* = 'E' for **bd** = 'FX' (fixed bound) yields $=$.

Related variables: VAR_ACT, VAR_CAP, VAR_NCAP

Purpose: Dynamic bound on the growth/decay in the capacity (CAP), new capacity (NCAP) or activity level (ACT) of a particular process (**p**) between period (**t**) and previous period (**t–1**).

Remarks:

- The input set **uc_dynbnd** must be used for flagging the pairs (uc_n,bd) to be reserved for dynamic bound constraints.
- The input parameters UC_CAP, UC_NCAP, and UC_ACT should be used for defining the growth/decay coefficients (side='LHS') and RHS constants (side='RHS').
- The growth/decay coefficients (side='LHS') are given as annual multipliers (e.g. 1.1 for a 10% annual growth). The RHS constants (side='RHS') represent annual absolute values of additional growth/decay.
- The LHS is by default interpolated using option 5. If no LHS is specified, the RHS is by default interpolated with the option 10, like other bounds. However, if the LHS is also specified, the RHS is by default interpolated by the same option as the LHS.
- Whenever any ***RHS values*** are specified, the constraints will be generated for those periods for which the RHS is defined after the interpolation/extrapolation. If no RHS is specified, the constraints are generated for the periods that have the LHS defined, but excluding the first period of technology availability.
- In the case of dynamic bounds on the activity (ACT), the UC_ACT values must be specified at the ANNUAL level, and constraint bounds the change in the total activity in a specific period (**t**), summing over the process vintages and timeslices.

Equations:

Case A. For CAP:

$$EQ(l)\_UCRTP_{uc\_n,r,t,p,'CAP',bd} \ni \left(\mathbf{rtp}_{r,t,p} \wedge \mathbf{uc\_dynbnd}_{uc\_n,bd} \wedge \left(\sum_{side} \left(UC\_CAP_{uc\_n,side,r,t,p} \right) > 0 \right) \right)$$

$$VAR\_CAP_{r,t,p}$$

$$\{\leq;=;\geq\}$$

$$VAR\_CAP_{r,t-1,p} \times \left(UC\_CAP_{uc\_n,'LHS',r,t,p} \right)^{(M(t)-M(t-1))} + UC\_CAP_{uc\_n,'RHS',r,t,p} \times (M(t)-M(t-1))$$

Case B. For NCAP:

$$EQ(l)\_UCRTP_{uc\_n,r,t,p,'NCAP',bd} \ni \left(\mathbf{rtp}_{r,t,p} \wedge \mathbf{uc\_dynbnd}_{uc\_n,bd} \wedge \left(\sum_{side} \left(UC\_NCAP_{uc\_n,side,r,t,p} \right) > 0 \right) \right)$$

$$VAR\_NCAP_{r,t,p}$$

$$\{\leq;=;\geq\}$$

$$VAR\_NCAP_{r,t-1,p} \times \left(UC\_NCAP_{uc\_n,'LHS',r,t,p} \right)^{(M(t)-M(t-1))} + UC\_NCAP_{uc\_n,'RHS',r,t,p} \times (M(t)-M(t-1))$$

Case C. For ACT:

$$EQ(l)\_UCRTP_{uc\_n,r,t,p,'ACT',bd} \ni \left(\mathbf{rtp}_{r,t,p} \wedge \mathbf{uc\_dynbnd}_{uc\_n,bd} \wedge \left(\sum_{side} \left(UC\_ACT_{uc\_n,side,r,t,p,'ANNUAL'} \right) > 0 \right) \right)$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{r,v,t,p}} \sum_{s \in \mathbf{prc\_ts}} VAR\_ACT_{r,t,p,s} \quad \{\leq;=;\geq\}$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{r,v,t,p}} \sum_{s \in \mathbf{prc\_ts}} VAR\_ACT_{r,t-1,p,s} \times \left(UC\_ACT_{uc\_n,'LHS',r,t,p,'ANNYAL'} \right)^{(M(t)-M(t-1))}$$

$$+UC\_ACT_{uc\_n,'RHS',r,t,p,'ANNUAL'} \times (M(t)-M(t-1))$$

6.4 User Constraints

This section on TIMES User Constraints explains the framework that may be employed by modellers to formulate additional linear constraints, which are not part of the generic constraint set of TIMES, without having to bother with any GAMS programming.

6.4.1 Overview

Indexes: region (r), time period (t), time slice (s), user constraint (uc_n)

Type: Any type, as determined by the bound index **bd** of $UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd}$:

- l = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.
- l = 'E' for **bd** = 'FX' (fixed bound) yields $=$.
- l = 'L' for **bd** = 'UP' (upper bound) yields $\leq$.

Related variables: VAR_ACT, VAR_CAP, VAR_FLO, VAR_IRE, VAR_NCAP, VAR_COMPRD, VAR_COMNET, VAR_CUMCOM, VAR_CUMFLO, VAR_UPS

Related equations: EQ(l)_COMBAL, EQ(l)_CPT

Purpose: The user constraints in TIMES provide a modeler with a flexible framework to add case-study specific constraints to the standard equation set embedded in TIMES. With the help of the user constraints virtually any possible linear relationship between variables in TIMES can be formulated. Examples of user constraints are quotas for renewables in electricity generation or primary energy consumption, GHG reduction targets, absolute bounds on the minimum amount of electricity generated by various biomass technologies, etc.

Four types of user constraints can be distinguished in TIMES:

- Pure LHS (left hand side) user constraints,
- Timeslice-dynamic user constraints,
- Dynamic user constraints of type (t, t+1), and
- Dynamic user constraints of type (t–1, t).

In addition, the dynamic bound constraints (see EQ_UCRTP) also employ user constraint names and UC_* attributes, but these constraints are based on prescribed expressions and are thus not considered genuine user constraints.

In the following four subsections, the different types of user constraints are briefly presented. Their mathematical formulations are then presented in a new section.

LHS user constraints

The so-called LHS user constraints have the following main structure:

$$EQ(l)\_UC(R)(T)(S)_{(r),uc\_n,(t),(s)} \quad \forall \begin{Bmatrix} UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}}\right) \end{Bmatrix}$$

$$\left(\sum_{r \in \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}}\right)\left(\sum_{t \in \mathbf{uc\_t\_sum}_{\mathbf{r,uc\_n,t}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}}}\right) LHS_{r,t,s} \{= / \geq / \leq\}\, UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd}$$

To identify the user constraint, the modeller has to give it a unique name **uc_n**. The LHS expression $LHS_{r,t,s}$ consists of the sum of various TIMES variables (VAR_ACT, VAR_FLO, VAR_COMPRD, VAR_COMNET, VAR_NCAP, VAR_CAP), multiplied by corresponding coefficients (UC_ACT, UC_FLO, UC_COMPRD, UC_COMCON, UC_NCAP, UC_CAP). The coefficients are input data given by the modeller and serve thus also as an indicator of which variables are being components of the user constraint.

With respect to region **r**, time period **t** and timeslice **s**, the user constraint is either specified for specific regions, periods or timeslices or the expression within the user constraint is summed over subsets of regions, periods and timeslices. In the first case, the regions, periods or timeslices for which the user constraint should be generated are given by the sets **uc_r_each**, **uc_t_each** or **uc_ts_each**, while in the latter case, summation sets are specified by the sets **uc_r_sum**, **uc_t_sum** and **uc_ts_sum**. The corresponding sets **uc_x_each/sum** are exclusive, so that for example, if **uc_t_each** has been specified, the set **uc_t_sum** cannot be specified and vice versa. By choosing **uc_x_each/sum** also the name and the index domain of the user constraint are specified, e.g. if **uc_r_each**, **uc_t_each** and **uc_ts_sum** are given, the user constraint has the name and index domain $EQ(l)\_UCRT_{r,uc\_n,t}$. It is generated for each region and period specified by **uc_r_each** and **uc_t_each**, respectively, and is summing within the user constraint over the timeslices given in **uc_ts_each**. The name of the RHS constraint depends in the same way on the choice of **uc_x_each/sum**. In the previous example, the RHS constant has the name and index domain $UC\_RHSRT_{r,uc\_n,t,bd}$. The knowledge of these naming rules is **important**, since the modeller has to give the correct RHS parameter names depending on the choice of **uc_x_each/sum** when defining a user constraint.

Since for each of the three dimensions (region, period, timeslice), two options (EACH or SUM) exist, this would result in 8 possible combinations of user constraint equations (Figure 5.6). However, the combinations EQ(l)_UCS and EQ(l)_UCRS, which would lead to a constraint being generated for specific timeslices while summing over time periods at the same time, have been considered unrealistic, so that 6 variants remain. It should be noted that the sets **uc_r_each/sum**, **uc_t_each/sum** and **uc_ts_each/sum** can contain an arbitrary combination of elements, e.g. the periods specified in **uc_t_each/sum** do not have to be consecutive.

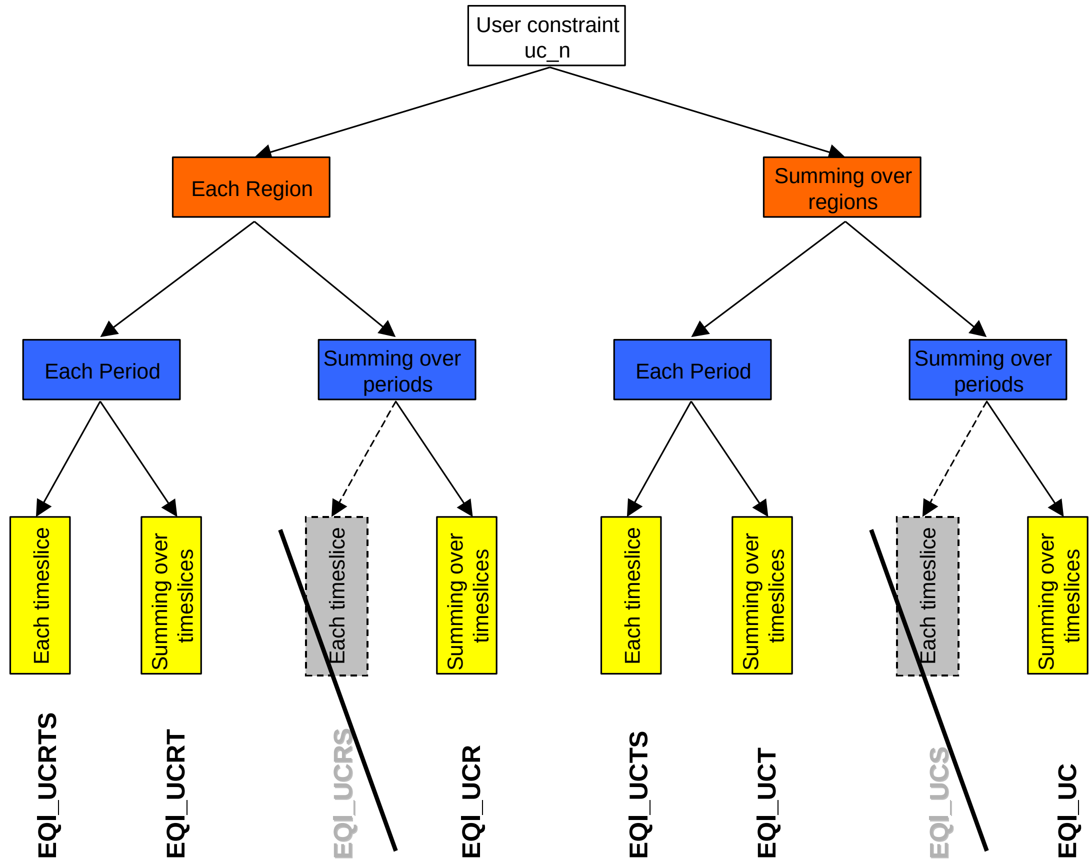


Figure 14: The allowed combinations of region, period and timeslice for user constraints.

The RHS (right hand side) of this category of user constraint consists of a constant $UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd}$ which is provided by the modeller. The RHS constant also defines the equation type of the user constraint. If the RHS constant has the index FX, the user constraint is generated as strict equality (=). If the RHS index is LO (respectively UP), the constraint has $\geq$ (respectively $\leq$) inequality sign. It should be noted that a RHS user constraint is only generated when a RHS constant is specified (this feature may be used to easily turn-on/off user constraints between different scenarios).

In addition to the coefficients UC_ACT, UC_FLO, etc. also some model input attributes may be used as coefficient for the variables in a user constraint. The model attribute being used as coefficient in a user constraint is specified by the set $UC\_ATTR_{r,uc\_n,'LHS',VAR,ATTR}$ with the indicator VAR for the variable (ACT, FLO, IRE, NCAP, CAP, COMNET, COMPRD) and the index ATTR representing the attribute being used (COST, SUB, TAX, DELIV, INVCOST, INVSUB, INVTAX, CAPACT, CAPFLO, NEWFLO, ONLINE, EFF, NET, CUMSUM, PERIOD, GROWTH, see Section 6.4.6 for more information).

Instead of defining different equality types of user constraints depending on the bound type of $UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd}$ an alternative formulation can be used in TIMES.

In this formulation a variable $VAR\_UC(R)(T)(S)_{(r),uc\_n,(t),(s)}$ is created that is set equal to the LHS expression. The RHS bounds are then applied to these variables.

$$EQE\_UC(R)(T)(S)_{(r),uc\_n,(t),(s)} \quad \forall \left\{ \begin{array}{l} UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each_{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_each_{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each_{r,uc\_n,s}}\right) \end{array} \right\}$$

$$\left(\sum_{r \in uc\_r\_sum_{r,uc\_n}}\right)\left(\sum_{t \in uc\_t\_sum_{r,uc\_n,t}}\right)\left(\sum_{s \in uc\_ts\_sum_{r,uc\_n,s}}\right) LHS_{r,t,s} = VAR\_UC(R)(T)(S)_{(r),uc\_n,(t),(s)}$$

$$VAR\_UC(R)(T)(S).LO_{(r),uc\_n,(t),(s)} = UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),'LO'}$$
$$VAR\_UC(R)(T)(S).UP_{(r),uc\_n,(t),(s)} = UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),'UP'}$$
$$VAR\_UC(R)(T)(S).FX_{(r),uc\_n,(t),(s)} = UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),'FX'}$$

The alternative formulation is created when the dollar control parameter VAR_UC (see Part III for the use of dollar control parameters) is set to YES by the modeller, while in the default case the first formulation is used.

Timeslice-dynamic user constraints

Timeslice-dynamic user constraints establish a relationship between two successive timeslices within a timeslice cycle. The LHS expression $LHS_{r,t,s}$ is generated for timeslice **s**, whereas the RHS expression $RHS_{r,t,s-1}$ is generated for the preceding timeslice **s**–RS_STG(r,s) under the same parent timeslice. Timeslice-dynamic user constraints of type can thus be written as follows:

$$EQ(l)\_UCRS_{r,uc\_n,t,tsl,s} \ni \left\{ \begin{array}{c} UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \left(r \in \mathbf{uc\_r\_each_{r,uc\_n}}\right) \wedge \left(t \in \mathbf{uc\_t\_each_{r,uc\_n,t}}\right) \\ \wedge (s \in \{ts \mid \mathbf{ts\_grp}_{r,tsl,s} \wedge \bigcup_{side} \mathbf{uc\_tsl}_{r,uc\_n,side,tsl}\}) \end{array} \right\}$$

$$LHS_{r,t,s} \qquad \{= / \geq / \leq\} \sum_{\mathbf{uc\_tsl}(r,ucn,'RHS',tsl)} RHS_{r,t,s-RS\_STG(r,s)} + UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd}$$

Timeslice-dynamic user constraints are always specific to a single region and period. To build a timeslice-dynamic user constraint, the modeller must identify the desired timeslice level of the constraint, by using the set $\mathbf{uc\_tsl}_{r,uc\_n,side,tsl}$, and the RHS constants must be defined by using the UC_RHSRTS parameter. The constraint will be genuinely dynamic only if **uc_tsl** is specified on the RHS. This is the only type of user constraint for which the RHS constant parameter is levelized, according the timeslice level identified by **uc_tsl**. That can make the RHS specification much easier.

Dynamic user constraints

Dynamic user constraints establish a relationship between two *consecutive* periods. The LHS expression $LHS_{r,t,s}$ is generated for period **t**, whereas the for the RHS expression either the term $RHS_{r,t+1,s}$ corresponding to the period **t+1**, or the term $RHS_{r,t-1,s}$ corresponding to the period **t–1** is generated, according to the dynamic type.

Dynamic user constraints of type (**t,t+1**) can thus be written as follows:

$$EQ(l)\_UC(R)SU(S)_{(r),uc\_n,t,(s)} \ni \left\{ \begin{array}{l} UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each_{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_succ_{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each_{r,uc\_n,s}}\right) \end{array} \right\}$$

$$\left(\sum_{r \in \mathbf{uc\_r\_sum_{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum_{r,uc\_n,s}}}\right) LHS_{r,t,s} \{= / \geq / \leq\} \left(\sum_{r \in \mathbf{uc\_r\_sum_{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum_{r,uc\_n,s}}}\right) RHS_{r,t+1,s} + UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd}$$

Similarly, dynamic user constraints of type (**t–1,t**) can be written as follows:

$$EQ(l)\_UC(R)SU(S)_{(r),uc\_n,t,(s)} \ni \left\{ \begin{array}{l} UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each_{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_succ_{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each_{r,uc\_n,s}}\right) \end{array} \right\}$$

$$\left(\sum_{r \in \mathbf{uc\_r\_sum_{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum_{r,uc\_n,s}}}\right) LHS_{r,t,s} \{= / \geq / \leq\} \left(\sum_{r \in \mathbf{uc\_r\_sum_{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum_{r,uc\_n,s}}}\right) RHS_{r,t-1,s} + UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd}$$

To build a dynamic user constraint of type (**t,t+1**), between the periods **t** and **t+1**, the modeller identifies the desired set of time periods that will be used as first periods in the pairs (**t**, **t+1**). This set is named **uc_t_succ** (note that the sets **uc_t_sum** and **uc_t_each** are not used in the context of dynamic user constraints, and are reserved for the pure LHS user constraints described in the previous section). In addition, the RHS constant parameter must be defined for all of these time periods.

To build a dynamic user constraint of type (**t–1,t**), between the periods **t–1** and **t**, the modeller should indicate the desired type by defining for the constraint any UC_ATTR attribute using the 'RHS' side. In addition, the desired set of time periods that will be used as the second period in the pairs(**t–1,t**) should be identified by defining the RHS constant parameter for those periods **t**.

The choice between the dynamic types (**t,t+1**) or (**t–1,t**) is usually only a matter of convenience. However, while using the (**t,t+1**) type requires explicit specification of **uc_t_succ**, for using the (**t–1,t**) type, any UC_ATTR on the RHS is sufficient to trigger that dynamic type and will cause auto-generation of **uc_t_succ** for all milestone years.

For both types of dynamic constraints, only four combinations with respect to the region and timeslice domain are possible:

- EQ(l)_UCSU: dynamic user constraint summing **r** over **uc_r_sum** and **s** over **uc_ts_sum**,
- EQ(l)_UCRSU: dynamic user constraint being generated for each region **uc_r_each** and summing **s** over **uc_ts_sum**,
- EQ(l)_UCRSUS: dynamic user constraint being generated for each region **uc_r_each** and timeslice **uc_ts_each** and
- EQ(l)_UCSUS: dynamic user constraint summing **r** over **uc_r_sum** and being generated for each timeslice **s** in set **uc_ts_each**.

The input parameters for defining the coefficients, UC_ACT, UC_FLO, UC_IRE, UC_COMCON, UC_COMNET, UC_COMPRD, UC_NCAP and UC_CAP all have an index **side**, which can be either LHS or RHS, to identify on which side of the user constraint the corresponding variables should appear. The LHS index corresponds always to the period **t**, while the RHS index is related either to the **t+1** or the **t–1** term.

As for LHS user constraints, setting the dollar control parameter VAR_UC to YES yields a strict equality type of dynamic user constraint (EQE_UCSU, EQE_UCRSU, EQE_UCRSUS, EQE_UCSUS) with the RHS constant replaced by a user constraint variable (VAR_UCT, VAR_UCRT, VAR_UCRTS, VAR_UCTS). The bound given by the RHS constant is then applied to the user constraint variable.

Growth constraints

Growth (or decay) constraints are a special type of dynamic constraints. A growth constraint may for example express that the capacity increase between two periods is limited by an annual growth rate. So, growth constraints relate variables in one period to the ones in the previous or following period as in dynamic constraints described in the previous section. In growth constraints, however, in addition some of the variable coefficients UC_ACT, UC_FLO, UC_IRE, UC_COMNET, UC_COMPRD, UC_NCAP, UC_CAP can represent annual growth (or decay) rates[44] by specifying the set $UC\_ATTR_{r,uc\_n,'LHS',VAR,ATTR}$ with the index ATTR being set to GROWTH. This will cause the coefficient of the corresponding variable being interpreted as an annual growth rate. If for example the input information $UC\_ATTR_{'REG1','G\_1','LHS','CAP','GROWTH'}$ is given for the user constraint G_1, the coefficient $UC\_CAP_{'G\_1','LHS','REG1',t,p}$ of the capacity variable of technology **p** will be interpreted as annual growth rate and the final coefficient of the variable VAR_CAP in the user constraint will be calculated in the following way:

$$\left(UC\_CAP_{'G\_1','LHS','REG1',t,p}\right)^{M(t+1)-M(t)}.$$

With the help of the input set UC_ATTR, growth coefficients can be defined for the variables in LHS expression (as in the example) or for the variables in RHS expression. If a

[44] If the coefficient UC_ACT, UC_FLO, etc. is greater than one, it represents an annual growth rate, while a coefficient smaller than one describes an annual decay rate.

growth rate is defined for variables on the LHS, the exponent is M(t+1)–M(t), whereas for RHS variables the exponent is equal to M(t)–M(t+1).

If at least one growth coefficient is defined for a LHS variable, the dynamic constraint will be assumed to be of type (**t**,**t+1**) described above. In this case, the growth constraints are generated for the period pairs **t** and **t+1** for all periods **t** of the model horizon with the exception of the last period.

If, however, all growth coefficients are specified for the RHS variables, the dynamic constraint will be assumed to be of type (**t–1,t**), and the growth constraints are now generated for the period pairs **t**–1 and **t** for all periods of the model horizon. In this alternative RHS formulation, it is possible to introduce boundary conditions that are usually needed for the first period.

Example of defining a simple growth constraint:

The annual capacity increase of technology E01 between two periods should not exceed 2% for model covering the three ten-year periods 1990, 2000 and 2010. So one wants to create user constraints expressing:

$$1.02^{10} \times VAR\_CAP_{'REG1','1990','E01'} + 1 \geq VAR\_CAP_{'REG1','2000','E01'}$$

$$1.02^{10} \times VAR\_CAP_{'REG1','2000','E01'} + 1 \geq VAR\_CAP_{'REG1','2010','E01'}$$

The summand 1 on the LHS expresses an initial capacity value, so that capacity growth can start from this starting point, e.g. if $VAR\_CAP_{'REG1','1990','E01'}$ is zero, the model can invest at most 1 capacity unit in the year 2000: $1 \geq VAR\_CAP_{'REG1','2000','E01'}$.

Since growth constraints should be generated for the first two periods, but not the last one, the growth constraint should be of type (**t**,**t**+1). The specification of the growth constraint called ‘G_1’ in GAMS looks like:

```
SET UC_N /
G_1
/

* Specify growth of capacity on the LHS
SET UC_ATTR /
REG1.G_1.LHS.CAP.GROWTH
/

* Specify growth coefficient for E01 on LHS (period 1) and coefficient
* for capacity on RHS (period t+1)
PARAMETER UC_CAP /
* on the LHS
G1.LHS.REG1.2000.E01       1.02
* on the RHS
G1.RHS.REG1.2000.E01       1
/

* Specify RHS constant for the years t to have the constraint
PARAMETER UC_RHSRTS /
REG1.G_1.1990.ANNUAL.LO    -1
REG1.G_1.2000.ANNUAL.LO    -1
/;
```

One should note that the period index used for the UC_CAP on the LHS is related to the period **t**, while the period index on the RHS is related to the period **t**+1. The RHS UC_RHSRTS constant is provided for the time period **t** of the LHS.

Since a growth coefficient is specified for the LHS, the user constraint is automatically identified as a dynamic growth constraint, so that the set **uc_t_succ** does not need to be provided by the user. The constraint will be generated for all periods for which the RHS parameter UC_RHSRTS is given.

In the following section, we give the full descriptions of the available user constraints in each category, along with a reminder of the corresponding variables.

Mathematical descriptions of user constraints

<u>List of user constraints and variables</u>

We first show the complete list of user constraints in the three categories.

The following types of LHS user constraints exist:

- $EQ(l)\_UC_{uc\_n}$: user constraint summing over regions **uc_r_sum**, periods **uc_t_sum** and timeslices **uc_ts_sum**,
- $EQ(l)\_UCR_{r,uc\_n}$: user constraint generated for regions **uc_r_each** and summing over periods **uc_t_sum** and timeslices **uc_ts_sum**,
- $EQ(l)\_UCT_{uc\_n,t}$: user constraint generated for periods **uc_t_each** and summing over regions **uc_r_sum** and timeslices **uc_ts_sum**,
- $EQ(l)\_UCRT_{r,uc\_n,t}$: user constraint generated for regions **uc_r_each** and periods **uc_t_each** and summing over timeslices **uc_ts_sum**,
- $EQ(l)\_UCTS_{uc\_n,t,s}$: user constraint generated for periods **uc_t_each**, timeslices **uc_ts_each** and summing over regions **uc_r_sum**,
- $EQ(l)\_UCRTS_{r,uc\_n,t,s}$: user constraint generated for regions **uc_r_each**, periods **uc_t_each** and timeslices **uc_ts_each**.

The placeholder **l** reflects the equation type of the user constraint (**l**=E, G or L) corresponding to the bound type of the RHS constant. In case the dollar control parameter VAR_UC is set to YES, the user constraints are always strict equalities (**l**=E) with the RHS constants replaced by the following user constraint variables:

- $VAR\_UC_{uc\_n}$: user constraint variable for EQE_UC,
- VAR_UCR_{r,uc_n} : user constraint variable for EQE_UCR,
- $VAR\_UCT_{uc\_n,t}$: user constraint variable for EQE_UCT,
- $VAR\_UCRT_{r,uc\_n,t}$: user constraint variable for EQE_UCRT,
- $VAR\_UCTS_{uc\_n,t,s}$: user constraint variable for EQE_UCTS,
- $VAR\_UCRTS_{uc\_n,r,t,s}$: user constraint variable for EQE_UCRTS.

The following types of dynamic user constraints and growth constraints exist:

- $EQ(l)\_UCSU_{uc\_n,t}$: user constraint generated for periods **uc_t_succ**, summing over regions **uc_r_sum** and timeslices **uc_ts_sum**,

- $EQ(l)\_UCRSU_{r,uc\_n,t}$: user constraint generated for regions **uc_r_each** and periods **uc_t_succ** and summing over timeslices **uc_ts_sum**,
- $EQ(l)\_UCSUS_{uc\_n,t,s}$: user constraint generated for periods **uc_t_succ**, timeslices **uc_ts_each** and summing over regions **uc_r_sum**,
- $EQ(l)\_UCRSUS_{r,uc\_n,t,s}$: user constraint generated for regions **uc_r_each**, periods **uc_t_succ** and timeslices **uc_ts_each**.

The placeholder ***l*** reflects the equation type of the user constraint (***l***=E, G or L) corresponding to the bound type of the RHS constant. In case the dollar control parameter VAR_UC is set to YES, the user constraints are always strict equalities (***l***=E) with the RHS constants replaced by the following user constraint variables:

- $VAR\_UCT_{uc\_n,t}$: user constraint variable for EQE_UCSU,
- $VAR\_UCRT_{r,uc\_n,t}$: user constraint variable for EQE_UCRSU,
- $VAR\_UCTS_{uc\_n,t,s}$: user constraint variable for EQE_UCSUS,
- $VAR\_UCRTS_{uc\_n,r,t,s}$: user constraint variable for EQE_UCRSUS.

Sets and parameters related to user constraints

The following sets and parameters are related to the user constraint framework in TIMES.

Sets

Predefined internal sets:

- **side** : set having the two elements *LHS* and *RHS* (elements are fixed and not under user control),
- **uc_grptype** : set having the elements *ACT*, *FLO*, *IRE*, *COMCON*, *COMPRD*, *NCAP*, *CAP* used in the multi-dimensional set *UC_ATTR* (elements are fixed and not under user control),
- **uc_name** : set having the following attribute names as elements: *COST*, *SUB*, *TAX*, *DELIV*, *INVCOST*, *INVSUB*, *INVTAX*, *CAPACT*, *CAPFLO*, *NEWFLO*, *ONLINE*, *EFF*, *NET*, *CUMSUM*, *PERIOD*, *GROWTH* and *SYNC* used in the multi-dimensional set *UC_ATTR* (elements are fixed and not under user control).

User-specified sets:

- **uc_n**: unique name of the user constraint,
- $\mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}$: regions **r** for which the user constraint **uc_n** is generated,
- $\mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}$: regions **r** being summed over in the user constraint **uc_n**,
- $\mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}$: periods **t** for which the user constraint **uc_n** is generated,
- $\mathbf{uc\_t\_sum}_{\mathbf{r,uc\_n,t}}$: periods **t** being summed over in the user constraint **uc_n**,
- $\mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,ts}}$: timeslices **ts** for which the user constraint **uc_n** is generated,
- $\mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,ts}}$: timeslices **ts** being summed over in the user constraint **uc_n**,
- $\mathbf{uc\_tsl}_{\mathbf{r,uc\_n,side,tslvl}}$: timeslice level **tslvl** of user constraint **uc_n**,

- **uc_attr**$_{\mathbf{r,uc\_n,side,uc\_grptype,uc\_name}}$: indicator that the attribute **uc_name** on the RHS or LHS **side** of the user constraint **uc_n** as coefficient of the variable given by **uc_grptype**.

If neither **uc_r_each** nor **uc_r_sum** are given, the default is set to all **uc_r_each** containing all internal regions. In a similar fashion **uc_t_each** being set to all milestoneyears is the default, if neither **uc_t_each** or **uc_t_sum** are specified. The default for the timeslice dimension is **uc_ts_each** being set to all timeslices for which the RHS constants UC_RHSRS or UC_RHSRTS are being specified.

Parameters

User-specified coefficients of variables:

- $UC\_ACT_{uc\_n,side,r,t,p,s}$: coefficient of the activity variable $VAR\_ACT_{r,v,t,p,s}$ in the user constraint **uc_n** on the LHS or RHS **side**,
- $UC\_FLO_{uc\_n,side,r,t,p,c,s}$: coefficient of the flow variable $VAR\_FLO_{r,v,t,p,c,s}$ in the user constraint **uc_n** on the LHS or RHS **side**,
- $UC\_IRE_{uc\_n,side,r,t,p,c,s,ie}$: coefficient of the inter-regional exchange variable $VAR\_IRE_{r,v,t,p,c,s,ie}$ in the user constraint **uc_n** on the LHS or RHS **side**,
- $UC\_COMCON_{uc\_n,side,r,t,c,s}$: coefficient of the virtual commodity consumption variable ($VAR\_COMPRD_{r,t,c,s}$–$VAR\_COMNET_{r,t,c,s}$) in the user constraint **uc_n** on the LHS or RHS **side**,
- $UC\_COMPRD_{uc\_n,side,r,t,c,s}$: coefficient of the gross commodity production variable $VAR\_COMPRD_{r,t,c,s}$ in the user constraint **uc_n** on the LHS or RHS **side**,
- $UC\_COMNET_{uc\_n,side,r,t,c,s}$: coefficient of the net commodity production variable $VAR\_COMNET_{r,t,c,s}$ in the user constraint **uc_n** on the LHS or RHS **side**,
- $UC\_CUMACT_{uc\_n,r,p,y1,y2}$: coefficient of the cumulative process activity variable $VAR\_CUMFLO_{r,p,'ACT',y1,y2}$ in the user constraint **uc_n** (only in cumulative constraints),
- $UC\_CUMCOM_{uc\_n,r,type,c,y1,y2}$: coefficient of the cumulative commodity net or gross production variable $VAR\_CUMCOM_{r,c,type,y1,y2}$ in the user constraint **uc_n**, where type =PRD/NET (only in cumulative constraints),
- $UC\_CUMFLO_{uc\_n,r,p,c,y1,y2}$: coefficient of the cumulative process flow variable $VAR\_CUMFLO_{r,p,c,y1,y2}$ in the user constraint **uc_n** (only in cumulative constraints),
- $UC\_NCAP_{uc\_n,side,r,t,p}$: coefficient of the investment variable $VAR\_NCAP_{r,t,p}$ in the user constraint **uc_n** on the LHS or RHS **side**,
- $UC\_CAP_{uc\_n,side,r,t,p}$: coefficient of the capacity variable $VAR\_CAP_{r,t,p}$ in the user constraint **uc_n** on the LHS or RHS **side**.

User-specified RHS constants:

- $UC\_RHS_{uc\_n,bd}$: RHS constant with bound type **bd** of the user constraint $EQl\_UC_{uc\_n}$ of type **l**,
- $UC\_RHSR_{r,uc\_n,bd}$: RHS constant with bound type **bd** of the user constraint EQl_UCR_{r,uc_n} of type **l**,
- $UC\_RHST_{uc\_n,t,bd}$: RHS constant with bound type **bd** of the user constraint $EQl\_UCT_{uc\_n,t}$ of type **l**,
- $UC\_RHSRT_{r,uc\_n,t,bd}$: RHS constant with bound type **bd** of the user constraint $EQl\_UCRT_{r,uc\_n,t}$ of type **l**,
- $UC\_RHSTS_{uc\_n,t,s,bd}$: RHS constant with bound type **bd** of the user constraint $EQl\_UCTS_{uc\_n,t,s}$ of type **l**,
- $UC\_RHSRTS_{r,uc\_n,t,s,bd}$: RHS constant with bound type **bd** of the user constraint $EQl\_UCRTS_{r,uc\_n,t,s}$ of type **l**.
- $UC\_TIME_{uc\_n,r,t}$: Defines an additional term in the RHS constant, which is either the time (in years) covered by the user constraint multiplied by *UC_TIME* (for static and cumulative constraints), or the time between the milestone years of the successive periods in the constraint (for dynamic user constraints).

6.4.2 LHS user constraints

Mathematical formulation of LHS user constraints

In the mathematical description of the different variants of LHS user constraints the following placeholders are used for clarity reasons: $ACT_{r,t,p,s,'LHS'}$, $FLO_{r,t,p,s,'LHS'}$, $IRE_{r,t,p,s,'LHS'}$, $COMPRD_{r,t,s,'LHS'}$, $COMNET_{r,t,s,'LHS'}$, $NCAP_{r,t,p,s,'LHS'}$, $CAP_{r,t,p,s,'LHS'}$, $CUMCOM_r$ and $CUMFLO_r$. For example the placeholder $ACT_{r,t,p,s,'LHS'}$ includes the part of the user constraint related to the activity variable.

$$ACT_{r,t,s,'LHS'} = \sum_{(v,p)\in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \sum_{ts\in \mathbf{prc\_ts}_{\mathbf{r,p,ts}}} \left(\begin{array}{l} VAR\_ACT_{r,v,t,p,ts} \times UC\_ACT_{uc\_n,'LHS',r,t,p,ts} \times \left(RS\_FR_{r,s,ts}\right) \\ \times \\ \left(\sum_{cur\in \mathbf{rdcur}_{\mathbf{r,cur}}} OBJ\_ACOST_{r,t,p,cur} \right) \quad if\ UC\_ATTR_{r,uc\_n,'LHS','ACT','COST'}\ is\ given \end{array} \right)$$

$$FLO_{r,t,s,'LHS'} = \sum_{(p,c,ts)\in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left\{ \begin{array}{l} VAR\_FLO_{r,v,t,p,c,ts} \times UC\_FLO_{uc\_n,'LHS',r,t,p,c,ts} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\ \times \\ \left[\sum_{cur\in \mathbf{rdcur}_{\mathbf{r,cur}}} \left(\begin{array}{l} OBJ\_FCOST_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','COST'}\ is\ given \\ + \\ OBJ\_FDELV_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','DELIV'}\ is\ given \\ - \\ OBJ\_FSUB_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','SUB'}\ is\ given \\ + \\ OBJ\_FTAX_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','TAX'}\ is\ given \end{array} \right) \right] \end{array} \right\}$$

$$
\begin{aligned}
&IRE_{r,t,s,'LHS'} \\
&= \sum_{(p,c,ts)\in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \sum_{ie\in \mathbf{rpc\_ire}_{\mathbf{r,p,c,ie}}} \\
&\left[\begin{array}{l}
VAR\_IRE_{r,v,t,p,c,ts,ie} \times UC\_IRE_{uc\_n,'LHS',r,t,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\
\times \\
\sum\limits_{cur\in \mathbf{rdcur}_{r,cur}} \left(\begin{array}{l}
OBJ\_FCOST_{r,t,p,c,s,cur} \ \ if \ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','COST'} \\
+ \\
OBJ\_FDELV_{r,t,p,c,s,cur} \ \ if \ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','DELIV'} \\
- \\
OBJ\_FSUB_{r,t,p,c,s,cur} \ \ if \ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','SUB'} \\
+ \\
OBJ\_FTAX_{r,t,p,c,s,cur} \ \ if \ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','TAX'}
\end{array} \right)
\end{array}\right]
\end{aligned}
$$

$$
\begin{aligned}
&COMPRD_{r,t,s,'LHS'} \\
&= \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t,c,ts}}} \left(\begin{array}{l}
VAR\_COMPRD_{r,t,c,ts} \times UC\_COMPRD_{uc\_n,'LHS',r,t,c,s} \\
\times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\
\sum\limits_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t,c,ts,uc\_\mathrm{cost},cur} \quad if \ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMPRD',uc\_cost} \right)
\end{array} \right)
\end{aligned}
$$

$$
\begin{aligned}
&COMNET_{r,t,s,'LHS'} \\
&= \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t,c,ts}}} \left(\begin{array}{l}
VAR\_COMNET_{r,t,c,ts} \times UC\_COMNET_{uc\_n,'LHS',r,t,c,s} \\
\times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\
\sum\limits_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_\mathrm{cost}}} \left(OBJ\_COMNT_{r,t,c,ts,uc\_\mathrm{cost},cur} \quad if \ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMNET',uc\_\mathrm{cost}} \right)
\end{array} \right)
\end{aligned}
$$

$$NCAP_{r,t,'LHS'}$$
$$=$$
$$\sum_{p}\left[\begin{array}{l} VAR\_NCAP_{r,t,p} \times UC\_NCAP_{uc\_n,'LHS',r,t,p} \times \\ \left(\begin{array}{l} \sum_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','COST'}\ is\ given \\ - \\ \sum_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','SUB'}\ is\ given \\ + \\ \sum_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','TAX'}\ is\ given \end{array}\right) \end{array}\right]$$

$$CAP_{r,t,p,'LHS'}$$
$$=$$
$$\sum_{p}\left[\begin{array}{l} VAR\_CAP_{r,t,p} \times UC\_CAP_{uc\_n,'LHS',r,t,p} \\ \times \\ PRC\_CAPACT_{r,p} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','CAP','CAPACT'}\ is\ given \end{array}\right]$$

$$CUMCOM_{r}$$
$$=$$
$$\sum_{\mathbf{rc\_cumcom_{r,com\_var,c}}} VAR\_CUMCOM_{r,c,com\_\mathrm{var},y1,y2} \times UC\_CUMCOM_{uc\_n,r,com\_\mathrm{var},c,y1,y2}$$

$$CUMFLO_{r}$$
$$=$$
$$\sum_{\mathbf{rpc\_cumflo_{r,p,c}}} \left(VAR\_CUMFLO_{r,p,c,y1,y2} \times UC\_CUMFLO_{uc\_n,r,p,c,y1,y2}\right) +$$
$$\sum_{\mathbf{rpc\_cumflo_{r,p,'ACT'}}} \left(VAR\_CUMFLO_{r,p,'ACT',y1,y2} \times UC\_CUMACT_{uc\_n,r,p,y1,y2}\right)$$

6.4.2.1 Equation: EQ(*l*)_UC / EQE_UC

Indices: user constraint (uc_n)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET, VAR_CUMCOM, VAR_CUMFLO

Purpose: The user constraint **EQ(*l*)_UC** is a user constraint, which is summing over specified regions (**uc_r_sum**), periods (**uc_t_sum**) and timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UC_{uc\_n} \ni UC\_RHS_{uc\_n,bd} \wedge \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}} \wedge \mathbf{uc\_t\_sum}_{\mathbf{r,uc\_n,t}}$$

$$\sum_{r\in\mathbf{uc\_r\_sum}} \sum_{t\in\mathbf{uc\_t\_sum}} \sum_{s\in\mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r\in\mathbf{uc\_r\_sum}} \left(\sum_{t\in\mathbf{uc\_t\_sum}} \left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'}\right) + \left(CUMCOM_r + CUMFLO_r\right) \right)$$

when control parameter VAR_UC is set to NO by the user or is missing:

$$\{\leq;=;\geq\}$$
$$UC\_RHS_{uc\_n,l} + \sum_{t\in\mathbf{uc\_t\_sum}} \sum_{r\in\mathbf{uc\_r\_sum}} UC\_TIME_{uc\_n,r,t} \times D_t$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the LHS is set equal to the variable VAR_UC. The bounds UC_RHS are then applied to the variable VAR_UC.

$$=$$
$$VAR\_UC_{uc\_n} + \sum_{t\in\mathbf{uc\_t\_sum}} \sum_{r\in\mathbf{uc\_r\_sum}} UC\_TIME_{uc\_n,r,t} \times D_t$$

with

$$VAR\_UC.LO_{uc\_n} = UC\_RHS_{uc\_n,'LO'}$$
$$VAR\_UC.UP_{uc\_n} = UC\_RHS_{uc\_n,'UP'}$$
$$VAR\_UC.FX_{uc\_n} = UC\_RHS_{uc\_n,'FX'}$$

6.4.2.2 Equation: EQ(*l*)_UCR / EQE_UCR

Indices: user constraint (uc_n), region (r)

Related variables: **VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET, VAR_CUMCOM, VAR_CUMFLO**

Purpose: The user constraint **EQ(*l*)_UCR** is a user constraint, which is created for each region of **uc_r_each** and is summing over periods (**uc_t_sum**) and timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UCR_{r,uc\_n} \ni UC\_RHSR_{r,uc\_n,bd} \wedge \mathbf{uc\_ts\_sum_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each_{r,uc\_n}} \wedge \mathbf{uc\_t\_sum_{r,uc\_n,t}}$$

$$\sum_{t \in \mathbf{uc\_t\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{t \in \mathbf{uc\_t\_sum}} \left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'} \right) + \quad \left(CUMCOM_r + CUMFLO_r \right)$$

when control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$
$$UC\_RHSR_{r,uc\_n,l} + \sum_{t \in \mathbf{uc\_t\_sum}} UC\_TIME_{uc\_n,r,t} \times D_t$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the LHS is set equal to the variable VAR_UCR. The bounds UC_RHSR are then applied to the variable VAR_UCR.

$$=$$
$$VAR\_UCR_{r,uc\_n} + \sum_{t \in \mathbf{uc\_t\_sum}} UC\_TIME_{uc\_n,r,t} \times D_t$$

with

$$VAR\_UCR.LO_{r,uc\_n} = UC\_RHSR_{r,uc\_n,'LO'}$$
$$VAR\_UCR.UP_{r,uc\_n} = UC\_RHSR_{r,uc\_n,'UP'}$$
$$VAR\_UCR.FX_{r,uc\_n} = UC\_RHSR_{r,uc\_n,'FX'}$$

6.4.2.3 Equation: EQ(*l*)_UCT / EQE_UCT

Indices: user constraint (uc_n), period (t)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET, VAR_CUMCOM, VAR_CUMFLO

Purpose: The user constraint **EQ(*l*)_UCT** is a user constraint, which is created for each period of **uc_t_each** and is summing over regions (**uc_r_sum**) and timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UCT_{uc\_n,t} \ni UC\_RHST_{uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}$$

$$\sum_{r\in\mathbf{uc\_r\_sum}}\sum_{s\in\mathbf{uc\_ts\_sum}}\begin{pmatrix} ACT_{r,t,s,'LHS'}+FLO_{r,t,s,'LHS'}+IRE_{r,t,s,'LHS'} \\ +COMNET_{r,t,s,'LHS'}+COMPRD_{r,t,s,'LHS'}\end{pmatrix}$$
$$+$$
$$\sum_{r\in\mathbf{uc\_r\_sum}}\left(NCAP_{r,t,'LHS'}+CAP_{r,t,'LHS'}\right)$$

when control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$
$$UC\_RHST_{uc\_n,t,l}+\sum_{r\in\mathbf{uc\_r\_sum}}UC\_TIME_{uc\_n,r,t}\times D_t$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the LHS is set equal to the variable VAR_UCT. The bounds UC_RHST are then applied to the variable VAR_UCT.

$$=$$
$$VAR\_UCT_{uc\_n,t}+\sum_{r\in\mathbf{uc\_r\_sum}}UC\_TIME_{uc\_n,r,t}\times D_t$$

with

$$VAR\_UCT.LO_{uc\_n,t}=UC\_RHST_{uc\_n,t,'LO'}$$
$$VAR\_UCT.UP_{uc\_n,t}=UC\_RHST_{uc\_n,t,'UP'}$$
$$VAR\_UCT.FX_{uc\_n,t}=UC\_RHST_{uc\_n,t,'FX'}$$

6.4.2.4 Equation: EQ(*l*)_UCRT / EQE_UCRT

Indices: user constraint (uc_n), region (r), period (t)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET, VAR_CUMCOM, VAR_CUMFLO

Purpose: The user constraint **EQ(*l*)_UCRT** is a user constraint, which is created for each region of **uc_r_each** and each period of **uc_t_each** and is summing over timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UCRT_{r,uc\_n,t} \ni UC\_RHSRT_{r,uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}$$

$$\sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'}\right)$$

when control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$
$$UC\_RHSRT_{r,uc\_n,t,l} + UC\_TIME_{uc\_n,r,t} \times D_t$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the LHS is set equal to the variable VAR_UCRT. The bounds UC_RHSRT are then applied to the variable VAR_UCRT.

$$=$$
$$VAR\_UCRT_{r,uc\_n,t} + UC\_TIME_{uc\_n,r,t} \times D_t$$

with

$$VAR\_UCRT.LO_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'LO'}$$
$$VAR\_UCRT.UP_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'UP'}$$
$$VAR\_UCRT.FX_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'FX'}$$

6.4.2.5 Equation: EQ(*l*)_UCRTS / EQE_UCRTS

Indices: user constraint (uc_n), region (r), period (t), timeslice (s)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET, VAR_CUMCOM, VAR_CUMFLO

Purpose: The user constraint **EQ(*l*)_UCRTS** is a user constraint, which is created for each region of **uc_r_each**, each period of **uc_t_each** and each timeslice of **uc_ts_each**.

Equation:

$$EQ(l)\_UCRTS_{r,uc\_n,t,s} \ni UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}} \wedge \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}$$

$$\begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'}\right)$$

when control parameter VAR_UC=NO:

$$\{\leq;=;\geq\}$$
$$UC\_RHSRTS_{r,uc\_n,t,s,l} + UC\_TIME_{uc\_n,r,t} \times D_t$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the LHS is set equal to the variable VAR_UCRTS. The bounds UC_RHSRTS are then applied to the variable VAR_UCRTS.

$$=$$
$$VAR\_UCRTS_{r,uc\_n,t,s} + UC\_TIME_{uc\_n,r,t} \times D_t$$

with

$$VAR\_UCRTS.LO_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'LO'}$$
$$VAR\_UCRTS.UP_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'UP'}$$
$$VAR\_UCRTS.FX_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'FX'}$$

6.4.2.6 Equation: EQ(*l*)_UCTS / EQE_UCTS

Indices: user constraint (uc_n), period (t), timeslice (s)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET, VAR_CUMCOM, VAR_CUMFLO

Purpose: The user constraint **EQ(*l*)_UCTS** is a user constraint, which is created for each period of **uc_t_each** and each timeslice of **uc_ts_each** and is summing over regions (**uc_r_sum**).

Equation:

$$EQE\_UCTS_{uc\_n,t,s} \ni UC\_RHSRTS_{uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum_{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_each_{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'} \right)$$

when control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$
$$UC\_RHSTS_{uc\_n,t,s,l} + \sum_{r \in \mathbf{uc\_r\_sum}} UC\_TIME_{uc\_n,r,t} \times D_t$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the LHS is set equal to the variable VAR_UCTS. The bounds UC_RHSTS are then applied to the variable VAR_UCTS.

$$=$$
$$VAR\_UCTS_{uc\_n,t,s} + \sum_{r \in \mathbf{uc\_r\_sum}} UC\_TIME_{uc\_n,r,t} \times D_t$$

with

$$VAR\_UCTS.LO_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'LO'}$$
$$VAR\_UCTS.UP_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'UP'}$$
$$VAR\_UCTS.FX_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'FX'}$$

6.4.3 Timeslice-dynamic user constraints

Mathematical formulation of timeslice-dynamic user constraints

In the mathematical description of the different variants of timeslice-dynamic user constraints, on the LHS the same placeholders can be used as for pure LHS user constraints: $ACT_{r,t,p,s,'LHS'}$, $FLO_{r,t,p,s,'LHS'}$, $IRE_{r,t,p,s,'LHS'}$, $COMPRD_{r,t,s,'LHS'}$, $COMNET_{r,t,s,'LHS'}$, $NCAP_{r,t,p,s,'LHS'}$, $CAP_{r,t,p,s,'LHS'}$. The LHS placeholder expressions are identical to those of LHS user constraints.

On the RHS (preceding timeslice ***ds*** = ***s***–RS_STG(r,s)), the following placeholders can be used: $ACT_{r,t,p,ds,'RHS'}$, $FLO_{r,t,p,ds,'RHS'}$, $IRE_{r,t,p,ds,'RHS'}$, $COMPRD_{r,t,ds,'RHS'}$, $COMNET_{r,t,ds,'RHS'}$, $NCAP_{r,t,p,ds,'RHS'}$, $CAP_{r,t,p,ds,'RHS'}$. The RHS placeholder expressions can be written by replacing in them the timeslice index **s** by **d(s)**.

Note that the timeslice-specific terms in the equation are all divided by $G\_YRFR_{r,s}$ in order to make it easy to combine flow and capacity terms in the constraint.

6.4.3.1 Equation: EQ(*l*)_UCRS / EQE_UCRS

Indices: user constraint (uc_n), period (t), timeslice (s)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The timeslice-dynamic constraint **EQ(*l*)_UCRS** establishes a constraint between two successive timeslices **s** and **d(s)** = **s**–RS_STG(**r,s**). The constraint is generated for all regions **r** in the set **uc_r_each**, all periods **t** in the set **uc_t_each**, and for each timeslice on the timeslice level defined by **uc_tsl**, such that also have the corresponding $UC\_RHSRT_{r,uc\_n,t,s,bd}$ specified.

Equation:

$$EQ(l)\_UCRS_{r,uc\_n,t,tsl,s} \ni UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \mathbf{uc\_r\_each_{r,uc\_n}} \wedge \mathbf{uc\_t\_each_{r,uc\_n,t}}$$
$$\wedge (s \in \{ts \mid \mathbf{ts\_grp}_{r,tsl,ts} \wedge \bigcup_{side} \mathbf{uc\_tsl}_{r,uc\_n,side,tsl}\})$$

$$\begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix} \times \frac{1}{G\_YRFR_{r,s}}$$
$$+$$
$$\left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'}\right)$$

When control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$

$$UC\_RHSRTS_{r,uc\_n,t,s,l} + UC\_TIME_{uc\_n,r,t} \times D_t$$

$$+$$

$$\left(\begin{array}{l} \left(\begin{array}{l} ACT_{r,t,d(s),'RHS'} + FLO_{r,t,d(s),'RHS'} + IRE_{r,t,d(s),'RHS'} \\ + COMNET_{r,t,d(s),'RHS'} + COMPRD_{r,t,d(s),'RHS'} \end{array} \right) \times \frac{1}{G\_YRFR_{r,d(s)}} \\ + \\ \left(NCAP_{r,t,'RHS'} + CAP_{r,t,'RHS'} \right) \\ \end{array} \right) \textit{if } \mathbf{uc\_tsl}_{r,uc\_n,'RHS',s}$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHSRTS is replaced by the variable VAR_UCRTS. The bounds UC_RHSRTS are then applied to the variable VAR_UCRTS.

$$=$$

$$VAR\_UCRTS_{r,uc\_n,t,s} + UC\_TIME_{uc\_n,r,t} \times D_t$$

$$+$$

$$\left(\begin{array}{l} \left(\begin{array}{l} ACT_{r,t,d(s),'RHS'} + FLO_{r,t,d(s),'RHS'} + IRE_{r,t,d(s),'RHS'} \\ + COMNET_{r,t,d(s),'RHS'} + COMPRD_{r,t,d(s),'RHS'} \end{array} \right) \times \frac{1}{G\_YRFR_{r,d(s)}} \\ + \\ \left(NCAP_{r,t,'RHS'} + CAP_{r,t,'RHS'} \right) \\ \end{array} \right) \textit{if } \mathbf{uc\_tsl}_{r,uc\_n,'RHS',s}$$

with

$$VAR\_UCRTS.LO_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'LO'}$$

$$VAR\_UCRTS.UP_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'UP'}$$

$$VAR\_UCRTS.FX_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'FX'}$$

6.4.4 Dynamic user constraints of type (t,t+1)

Mathematical formulation of dynamic user constraints and growth constraints of type (t,t+1)

In the mathematical description of the dynamic user constraints and growth constraints of type (**t**, **t+1**), the following placeholders are used for variable terms on the LHS (period **t**): $ACT\_GROW_{r,t,p,s,'LHS'}$, $FLO\_GROW_{r,t,p,s,'LHS'}$, $IRE\_GROW_{r,t,p,s,'LHS'}$, $COMPRD\_GROW_{r,t,s,'LHS'}$, $COMNET\_GROW_{r,t,s,'LHS'}$, $NCAP\_GROW_{r,t,p,'LHS'}$, $CAP\_GROW_{r,t,p,'LHS'}$

and on the RHS (period **t+1**):
$ACT\_GROW_{r,t+1,p,s,'RHS'}$, $FLO\_GROW_{r,t+1,p,s,'RHS'}$, $IRE\_GROW_{r,t+1,p,s,'RHS'}$, $COMPRD\_GROW_{r,t+1,s,'RHS'}$, $COMNET\_GROW_{r,t+1,s,'RHS'}$, $NCAP\_GROW_{r,t+1,p,'RHS'}$, $CAP\_GROW_{r,t+1,p,'RHS'}$.

The expressions for the variable terms on the LHS can be written as follows:

$$ACT\_GROW_{r,t,s,'LHS'}$$

$$=$$

$$\sum_{(p,v)\in \mathbf{rtp\_vintyr_{r,v,t,p}}} \sum_{ts\in \mathbf{prc\_ts_{r,p,ts}}} \left(\begin{array}{l} VAR\_ACT_{r,v,t,p,ts} \times UC\_ACT_{uc\_n,'LHS',r,t,p,ts} \times \left(RS\_FR_{r,s,ts}\right) \\ \times \\ \left(\sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ACOST_{r,t,p,cur} \right) \quad if\ UC\_ATTR_{r,uc\_n,'LHS','ACT','COST'}\ is\ given \\ \times \\ \left(UC\_ACT_{uc\_n,'LHS',r,t,p,ts}\right)^{M(t+1)-M(t)-1} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','ACT','GROWTH'}\ is\ given \end{array} \right)$$

$$
\begin{aligned}
&FLO\_GROW_{r,t,s,'LHS'} \\
&= \\
&\sum_{(p,c,ts)\in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \\
&\left\{
\begin{array}{l}
VAR\_FLO_{r,v,t,p,c,ts} \times UC\_FLO_{uc\_n,'RHS',r,t,p,c,ts} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\
\times \\
\left[\sum_{cur \in \mathbf{rdcur}_{\mathbf{r,cur}}} \left(\begin{array}{l}
OBJ\_FCOST_{r,t,p,c,ts,cur} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','FLO','COST'} \text{ is given} \\
+ \\
OBJ\_FDELV_{r,t,p,c,ts,cur} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','FLO','DELIV'} \text{ is given} \\
+ \\
OBJ\_FSUB_{r,t,p,c,ts,cur} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','FLO','SUB'} \text{ is given} \\
+ \\
OBJ\_FTAX_{r,t,p,c,ts,cur} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','FLO','TAX'} \text{ is given}
\end{array} \right) \right] \\
\times \\
\left(UC\_FLO_{uc\_n,'LHS',r,t,p,c,ts} \right)^{M(t+1)-M(t)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','FLO','GROWTH'} \text{ is given}
\end{array}
\right\}
\end{aligned}
$$

$$
\begin{aligned}
&IRE\_GROW_{r,t,s,'LHS'} \\
&= \sum_{(p.c,ts)\in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \sum_{ie \in \mathbf{rpc\_ire}_{\mathbf{r,p,c,ie}}} \\
&\left[
\begin{array}{l}
VAR\_IRE_{r,v,t,p,c,ts,ie} \times UC\_IRE_{uc\_n,'LHS',r,t,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\
\times \\
\sum_{cur \in \mathbf{rdcur}_{r,cur}} \left(\begin{array}{l}
OBJ\_FCOST_{r,t,p,c,s,cur} \text{ if } \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','COST'} \\
+ \\
OBJ\_FDELV_{r,t,p,c,s,cur} \text{ if } \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','DELIV'} \\
- \\
OBJ\_FSUB_{r,t,p,c,s,cur} \quad \text{if } \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','SUB'} \\
+ \\
OBJ\_FTAX_{r,t,p,c,s,cur} \quad \text{if } \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','TAX'}
\end{array} \right) \\
\times \left(UC\_IRE_{uc\_n,'LHS',r,t,p,c,ts,ie} \right)^{M(t+1)-M(t)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','IRE','GROWTH'} \text{ is given}
\end{array}
\right]
\end{aligned}
$$

$$COMPRD\_GROW_{r,t,s,'LHS'}$$

$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{r,t,c,ts}} \left(\begin{array}{l} VAR\_COMPRD_{r,t,c,ts} \times UC\_COMPRD_{uc\_n,'LHS',r,t,c,s} \\ \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMPRD',uc\_cost} \right) \\ \times \left(UC\_COMPRD_{uc\_n,'LHS',r,t,c,s} \right)^{M(t+1)-M(t)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMPRD','GROWTH'}\ given \end{array} \right)$$

$$COMNET\_GROW_{r,t,s,'LHS'}$$

$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{r,t,c,ts}} \left(\begin{array}{l} VAR\_COMNET_{r,t,c,ts} \times UC\_COMNET_{uc\_n,'LHS',r,t,c,s} \\ \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMNT_{r,t,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMNET',uc\_cost} \right) \\ \times \left(UC\_COMNET_{uc\_n,'LHS',r,t,c,s} \right)^{M(t+1)-M(t)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMNET','GROWTH'}\ given \end{array} \right)$$

$$NCAP\_GROW_{r,t,p,'LHS'}$$

$$= \sum_{p} \left[\begin{array}{l} VAR\_NCAP_{r,t,p} \times UC\_NCAP_{uc\_n,'LHS',r,t,p} \times \\ \left(\begin{array}{l} \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','COST'}\ is\ given \\ - \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','SUB'}\ is\ given \\ + \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','TAX'}\ is\ given \end{array} \right) \\ \times \left(UC\_NCAP_{uc\_n,'LHS',r,t,p} \right)^{M(t+1)-M(t)-1} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','GROWTH'}\ given \end{array} \right]$$

$$
\begin{aligned}
&CAP\_GROW_{r,t,'LHS'} \\
&= \\
&\sum_{p}\left(\begin{array}{l}
VAR\_CAP_{r,t,p} \times UC\_CAP_{uc\_n,'LHS',r,t,p} \\
\times \\
PRC\_ACTFLO_{r,p} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','CAP','CAPACT'} \text{ is given} \\
\times \\
\left(UC\_CAP_{uc\_n,'LHS',r,t,p}\right)^{M(t+1)-M(t)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','CAP','GROWTH'} \text{ is given}
\end{array}\right)
\end{aligned}
$$

The expressions for the variable terms on the RHS can be written as follows:

$$
\begin{aligned}
&ACT\_GROW_{r,t+1,s,'RHS'} \\
&= \\
&\sum_{(p,v)\in \mathbf{rtp\_vintyr_{r,v,t+1,p}}} \; \sum_{ts\in \mathbf{prc\_ts_{r,p,ts}}} \left(\begin{array}{l}
VAR\_ACT_{r,v,t+1,p,ts} \times UC\_ACT_{uc\_n,'RHS',r,t+1,p,ts} \times \left(RS\_FR_{r,s,ts}\right) \\
\times \\
\left(\sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ACOST_{r,t+1,p,cur}\right) \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','ACT','COST'} \text{ is given} \\
\times \\
\left(UC\_ACT_{uc\_n,'RHS',r,t+1,p,ts}\right)^{M(t)-M(t+1)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','ACT','GROWTH'} \text{ is given}
\end{array}\right)
\end{aligned}
$$

$$
\begin{aligned}
&FLO\_GROW_{r,t+1,s,'RHS'} \\
&= \\
&\sum_{(p,c,ts)\in \mathbf{rtpcs\_varf_{r,t+1,p,c,ts}}} \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t+1,p}}} \\
&\left\{
\begin{array}{l}
VAR\_FLO_{r,v,t+1,p,c,ts} \times UC\_FLO_{uc\_n,'RHS',r,t+1,p,c,ts} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\
\times \\
\left[\sum_{cur\in \mathbf{rdcur_{r,cur}}} \left(\begin{array}{ll}
OBJ\_FCOST_{r,t+1,p,c,ts,cur} & if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','COST'}\ is\ given \\
+ & \\
OBJ\_FDELV_{r,t+1,p,c,ts,cur} & if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','DELIV'}\ is\ given \\
+ & \\
OBJ\_FSUB_{r,t+1,p,c,ts,cur} & if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','SUB'}\ is\ given \\
+ & \\
OBJ\_FTAX_{r,t+1,p,c,ts,cur} & if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','TAX'}\ is\ given
\end{array} \right) \right] \\
\times \\
\left(UC\_FLO_{uc\_n,'RHS',r,t+1,p,c,ts} \right)^{M(t)-M(t+1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','GROWTH'}\ is\ given
\end{array}
\right\}
\end{aligned}
$$

$$
\begin{aligned}
&IRE\_GROW_{r,t+1,s,'RHS'} \\
&= \sum_{(p,c,ts)\in \mathbf{rtpcs\_varf_{r,t+1,p,c,ts}}} \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t+1,p}}} \sum_{ie\in \mathbf{rpc\_ire_{r,p,c,ie}}} \\
&\left[
\begin{array}{l}
VAR\_IRE_{r,v,t+1,p,c,ts,ie} \times UC\_IRE_{uc\_n,'RHS',r,t+1,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t+1,c,s,ts} \right) \\
\times \\
\sum_{cur\in \mathbf{rdcur}_{r,cur}} \left(\begin{array}{ll}
OBJ\_FCOST_{r,t+1,p,c,s,cur} & if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','COST'} \\
+ & \\
OBJ\_FDELV_{r,t+1,p,c,s,cur} & if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','DELIV'} \\
- & \\
OBJ\_FSUB_{r,t+1,p,c,s,cur} & if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','SUB'} \\
+ & \\
OBJ\_FTAX_{r,t+1,p,c,s,cur} & if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','TAX'}
\end{array} \right) \\
\times \left(UC\_IRE_{uc\_n,'RHS',r,t+1,p,c,ts,ie} \right)^{M(t)-M(t+1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','GROWTH'}\ given
\end{array}
\right]
\end{aligned}
$$

$$COMPRD\_GROW_{r,t+1,s,'RHS'}$$

$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t,+1c,ts}}} \left(\begin{array}{l} VAR\_COMPRD_{r,t+1,c,ts} \times UC\_COMPRD_{uc\_n,'RHS',r,t+1,c,s} \\ \times \left(RTCS\_TSFR_{r,t+1,c,s,ts} \right) \times \\ \sum\limits_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t+1,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD',uc\_cost} \right) \\ \times \left(UC\_COMPRD_{uc\_n,'RHS',r,t+1,c,s} \right)^{M(t)-M(t+1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD','GROWTH'}\ given \end{array} \right)$$

$$COMNET\_GROW_{r,t+1,s,'RHS'}$$

$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t+1,c,ts}}} \left(\begin{array}{l} VAR\_COMNET_{r,t+1,c,ts} \times UC\_COMNET_{uc\_n,'RHS',r,t+1,c,s} \\ \times \left(RTCS\_TSFR_{r,t+1,c,s,ts} \right) \times \\ \sum\limits_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMNT_{r,t+1,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET',uc\_cost} \right) \\ \times \left(UC\_COMNET_{uc\_n,'RHS',r,t+1,c,s} \right)^{M(t)-M(t+1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET','GROWTH'}\ given \end{array} \right)$$

$$NCAP\_GROW_{r,t+1,'RHS'}$$

$$=$$

$$\sum_{p} \left(\begin{array}{l} VAR\_NCAP_{r,t+1,p} \times UC\_NCAP_{uc\_n,'RHS',r,t+1,p} \\ \times \\ \left(\begin{array}{l} \sum\limits_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t+1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','COST'}\ is\ given \\ - \\ \sum\limits_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t+1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','SUB'}\ is\ given \\ + \\ \sum\limits_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t+1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','TAX'}\ is\ given \end{array} \right) \\ \times \\ \left(UC\_NCAP_{uc\_n,'RHS',r,t+1,p} \right)^{M(t)-M(t+1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','GROWTH'}\ is\ given \end{array} \right)$$

$$CAP\_GROW_{r,t+1,'RHS'}$$

$$=$$

$$\sum_{p}\left(\begin{array}{l} VAR\_CAP_{r,t+1,p} \times UC\_CAP_{uc\_n,'RHS',r,t+1,p} \\ \times \\ PRC\_CAPACT_{r,p} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','CAP','CAPACT'}\ is\ given \\ \times \\ \left(UC\_CAP_{uc\_n,'RHS',r,t+1,p}\right)^{M(t)-M(t+1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','CAP','GROWTH'}\ is\ given \end{array}\right)$$

6.4.4.1 Equation: EQ(*l*)_UCSU / EQE_UCSU

Indices: user constraint (uc_n), period (t)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The dynamic user constraint or growth constraint of type (**t,t+1**) **EQ(*l*)_UCSU** establishes a constraint between two successive periods **t** and **t+1**. For dynamic user constraints the period **t** is specified by the set **uc_t_succ**, growth constraints are generated for all periods bur the last. The constraint is summing over regions (**uc_r_sum**) and timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UCSU_{uc\_n,t} \ni UC\_RHST_{uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum_{r,uc\_n}}$$
$$\wedge \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'} \right)$$

When control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$
$$UC\_RHST_{uc\_n,t,l}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'} \right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHST is replaced by the variable VAR_UCT. The bounds UC_RHST are then applied to the variable VAR_UCT.

$$=$$

$$VAR\_UCT_{uc\_n,t}$$

$$+$$

$$\sum_{r\in\mathbf{uc\_r\_sum}}\sum_{s\in\mathbf{uc\_ts\_sum}}\begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\sum_{r\in\mathbf{uc\_r\_sum}}\left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'}\right)$$

with

$$VAR\_UCT.LO_{uc\_n,t} = UC\_RHST_{uc\_n,t,'LO'}$$
$$VAR\_UCT.UP_{uc\_n,t} = UC\_RHST_{uc\_n,t,'UP'}$$
$$VAR\_UCT.FX_{uc\_n,t} = UC\_RHST_{uc\_n,t,'FX'}$$

6.4.4.2 Equation: EQ(*l*)_UCRSU / EQE_UCRSU

Indices: region (r), user constraint (uc_n), period (t)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The dynamic user constraint or growth constraint of type (**t,t+1**) **EQ(*l*)_UCSU** establishes a constraint between two successive periods **t** and **t+1**. For dynamic user constraints the period **t** is specified by the set **uc_t_succ**, growth constraints are generated for all periods but the last. The constraint is generated for each region of the set **uc_r_each** and is summing over timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UCRSU_{r,uc\_n,t} \ni UC\_RHSRT_{r,uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each_{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$

$$\sum_{s\in\mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'}\right)$$

When control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$
$$UC\_RHSRT_{r,uc\_n,t,l}$$
$$+$$
$$\sum_{s\in\mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'}\right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHSRT is replaced by the variable VAR_UCRT. The bounds UC_RHSRT are then applied to the variable VAR_UCRT.

$$=$$

$$VAR\_UCRT_{r,uc\_n,t}$$

$$+$$

$$\sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'}\right)$$

with

$$VAR\_UCRT.LO_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'LO'}$$

$$VAR\_UCRT.UP_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'UP'}$$

$$VAR\_UCRT.FX_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'FX'}$$

6.4.4.3 Equation: EQ(*l*)_UCRSUS / EQE_UCRSUS

Indices: region (r), user constraint (uc_n), period (t), timeslice (s)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The dynamic user constraint or growth constraint of type (**t,t+1**) **EQ(*l*)_UCSUS** establishes a constraint between two successive periods **t** and **t+1**. For dynamic user constraints the period **t** is specified by the set **uc_t_succ**, growth constraints are generated for all periods but the last. The constraint is generated for each region of the set **uc_r_each** and each timeslice of the set **uc_ts_each**.

Equation:

$$EQ(l)\_UCRSUS_{r,uc\_n,t,s} \ni UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}$$

$$\wedge\, \mathbf{uc\_t\_succ}_{\mathbf{r,uc\_n,t}}$$

$$\begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$

$$+$$

$$\left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'}\right)$$

When control parameter VAR_UC=NO:

$$\{\leq;=;\geq\}$$

$$UC\_RHSRTS_{r,uc\_n,t,s,l}$$

$$+$$

$$\begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'}\right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHSRTS is replaced by the variable VAR_UCRTS. The bounds UC_RHSRTS are then applied to the variable VAR_UCRTS.

$$=$$

$$VAR\_UCRTS_{r,uc\_n,t,s}$$

$$+$$

$$\left(\begin{array}{l} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{array}\right)$$

$$+$$

$$\left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'}\right)$$

with

$$VAR\_UCRTS.LO_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'LO'}$$

$$VAR\_UCRTS.UP_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'UP'}$$

$$VAR\_UCRTS.FX_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'FX'}$$

6.4.4.4 Equation: EQ(*l*)_UCSUS / EQE_UCSUS

Indices: user constraint (uc_n), period (t), timeslice (s)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The dynamic user constraint or growth constraint of type (**t,t+1**) **EQ(*l*)_UCSUS** establishes a constraint between two successive periods **t** and **t+1**. For dynamic user constraints the period **t** is specified by the set **uc_t_succ**, growth constraints are generated for all periods but the last. The constraint generated for each timeslice **uc_ts_each** and is summing over regions (**uc_r_sum**).

Equation:

$$EQ(l)\_UCSUS_{uc\_n,t,s} \ni UC\_RHSTS_{uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}$$

$$\wedge\, \mathbf{uc\_t\_succ}_{\mathbf{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'} \right)$$

When control parameter VAR_UC=NO:

$$\{\leq;=;\geq\}$$

$$UC\_RHSTS_{uc\_n,t,s,l}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'} \right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHSTS is replaced by the variable VAR_UCTS. The bounds UC_RHSTS are then applied to the variable VAR_UCTS.

$$=$$

$$VAR\_UCTS_{uc\_n,t,s}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'} \right)$$

with

$$VAR\_UCTS.LO_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'LO'}$$

$$VAR\_UCTS.UP_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'UP'}$$

$$VAR\_UCTS.FX_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'FX'}$$

6.4.5 Dynamic user constraints of type (t–1,t)

Mathematical formulation of dynamic user constraints and growth constraints of type (t-1, t)

In the mathematical description of the dynamic user constraints and growth constraints of type (**t–1, t**), the following placeholders are used for variable terms on the RHS (period **t–1**): $ACT\_GROW_{r,t-1,p,s,'RHS'}$, $FLO\_GROW_{r,t-1,p,s,'RHS'}$, $IRE\_GROW_{r,t-1,p,s,'RHS'}$, $COMPRD\_GROW_{r,t-1,s,'RHS'}$, $COMNET\_GROW_{r,t-1,s,'RHS'}$, $NCAP\_GROW_{r,t-1,p,'RHS'}$, $CAP\_GROW_{r,t-1,p,'RHS'}$.

For the LHS terms (period **t**), the placeholders are the same ones as defined for the LHS user constraints.

The expressions for the variable terms on the RHS can be written as follows:

$$ACT\_GROW_{r,t-1,s,'RHS'}$$

$$=$$

$$\sum_{(p,v)\in \mathbf{rtp\_vintyr_{r,v,t-1,p}}} \sum_{ts\in \mathbf{prc\_ts_{r,p,ts}}} \left(\begin{array}{l} VAR\_ACT_{r,v,t-1,p,ts} \times UC\_ACT_{uc\_n,'RHS',r,t-1,p,ts} \times \left(RS\_FR_{r,s,ts}\right) \\ \times \\ \left(\sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ACOST_{r,t-1,p,cur} \right) \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','ACT','COST'} \text{ is given} \\ \times \\ \left(UC\_ACT_{uc\_n,'RHS',r,t-1,p,ts}\right)^{M(t)-M(t-1)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','ACT','GROWTH'} \text{ is given} \end{array} \right)$$

$$FLO\_GROW_{r,t-1,s,'RHS'}$$

$$=$$

$$\sum_{(p,c,ts)\in\mathbf{rtpcs\_varf_{r,t-1,p,c,ts}}} \sum_{v\in\mathbf{rtp\_vintyr_{r,v,t-1,p}}}$$

$$\left\{\begin{array}{l} VAR\_FLO_{r,v,t-1,p,c,ts} \times UC\_FLO_{uc\_n,'RHS',r,t-1,p,c,ts} \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \\ \times \\ \left[\sum_{cur\in\mathbf{rdcur_{r,cur}}} \left(\begin{array}{l} OBJ\_FCOST_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','COST'}\ is\ given \\ + \\ OBJ\_FDELV_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','DELIV'}\ is\ given \\ + \\ OBJ\_FSUB_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','SUB'}\ is\ given \\ + \\ OBJ\_FTAX_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','TAX'}\ is\ given \end{array} \right) \right] \\ \times \\ \left(UC\_FLO_{uc\_n,'RHS',r,t-1,p,c,ts} \right)^{M(t)-M(t-1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','GROWTH'}\ is\ given \end{array}\right\}$$

$$IRE\_GROW_{r,t-1,s,'RHS'}$$

$$= \sum_{(p,c,ts)\in\mathbf{rtpcs\_varf_{r,t-1,p,c,ts}}} \sum_{v\in\mathbf{rtp\_vintyr_{r,v,t-1,p}}} \sum_{ie\in\mathbf{rpc\_ire_{r,p,c,ie}}}$$

$$\left[\begin{array}{l} VAR\_IRE_{r,v,t-1,p,c,ts,ie} \times UC\_IRE_{uc\_n,'RHS',r,t-1,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \\ \times \\ \sum_{cur\in\mathbf{rdcur}_{r,cur}} \left(\begin{array}{l} OBJ\_FCOST_{r,t-1,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','COST'} \\ + \\ OBJ\_FDELV_{r,t-1,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','DELIV'} \\ - \\ OBJ\_FSUB_{r,t-1,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','SUB'} \\ + \\ OBJ\_FTAX_{r,t-1,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','TAX'} \end{array} \right) \\ \times \left(UC\_IRE_{uc\_n,'RHS',r,t-1,p,c,ts,ie} \right)^{M(t)-M(t-1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','GROWTH'}\ given \end{array}\right]$$

$$COMPRD\_GROW_{r,t-1,s,'RHS'} = \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{r,t,-1c,ts}} \begin{pmatrix} VAR\_COMPRD_{r,t-1,c,ts} \times UC\_COMPRD_{uc\_n,'RHS',r,t-1,c,s} \\ \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t-1,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD',uc\_cost} \right) \\ \times \left(UC\_COMPRD_{uc\_n,'RHS',r,t-1,c,s} \right)^{M(t)-M(t-1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD','GROWTH'}\ given \end{pmatrix}$$

$$COMNET\_GROW_{r,t-1,s,'RHS'} = \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{r,t-1,c,ts}} \begin{pmatrix} VAR\_COMNET_{r,t-1,c,ts} \times UC\_COMNET_{uc\_n,'RHS',r,t-1,c,s} \\ \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMNT_{r,t-1,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET',uc\_cost} \right) \\ \times \left(UC\_COMNET_{uc\_n,'RHS',r,t-1,c,s} \right)^{M(t)-M(t-1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET','GROWTH'}\ given \end{pmatrix}$$

$$NCAP\_GROW_{r,t-1,'RHS'} = \sum_{p} \begin{pmatrix} VAR\_NCAP_{r,t-1,p} \times UC\_NCAP_{uc\_n,'RHS',r,t-1,p} \\ \times \\ \begin{pmatrix} \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t-1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','COST'}\ is\ given \\ - \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t-1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','SUB'}\ is\ given \\ + \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t-1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','TAX'}\ is\ given \end{pmatrix} \\ \times \\ \left(UC\_NCAP_{uc\_n,'RHS',r,t-1,p} \right)^{M(t)-M(t-1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','GROWTH'}\ is\ given \end{pmatrix}$$

$$CAP\_GROW_{r,t-1,'RHS'}$$
$$=$$
$$\sum_{p}\left(\begin{array}{l} VAR\_CAP_{r,t-1,p} \times UC\_CAP_{uc\_n,'RHS',r,t-1,p} \\ \times \\ PRC\_CAPACT_{r,p} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','CAP','CAPACT'}\ is\ given \\ \times \\ \left(UC\_CAP_{uc\_n,'RHS',r,t-1,p}\right)^{M(t+1)-M(t)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','CAP','GROWTH'}\ is\ given \end{array}\right)$$

6.4.5.1 Equation: EQ(*l*)_UCSU / EQE_UCSU

Indices: user constraint (uc_n), period (t)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The growth constraint of type (**t–1,t**) **EQ(*l*)_UCSU** establishes a constraint between two successive periods **t–1** and **t**. The growth constraint is generated for all periods **t** having the *UC_RHST* constant specified. The constraint is summing over regions (**uc_r_sum**) and timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UCSU_{uc\_n,t} \ni UC\_RHST_{uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum_{r,uc\_n}} \wedge \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \left(\begin{array}{l} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{array} \right)$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'} \right)$$

When control parameter VAR_UC=NO:

$$\{\leq ; = ; \geq\}$$

$$UC\_RHST_{uc\_n,t,l}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \left(\begin{array}{l} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{array} \right)$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'} \right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHST is replaced by the variable VAR_UCT. The bounds UC_RHST are then applied to the variable VAR_UCT.

$$=$$

$$VAR\_UCT_{uc\_n,t}$$

$$+$$

$$\sum_{r\in\mathbf{uc\_r\_sum}}\sum_{s\in\mathbf{uc\_ts\_sum}}\left(\begin{aligned}&ACT\_GROW_{r,t-1,s,'RHS'}+FLO\_GROW_{r,t-1,s,'RHS'}+IRE\_GROW_{r,t-1,s,'RHS'}\\&+COMNET\_GROW_{r,t-1,s,'RHS'}+COMPRD\_GROW_{r,t-1,s,'RHS'}\end{aligned}\right)$$

$$+$$

$$\sum_{r\in\mathbf{uc\_r\_sum}}\left(NCAP\_GROW_{r,t-1,'RHS'}+CAP\_GROW_{r,t-1,'RHS'}\right)$$

with

$$VAR\_UCT.LO_{uc\_n,t}=UC\_RHST_{uc\_n,t,'LO'}$$

$$VAR\_UCT.UP_{uc\_n,t}=UC\_RHST_{uc\_n,t,'UP'}$$

$$VAR\_UCT.FX_{uc\_n,t}=UC\_RHST_{uc\_n,t,'FX'}$$

6.4.5.2 Equation: EQ(*l*)_UCRSU / EQE_UCRSU

Indices: region (r), user constraint (uc_n), period (t)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The growth constraint of type (**t–1,t**) **EQ(*l*)_UCSU** establishes a constraint between two successive periods **t–1** and **t**. The growth constraint is generated for all periods **t** having the *UC_RHSRT* attribute defined. The constraint is generated for each region of the set **uc_r_each** and is summing over timeslices (**uc_ts_sum**).

Equation:

$$EQ(l)\_UCRSU_{r,uc\_n,t} \ni UC\_RHSRT_{r,uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_succ}_{\mathbf{r,uc\_n,t}}$$

$$\sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'}\right)$$

When control parameter VAR_UC=NO:

$$\{\leq;=;\geq\}$$
$$UC\_RHSRT_{r,uc\_n,t,l}$$
$$+$$
$$\sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'}\right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHSRT is replaced by the variable VAR_UCRT. The bounds UC_RHSRT are then applied to the variable VAR_UCRT.

$$=$$

$$VAR\_UCRT_{r,uc\_n,t}$$

$$+$$

$$\sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'}\right)$$

with

$$VAR\_UCRT.LO_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'LO'}$$

$$VAR\_UCRT.UP_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'UP'}$$

$$VAR\_UCRT.FX_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'FX'}$$

6.4.5.3 Equation: EQ(*l*)_UCRSUS / EQE_UCRSUS

Indices: region (r), user constraint (uc_n), period (t), timeslice (s)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The growth constraint of type (**t–1,t**) **EQ(*l*)_UCSUS** establishes a constraint between two successive periods **t–1** and **t**. The growth constraint is generated for all periods **t** having the *UC_RHSRTS* attribute defined. The constraint is generated for each region of the set **uc_r_each** and each timeslice of the set **uc_ts_each**.

Equation:

$$EQ(l)\_UCRSUS_{r,uc\_n,t,s} \ni UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each_{r,uc\_n}}$$

$$\wedge\, \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$

$$\begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$

$$+$$

$$\left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'}\right)$$

When control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$

$$UC\_RHSRTS_{r,uc\_n,t,s,l}$$

$$+$$

$$\begin{pmatrix} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'}\right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHSRTS is replaced by the variable VAR_UCRTS. The bounds UC_RHSRTS are then applied to the variable VAR_UCRTS.

$$=$$

$$VAR\_UCRTS_{r,uc\_n,t,s}$$

$$+$$

$$\left(\begin{array}{l} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{array}\right)$$

$$+$$

$$\left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'}\right)$$

with

$$VAR\_UCRTS.LO_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'LO'}$$

$$VAR\_UCRTS.UP_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'UP'}$$

$$VAR\_UCRTS.FX_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'FX'}$$

6.4.5.4 Equation: EQ(*l*)_UCSUS / EQE_UCSUS

Indices: user constraint (uc_n), period (t), timeslice (s)

Related variables: VAR_FLO, VAR_IRE, VAR_NCAP, VAR_CAP, VAR_ACT, VAR_COMPRD, VAR_COMNET

Purpose: The growth constraint of type (**t-1,t**) **EQ(*l*)_UCSUS** establishes a constraint between two successive periods **t-1** and **t**. The growth constraint is generated for all periods **t** having the *UC_RHSTS* attribute defined. The constraint generated for each timeslice **uc_ts_each** and is summing over regions (**uc_r_sum**).

Equation:

$$EQ(l)\_UCSUS_{uc\_n,t,s} \ni UC\_RHSTS_{uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}$$
$$\wedge\, \mathbf{úc\_t\_succ}_{\mathbf{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'} \right)$$

When control parameter VAR_UC=NO:

$$\{\leq; =; \geq\}$$
$$UC\_RHSTS_{uc\_n,t,s,l}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'} \right)$$

When control parameter VAR_UC=YES, the user constraint is created as strict equality and the RHS constant UC_RHSTS is replaced by the variable VAR_UCTS. The bounds UC_RHSTS are then applied to the variable VAR_UCTS.

$$=$$

$$VAR\_UCTS_{uc\_n,t,s}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'} \right)$$

with

$$VAR\_UCTS.LO_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'LO'}$$
$$VAR\_UCTS.UP_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'UP'}$$
$$VAR\_UCTS.FX_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'FX'}$$

6.4.6 User constraint modifiers

6.4.6.1 Overview

The user constraint facility in TIMES provides a very powerful tool for specifying a large variety of custom user constraints in a TIMES model. Such constraints can refer to practically any combination of individual variables. Moreover, the constraint definitions can be optionally refined by specifying additional modifier attributes that are applied to specific components (variable terms) of the constraints. The modifier attributes available in the current version are listed in Table 25. The note "DYN only" in the table means that the attribute is valid for dynamic constraints only (the constraint is in those cases automatically defined as dynamic if the attribute is used).

As indicated in Table 25, one can easily specify, for example, that the FLO coefficients of the user constraint should apply to the sum of all annual flows in each period, by using the PERIOD attribute. In addition, as cumulative user constraints (summed over periods) are typically almost always meant to be applied also to the sum of the annual flows/activities in each period, the PERIOD modifier is now by default applied to the FLO, ACT, IRE, COMPRD and COMCON components of all cumulative constraints (this can be overridden by the explicit use of the input set **uc_ts_sum** for the constraint). The specification of various kinds of cumulative constraints is thus possible quite easily.

6.4.6.2 Cost modifiers (COST, TAX, SUB, DELIV)

The cost modifiers are applied to the variable terms by multiplying them with the corresponding cost attribute, or with the sum of multiple cost attributes, if several cost modifiers are specified. The expressions for the $FLO_{r,t,p,s,'LHS'}$ term above in section 6.4.2 give a detailed example of how the cost attributes are applied. The cost attributes applied are the following:

- COST: ACT_COST for ACT, FLO_COST for FLO/IRE, NCAP_COST for NCAP, COM_CSTNET for COMNET, and COM_CSTPRD for COMPRD
- TAX: FLO_TAX for FLO/IRE, NCAP_ITAX for NCAP, COM_TAXNET for COMNET, and COM_TAXPRD for COMPRD
- SUB: FLO_SUB for FLO/IRE, NCAP_ISUB for NCAP, COM_SUBNET for COMNET, and COM_SUBPRD for COMPRD
- DELIV: FLO_DELIV for FLO/IRE

6.4.6.3 Annuity modifiers (INVCOST, INVTAX, INVSUB)

The annuity modifiers are applied to the variable terms by summing the VAR_NACP variables over all vintage periods tt ≤ t that have an annual investment payment in period t, and multiplying these with the annual cost coefficient. The INVCOST modifier applies the investment cost payments, the INVTAX modifier the tax payments, and the INVSUB modifier the subsidy payments (taken as negative values). By combining several of these modifiers, the payments are summed together.

Table 25. User constraint modifier attributes available in TIMES.

Attribute	Description	Applicable UC components
COST	Multiple by primary cost attribute (summing together with other cost attributes requested)	NCAP,ACT, FLO,COMPRD, COMCON
TAX	Multiple by tax attribute (summing together with other cost attributes requested)	NCAP,FLO
SUB	Multiple by subsidy attribute (summing together with other cost attributes requested)	NCAP,FLO
DELIV	Multiple by delivery cost attribute (summing together with other cost attributes requested)	FLO
INVCOST	Multiply by investment cost annuities; implies CUMSUM	NCAP
INVTAX	Multiply by investment tax annuities; implies CUMSUM	NCAP
INVSUB	Multiply by investment subsidy annuities; implies CUMSUM	NCAP
CAPACT	Multiply by PRC_CAPACT	CAP
CAPFLO	Apply coefficients also to any capacity-related flows	FLO
CUMSUM	Sum over all periods up to current or previous period (DYN only)	All
EFF	Multiply by COM_IE (UC_COMPRD), divide by COM_IE (UC_COMCON)	COMPRD, COMCON
GROWTH	Interpret coefficients as annual change coefficients (DYN only)	All
NET	Apply to *net* production (UC_COMPRD) or consumption UC_COMCON)	COMPRD, COMCON
NEWFLO	Apply coefficient to the flows of the new vintage only	ACT, FLO, IRE
ONLINE	Apply coefficient to the on-line capacity only (assumed equal to the full capacity if ACT_MINLD has not been defined).	CAP
PERIOD	Multiply by period length (all but NCAP) or COEF_RPTI (NCAP)	All but CAP
SYNC	Synchronize LHS and RHS sides to refer to the same period	All (RHS only)
YES	Declares the constraint to be dynamic, of type (t−1, t)	All (RHS only)

6.4.6.4 <u>CAPACT modifier</u>

The CAPACT modifier is applied only to the CAP terms of user constraints, if such exist. The CAP term for each process is multiplied by the PRC_CAPACT parameter of that process when the modifier is used.

6.4.6.5 CAPFLO modifier

The CAPFLO modifier is applied only to the FLO terms of user constraints, if such exist. When the modifier is used, the FLO term for each process/commodity, which normally includes only the VAR_FLO variables, is augmented by the capacity-related flows of the same commodity, if such exists.

6.4.6.6 CUMSUM modifier

The CUMSUM attribute means that the corresponding variable term for any milestone year **t** consists of the cumulative sum of the variables in all previous periods up to (and including) the year **t**. For example, when combined with the INVCOST attribute, the resulting NCAP variable term represents the cumulative sum of capital cost annuities related to all new capacities installed up to the year **t**, which are paid in year **t**.

6.4.6.7 EFF modifier

The EFF modifier can currently only be applied to COMPRD/COMCON/COMNET, and it causes the variable terms to be either multiplied or divided by the commodity efficiency COM_IE. For COMPRD, the variable terms will refer to the net production after taking into account the commodity efficiency COM_IE. For COMCON, it will refer to the gross consumption before applying the commodity efficiency COM_IE, and with COMNET it will refer to the gross NET production (gross production minus gross consumption).

6.4.6.8 GROWTH modifier

The GROWTH modifier used for a variable term causes the user constraint coefficients specified for that term to be interpreted as annual growth/decay coefficients. For example, a coefficient of 1.1 will be interpreted as a growth coefficient corresponding to a 10% annual growth. The final effective coefficient applied to the variable term will be the growth/decay coefficient raised to the power (M(t)–M(t–1)) or (M(t+1)–M(t)), depending on the dynamic type.

6.4.6.9 NET modifier

The NET modifier can only be applied to COMPRD/COMCON. For COMPRD, it causes the variable terms to represent the net amount of production after consumption is subtracted (i.e. gross production in excess of gross consumption). For COMCON, it represents the net consumption in excess of net production (after applying COM_IE), which is normally a non-positive value. Using COMPRD with the NET modifier will thus result in the same variable term as when using COMNET with the EFF modifier.

6.4.6.10 NEWFLO modifier

The NEWFLO modifier can only be applied to the ACT, FLO and IRE variable terms. The modifier causes the variable terms to be restricted to the flows or activities of newly installed capacity vintages in the commissioning period only, whenever the process is vintaged. For non-vintaged processes, the modifier has no effect. Therefore, for making consistent use of the modifier, usually all processes referred to in the variable terms should be vintaged when this modifier is used.

6.4.6.11 ONLINE modifier

The ONLINE modifier can only be used with the CAP variable term. It causes the capacity term to be referring to the on-line capacity instead of the full installed capacity. However, in TIMES, the on-line capacity of a process may differ from the full capacity only when start-ups and shut-downs have been modelled, by defining a minimum stable operating level with the parameter ACT_MINLD.

6.4.6.12 PERIOD modifier

The PERIOD modifier can be used for all other components (variable terms) except for the CAP term. For the ACT, FLO, IRE, COMPRD, COMCON, and COMNET terms, the modifier adds the multiplier D(t), i.e. they are multiplied by the period length. For the NCAP term, the modifier multiplies the new capacity variable for each period by the number of repeated investments in that period.

6.4.6.13 SYNC modifier

The SYNC modifier can be used on the RHS for any component of a user component to signify that the RHS term should refer to the same period as the LHS term. It will also automatically declare the constraint to be dynamic of type (**t–1,t**), unless **uc_t_succ** is defined or the GROWTH modifier is used on the LHS side, which both force it to be of type (**t,t+1**).

6.4.6.14 YES modifier

The only function of the YES modifier is to declare the user constraint to be dynamic. The constraint will be of type (**t–1,t**), unless **uc_t_succ** is defined or the GROWTH modifier is used on the LHS side, which forces it to be of type (**t,t+1**). Using this modifier can thus be useful, if there are no other relevant modifiers to be used on the RHS of the constraint, which would automatically declare the constraint dynamic.

6.4.7 Non-binding user constraints

Non-binding user constraints of any type (intoduced for reporting purposes) can be defined in the same way as binding constraints, but using the 'N' lim type when specifying the UC_RHSxxx constant, with any value defined for it (–1 is recommended). Non-binding user constraints can only be defined when user constraint variables are enabled (i.e. when using the option VAR_UC==YES). The levels of the non-ninding constraints (i.e. the levels of the slack variables) are reported in the PAR_UCSL reporting attribute (see Section 3.3).

Appendix A
The TIMES Climate Module

1 Introduction

This Appendix contains the updated (November 2010) documentation on the Climate Module option for the TIMES model. It provides is a streamlined version of the older version, and contains 5 sections: section 2 contains a detailed description of the theoretical approach taken, section 3 describes the parameters of the climate module, section 4 the variables and section 5 the equations. This version of the documentation does not include the complete formulations for all of the Climate equations (in GAMS form), and neither does it include the full GAMS specifications. However, it should be sufficient to gain a complete understanding of the equations in mathematical form, and should enable the user to define the parameters of the climate module.

2 Mathematical formulation

The Climate Module starts from the global emissions of CO_2, CH_4, and N_2O, as generated by the TIMES global model, and proceeds to compute successively:

- the changes in CO_2, CH_4, and N_2O concentrations via three separate sets of equations,
- the total change (over pre-industrial times) in atmospheric radiative forcing resulting from the three gases plus an exogenously specified additional forcing resulting from other causes (other anthropogenic and/or natural causes, as defined by the user), and
- the temperature changes (over pre-industrial times) in two reservoirs (surface and deep ocean).

The Climate Equations used to perform these calculations were initially adapted from Nordhaus and Boyer (1999), who proposed linear recursive equations for calculating concentrations and temperature changes based on the CO_2 life cycle. These linear equations give results that are good approximations of those obtained from more complex climate models (Drouet *et al.*, 2004; Nordhaus and Boyer, 1999). The non-linear radiative forcing equation used by these authors and in TIMES is the same as the one used in most models. The choice of the Nordhaus and Boyer's climate equations is motivated by the simplicity of their approach and by the fact that their climate module is well-documented and acceptably accurate. In our implementation, the forcing equation has been replaced by a linear approximation whose values closely approach the exact ones as long as the useful range is carefully selected. This was done in order to keep the entire model linear, and therefore to allow the user to set constraints on forcing and on temperature as well as on concentrations and on emissions.

Rigorously, the concentration and forcing equations used in the climate module are applicable only to CO2 emissions, since the concentration equations simulate the carbon cycle. In order to model other GHGs, one way is to use these same equations, while replacing CO2 emissions by CO2-equivalent emissions of any number of gases endogenous to the model. However, a more detailed and generally preferable approach is to model separately the life cycle of each endogenous emission separately, and this is the approach used in TIMES. The additional forcing due to the remaining (non endogenous) emissions, is accounted for via an exogenous forcing quantity directly defined by the user.

We now describe the mathematical equations used at each of the three steps of the climate module.

2.1 Concentrations (accumulation of CO2, CH4, N2O)

a) CO2 accumulation is represented as the linear three-reservoir model below[45]: the atmosphere, the quickly mixing upper ocean + biosphere, and the deep ocean. CO2 flows in both directions between adjacent reservoirs. The 3-reservoir model is represented by the following 3 equations when the step of the recursion is equal to one year:

$$M_{atm}(y) = E(y) + (1 - \varphi_{atm\text{-}up})\, M_{atm}(y-1) + \varphi_{up\text{-}atm}\, M_{up}(y-1) \quad (1)$$

$$M_{up}(y) = (1 - \varphi_{up\text{-}atm} - \varphi_{up\text{-}lo})\, M_{up}(y-1) + \varphi_{atm\text{-}up}\, M_{atm}(y-1) + \varphi_{lo\text{-}up}\, M_{lo}(y-1) \quad (2)$$

$$M_{lo}(y) = (1 - \varphi_{lo\text{-}up})\, M_{lo}(y-1) + \varphi_{up\text{-}lo}\, M_{up}(y-1) \quad (3)$$

with

- $M_{atm}(y)$, $M_{up}(y)$, $M_{lo}(y)$: masses of CO_2 in atmosphere, in a quickly mixing reservoir representing the upper level of the ocean and the biosphere, and in deep oceans (GtC), respectively, in year *y* (GtC)
- $E(y-1)$ = CO_2 emissions in previous year (GtC)
- φ_{ij}, transport rate from reservoir *i* to reservoir *j* (*i, j = atm, up, lo*) from year *y-1* to *y*

b) CH4 accumulation is represented by a so-called single-box model in which the atmospheric methane concentration obeys the following equations assuming a constant annual decay rate of the anthropogenic concentrations Φ_{CH4} (whereas the natural concentration is assumed in equilibrium):

$$CH4_{atm}(y) = (1 - \Phi_{CH4}) \cdot CH4_{atm}(y-1) + EA_{CH4}(y) \quad (1a)$$

$$CH4_{up}(y) = CH4_{up}(y-1) \quad (1b)$$

$$CH4_{tot}(y) = CH4_{atm}(y) + CH4_{up}(y) \quad (1c)$$

where

[45] There exists another well-known representation of CO_2 accumulation equations, using a five-box model.

- $CH4_{atm}$, $CH4_{up}$, and EA_{CH4} are respectively: the atmospheric concentration, the natural concentration[46] (both expressed in Mt), and the anthropogenic emission of CH_4 (expressed in Mt/yr). EA_{CH4} is generated within the model, but $CH4_{up}$ is fully exogenous (see values for CH4-UP and CH4-ATM in Table A-2). All quantities are indexed by year.
- $d_{CH4} = 2.84$ (the density of *CH4*, expressed in *Mt/ppbv*) is then used to convert concentration in Mt into ppbv.
- $1-\Phi_{CH4}$ is the one-year retention rate of CH_4 in the atmosphere, see Table A-1.

c) N2O accumulation is also represented by a single-box model in which the atmospheric N2O concentration obeys the following equations:

$$N2O_{atm}(y) = (1-\Phi_{N2O}) \cdot N2O_{atm}(y-1) + EA_{N2O}(y) \qquad (1b)$$
$$N2O_{up}(y) = N2O_{up}(y-1) \qquad (2b)$$
$$N2O_{tot}(y) = N2O_{atm}(y) + N2O_{up}(y) \qquad (2c)$$

where

- $N2O_{atm}$, $N2O_{up}$, and EA_{N2O} , are respectively: the atmospheric concentration, the natural concentration (both expressed in Mt), and the anthropogenic emission of N_2O (expressed in Mt/yr). EA_{N2O} is generated within the model, but $N2O_{up}$ is fully exogenous (see values for N2O-UP and N2O-ATM in Table A-2). All quantities are indexed by year,
- d_{N2O} = 7.81 (the density of *N2O*, expressed in *Mt/ppbv*) is then used to convert concentration in Mt to ppbv units.
- $1-\Phi_{N2O}$ is the one-year retention rate of N_2O in the atmosphere, see table A-1.

Note: For both CH_4 and N_2O, the total atmospheric concentrations (UP+ATM) are used in the forcing expressions (see below) and are reported in the results.

2.2 Radiative forcing

We assume, as is routinely done in atmospheric science, that the atmospheric radiative forcing caused by the various gases are additive (IPCC, 2007). Thus:

$$\Delta F(y) = \Delta F_{CO2}(y) + \Delta F_{CH4}(y) + \Delta F_{N2O}(y) + EXOFOR(y) \qquad (3)$$

We now explain these four terms.

a) The relationship between CO2 accumulation and increased radiative forcing, $\Delta F_{CO2}(y)$, is derived from empirical measurements and climate models (IPCC 2007).

[46] Note that the subscripts *atm* and *up*, which for the CO2 equations referred to the atmosphere and upper reservoirs, have been reused for the CH4 and N2O equations to stand for anthropogenic and natural concentrations.

$$\Delta F_{CO2}(y) = \gamma \times \frac{\ln (M_{atm}(y)/M_0)}{\ln 2} \quad (4a)$$

where:

- M_0 (i.e.CO2ATM_PRE_IND) is the pre-industrial (circa 1750) reference atmospheric concentration of CO2 = 596.4 GtC
- γ is the radiative forcing sensitivity to atmospheric CO_2 concentration doubling = 3.7 W/m^2

b) The radiative forcing due to atmospheric CH4 is given by the following expression (IPCC, 2001)

$$\Delta F_{CH4}(y) = 0.036 \cdot \left(\sqrt{CH4_y} - \sqrt{CH4_0}\right) - \left[f(CH4_y, N2O_0) - f(CH4_0, N2O_0)\right] \quad (4b)$$

c) The radiative forcing due to atmospheric N2O is given by the following expression (IPCC, 2001)

$$\Delta F_{N2O}(y) = 0.12 \cdot \left(\sqrt{N2O_y} - \sqrt{N2O_0}\right) - \left[f(CH4_0, N2O_y) - f(CH4_0, N2O_0)\right] \quad (4c)$$

where:

$$f(x,y) = 0.47 \cdot \ln\left[1 + 2.01 \cdot 10^{-5} \cdot (xy)^{0.75} + 5.31 \cdot 10^{-15} \cdot x(xy)^{1.52}\right] \quad (4d)$$

Note that the *f(x,y)* function, which quantifies the cross-effects on forcing of the presence in the atmosphere of both gases (CH4 and N2O), is not quite symmetrical in the two gases. As usual, the 0 subscript indicates the pre-industrial times (1750)

d) *EXOFOR(y)* is the increase in total radiative forcing at period *t* relative to pre-industrial level due to GHGs that are not represented explicitly in the model. Units = W/m^2. In Nordhaus and Boyer (1999), only emissions of CO_2 were explicitly modeled, and therefore O(y) accounted for all other GHG's. In TIMES, N_2O and CH_4 are fully accounted for, but some other substances are not (e.g. CFC's, aerosols, ozone, etc.). Therefore, our values for *EXOFOR(y)* will differ from those in Nordhaus and Boyer. It is the modeler's responsibility to include in the calculation of *EXOFOR(y)* only the forcings from those gases and other causes that are not modeled. Table A-3 shows a possible trajectory for EXOFOR.

The parameterization of the three forcing equations (4a, 4b, 4c) is not controversial and relies on the results reported by Working Group I in the IPCC. IPCC (2001, Table 6.2, p.358) provides a value of 3.7 for γ, smaller than the one used by Nordhaus and Boyer (γ = 4.1). We have adopted this lower value of 3.7 W/m^2 as default in TIMES. Users are free to experiment with other values of the γ parameter. The same reference provides the entire expressions for all three forcing equations.

2.3 Linear approximations

In TIMES, each of the three forcing expressions is replaced by a linear approximation, in order to preserve linearity of the entire model. All three forcing expressions (4a, 4b, 4c) happen to be concave functions. Therefore, two linear approximations are obvious candidates. The first one is an approximation from below, consisting of the chord of the graph between two selected points. The second one has the same slope as the chord and is tangent to the graph, thus approximating the function from above. The final approximation is taken to be the arithmetic average of the two approximations. These linear expressions are easily derived once a range of interest is defined by the user.

As an example, we derive below the linear approximation for the CO2 forcing expression. The other approximations are obtained in a similar manner, and the parameters of the linear approximations are shown in the next section.

Linear approximation for the CO2 forcing expression:

First, an interval of interest for the concentration M must be selected by the user. The interval should be wide enough to accommodate the anticipated values of the concentrations, but not so wide as to make the approximation inaccurate. We denote the interval (M_1,M_2).

Next, the linear forcing equation is taken as the half sum of two linear expressions, which respectively underestimate and overestimate the exact forcing value. The underestimate consists of the chord of the logarithmic curve, whereas the overestimate consists of the tangent to the logarithmic curve that is parallel to the chord.

By denoting the pre-industrial concentration level as M_0, the general formulas for the two estimates are as follows:

Overestimate:
$$F_1(M) = \frac{\gamma}{\ln 2} \cdot \left[\ln\left(\frac{\gamma}{slope \cdot \ln(2) \cdot M_0}\right) - 1 \right] + slope \cdot M \qquad (5)$$

Underestimate:
$$F_2(M) = \gamma \cdot \ln(M_1 / M_0) / \ln 2 + slope \cdot (M - M_1) \qquad (6)$$

Final approximation:
$$F_3(M) = \frac{F_1(M) + F_2(M)}{2} \qquad (7)$$

where:
$$slope = \gamma \cdot \frac{\ln(M_2 / M_1) / \ln 2}{(M_2 - M_1)}$$

The linearized forcing expression implemented in TIMES is the average of the two linear estimates.

2.4 Temperature increase

In the TIMES Climate Module as in many other integrated models, climate change is represented by the global mean surface temperature. The idea behind the two-reservoir model is that a higher radiative forcing warms the atmospheric layer, which then quickly warms the upper ocean. In this model, the atmosphere and upper ocean form a single layer, which slowly warms the second layer consisting of the deep ocean.

$$\Delta T_{up}(y) = \Delta T_{up}(y\text{-}1) + \sigma_1\{ F(y) - \lambda\, \Delta T_{up}(y\text{-}1) - \sigma_2\, [\Delta T_{up}(y\text{-}1) - \Delta T_{low}(y\text{-}1)]\} \qquad (8)$$

$$\Delta T_{low}(y) = \Delta T_{low}(y\text{-}1) + \sigma_3\, [\Delta T_{up}(y\text{-}1) - \Delta T_{low}(y\text{-}1)] \qquad (9)$$

with

- ΔT_{up} = globally averaged surface temperature increase above pre-industrial level,
- ΔT_{low} = deep-ocean temperature increase above pre-industrial level,
- σ_1 = 1-year speed of adjustment parameter for atmospheric temperature (also known as the *lag* parameter),
- σ_2 = coefficient of heat loss from atmosphere to deep oceans,
- σ_3 = 1-year coefficient of heat gain by deep oceans,
- λ = feedback parameter (climatic retroaction). It is customary to write λ as $\lambda = \gamma / C_s$, C_s being the *climate sensitivity* parameter, defined as the change in equilibrium atmospheric temperature induced by a doubling of CO_2 concentration.

Remark: in contrast with most other parameters, the value of C_s is highly uncertain, with a possible range of values from 1°C to 10°C. This parameter is therefore a prime candidate for sensitivity analysis, or for treatment by probabilistic methods such as stochastic programming. In Table A-2, a best estimate value of 2.9 °C is shown, as per IPCC (2001, 2007).

In the next section we describe all the input parameters required to define the climate equations and those needed to define climate constraints. With few exceptions (such as the densities of the gases), all parameters are modifiable by the user, should the need arise. We also provide Table A-2 summarizing the default values of the parameters.

3 Switches and Parameters

3.1 Activating the Climate Module

The Climate Module (CLI) extension of TIMES can be activated and employed by using the Parameters and Switches described in this chapter.

Besides the basic input data parameters described in Table A-1, the user also has full control over the CLI component being activated by means of the $SET CLI YES switch. This switch is provided by the data handling system when the user indicates that the CLI option is to be included:

```
$SET CLI YES
```

3.2 Calibration

The calibration of the Climate Module to historical values is an important aspect of using the module. The mass balance and temperature equations can be calibrated for the first period by using three alternative calibration years B(1)–1, m(1)–1, and m(1). Whenever D(1)=1, the first two alternatives are equal. The default calibrating year is m(1)–1. The alternative calibration years can be activated by using one of the following two settings in the run-file:

```
$SET CM_CALIB B          ! Calibrate at the end of B(1)–1
$SET CM_CALIB M          ! Calibrate at the end of m(1)
```

3.3 Controlling the years considered beyond EOH

The Climate Equations will be calculated beyond EOH at each of the years for which either a user-defined emission target or a temperature or concentration bound is specified. The years considered thus span between the EOH and the last year for which a CM_MAXC is specified.

In addition, by default any Climate Equations beyond EOH will be calculated only at each year having a year value divisible by 20. This default year resolution can be changed by using the Climate Module constant **'BEOHMOD'**. However, note that the years available in the model extend by default to 2200 only, and therefore one may need to adjust the year-span e.g. to 2300 by using the following switch:

```
$SET EOTIME 2300
```

The **reporting years** for the climate variables are the same as the calculation years.

3.4 Input parameters

Like all other aspects of TIMES, the user defines the Climate Module components of the energy system by means of input parameters, which are described in this section. Table A-1 below describes the User Input Parameters that are associated with the Climate Module option.

Table A-1. Definition of Climate Module user input parameters.

Input Parameter (Indexes)	Units & Defaults	Description
CM_CONST (item)	Units: See on the right Defaults: See below	Various Climate Module constants, where item can be: PHI-UP-AT: carbon transfer coefficient UP→ATM PHI-AT-UP: carbon transfer coefficient ATM →UP PHI-LO-UP: carbon transfer coefficient LO→UP PHI-UP-LO: carbon transfer coefficient UP →LO GAMMA: radiative forcing sensitivity, in W/m2 CS: temperature sensitivity, in °C LAMBDA: $\lambda = \gamma / C_s$ SIGMA1: speed of adjustment, in W-yr/m^2/°C SIGMA2: thermal capacity ratio, in W/m^2/°C SIGMA3: transfer rate upper to deep ocean, in yr $^{-1}$ CO2-PREIND: pre-industrial atmosph. CO2, in GtC PHI-CH4: annual decay of atmospheric CH4, fraction PHI-N2O: annual decay of atmospheric N2O, fraction EXT-EOH: activates horizon extension, ≥0, year BEOHMOD: defines year interval for reporting, years
CM_HISTORY (y,cm_var)	Units: See on the right Defaults: See below	Historical calibration values at years *y*, for *cm_var*: CO2-ATM: atmospheric mass of CO2, in GtC CO2-UP: mass of CO2 in biosphere, in GtC CO2-LO: mass of CO2 in lower ocean, in GtC DELTA-ATM: atmospheric temperature change, in °C DELTA-LO: oceanic temperature change, in °C CH4-ATM: anthropogenic CH4 concentration, in Mt CH4-UP: natural CH4 concentration, in Mt N2O-ATM: anthropogenic N2O concentration, in Mt N2O-UP: natural N2O concentration, in Mt
CM_GHGMAP (r,c,cg)	Global units: CO2: GtC CH4: Mt N2O: Mt	Conversion factors from regional GHG commodities (c) to global emissions (*cg*) in the Climate Module, where cg= CO2-GtC: global CO2 emissions in GtC CH4-Mt: global CH4 emissions in Mt N2O-Mt: global N2O emissions in Mt
CM_EXOFORC (y)	Unit: W/m2	Radiative forcing from exogenous sources (from greenhouse gases not modelled) in year *y*.
CM_LINFOR (y,cm_var,lim)	Unit: For CO2: ppm CH4/N2O: W/m2/ppb Default:	Parameters for the linear forcing functions for cm_var: CO2-PPM: lower (LO) and upper (UP) end of the concentration range over which the forcing function for CO_2 is linearized (in ppm) CH4-PPB: multiplier (N) for the CH_4 concentration and

Input Parameter (Indexes)	Units & Defaults	Description
	none	constant term (FX) of the linear forcing function N2O-PPB: multiplier (N) for the N_2O concentration and constant term (FX) of the linear forcing function
CM_MAXC (y,cm_var)	Default: none	Maximum level of climate indicator *cm_var* in year *y*. CO2-GtC: CO2 emissions in GtC CH4-Mt: CH4 emissions in Mt N2O-Mt: N2O emissions in Mt CO2-ATM: atm. CO2 concentration / pre-industrial ratio CO2-PPM: atm. CO2 concentration in ppm CH4-PPB: atm. CH4 concentration in ppb N2O-PPB: atm. N2O concentration in ppb DELTA-ATM: atmospheric temperature change, in °C FORCING: total radiative forcing, in W/m2
CM_MAXCO2C (y)	Unit: GtC	Maximum level of CO2 concentration in GtC.

3.4.1 Mapping of regional emissions to global emissions

Conversion from regional emissions to global emissions must be done by using the CM_GHGMAP(r,c,cg) parameter, in adequate units. The labels for the global emissions **cg** are 'CO2-GtC', 'CH4-Mt' and 'N2O-Mt'. The parameter IRE_CCVT(r,c,r,cg) can alternatively be also used, if CM_GHGMAP is not available.

Assuming here that the total regional emissions are represented by the commodities TOTCO2, TOTCH4 and TOTN2O, and are measured in kt, as is the case in TIAM models for instance, the mapping and conversion would be the following:

```
CM_GHGMAP(R,'TOTCH4','CH4-MT') = 1E-3;
CM_GHGMAP(R,'TOTN2O','N2O-MT') = 1E-3;
CM_GHGMAP(R,'TOTCO2','CO2-GtC') = 2.727272 E-7
```

3.4.2 Deterministic input parameters for CO_2

- CM_CONST({PHI_AT_UP, PHI_UP_AT, PHI_UP_LO, PHI_LO_UP}) (also denoted φ_{atm-up}, φ_{up-atm}, etc, in the equations of section 2): annual CO_2 flow coefficients between the three reservoirs (AT=Atmosphere, UP=Upper ocean layer, LO=Deep ocean layer). These are time-independent coefficients. Units: none
- CM_HISTORY(y,{CO2-ATM, CO2-UP, CO2-LO}): Values at the end of the calibration year *y* of the masses of CO_2 in the atmosphere, the upper ocean layer, and the deep ocean layer, respectively. Note that these values are time- indexed so that the model generator can pick up the correct value according to the calibration year chosen by the user. Units: GtC, Mt(CH4), Mt(N2O).
- CM_CONST(CO2-PREIND): Pre-industrial atmospheric mass of CO2. Units = GtC

3.4.3 Parameters for the linear CO_2 forcing approximation

CM_LINFOR(datayear,item,lim): lower and upper limit for the concentration of CO_2 in atmosphere, used in the approximation of the radiative forcing equation for CO_2 (see section 2.2 above). *item* may be equal to CO2-ATM (in which case the limit is expressed as a ratio of concentration over pre-industrial concentration), or to CO2-PPM (in which case the limit is expressed in ppm of CO2-equivalent). The index *lim* is either equal to LO or to UP, depending on whether the lower or the upper limit of the range is being specified. For example, the following specifications may be used to select a range from 375 to 550 ppm for the approximation at year 2020:

- `CM_LINFOR('2020', 'CO2-PPM', 'LO') = 375;`
- `CM_LINFOR('2020', 'CO2-PPM', 'UP') = 550;`

Note that the values of LINFOR are systematically interpolated. The range can also be specified in a time-dependent manner taking into account the gradual increase in the expected range of possible concentration levels over time. That would further improve the accuracy of the linearization. For example, for 2005 the range could be specified to consist of only a single value, because the actual concentration in 2005 is well-known.

3.4.4 Parameters for modeling the concentrations and forcings of other greenhouse gases

Historical base year values of natural (UP) and anthropogenic (ATM) concentrations (in Mt), needed at for the base year of the model (default 2005):

CM_HISTORY('2005', 'CH4-UP') = 1988;
CM_HISTORY('2005', 'CH4-ATM') = 3067;
CM_HISTORY('2005', 'N2O-UP') = 2109;
CM_HISTORY('2005', 'N2O-ATM') = 390;

In the results the total concentrations (UP+ATM) are reported for both CH_4 and N_2O.

Annual exponential decay of concentrations (PHI-xxx = 1/Life):

CM_CONST('PHI-CH4') = 0.09158;
CM_CONST('PHI-N2O') = 0.008803;

Here Φ_{CH4}, Φ_{N2O}, are the one-year decay rates for methane and N_2O respectively

Parameters for the linear CH_4 and N_2O forcing approximations:
Note that for specifying the linear forcing functions for CH_4 and N_2O, the LO/UP bounds cannot be used, but the slope ('N') and constant ('FX') of the forcing functions must be directly defined by the user. Example:

CM_LINFOR('2010','CH4-PPB','N') = 0.000340;

CM_LINFOR('2010','CH4-PPB','FX') = –0.110;
CM_LINFOR('2010','N2O-PPB','N') = 0.00292;
CM_LINFOR('2010','N2O-PPB','FX') = –0.769;

Parameter for the exogenous radiative forcing from non-modeled gases in each year from initial year: CM_EXOFOR(y)

Units: Watts/m^2.

3.4.5 Parameters for the temperature equations

- CM_CONST(SIGMA1) (also denoted σ_1): speed of adjustment parameter for atmospheric temperature. $1/\sigma_1$ represents the thermal capacity of the atmospheric + upper ocean layer (W-yr/m^2/^oC). Note however that when SIGMA1 is assumed stochastic, its multiple values are specified via the generic S_CM_CONST parameter described below.
- CM_CONST(SIGMA2) (also denoted σ_2): ratio of the thermal capacity of the deep oceans to the transfer rate from shallow to deep ocean (W/m^2/^oC).
- CM_CONST(SIGMA3) (also denoted σ_3): $1/\sigma_3$ is the transfer rate (per year) from the upper level of the ocean to the deep ocean (yr^{-1}).
- CM_CONST(GAMMA) (also denoted γ): radiative forcing sensitivity to a doubling of the atmospheric CO_2 concentration. Units: Watts/m^2.
- CM_CONST(CS): C_s, the temperature sensitivity to a doubling of the CO_2 concentration (^oC).
- CM_CONST(LAMBDA) (also denoted λ): a feedback parameter, representing the equilibrium impact of CO_2 concentrations doubling on climate. $\lambda = \gamma / C_s$. Note however that when C_s is assumed stochastic, its multiple values are specified via the generic S_CM_CONST parameter described below. If all three of λ, γ and C_s are specified, the user-specified λ is overridden by the derived value γ / C_s.
- CM_HISTORY(y,{DELTA_ATM, DELTA_LOW}): values at the end of the calibration year *y* of the temperature changes (wrt to pre-industrial time) in atmosphere and deep layer, respectively. Units: ^oC

3.4.6 Upper bounds on climate variables

The following parameters are needed if constraints on some climate variables are desired. In TIMES, several climate upper bounds may be specified at any year. These upper bounds are specified via the single generic parameter CM_MAXC(datayear,item), where *datayear* is the year at which the bound applies, and *item* may be any of the following nine choices:

- CO2-ATM: for bounding the ***ratio*** of GHG concentration to the preindustrial concentration (where the pre-industrial concentration is defined by CO2-PREIND);
- CO2-PPM: for bounding the CO2 concentration expressed in ppm;
- CH4_PPB: for bounding the CH4 concentration expressed in ppbv;
- N2O-PPB: for bounding the N2O concentration expressed in ppbv;

- FORCING: for bounding the total atmospheric radiative forcing expressed in W/ m^2. (If this bound or the next one on temperature is used, the linearized forcing equation is used rather than the exact forcing equation);
- DELTA-ATM: for bounding the change in global atmospheric temperature over pre-industrial temperature, expressed in ^oC;
- CO2-GTC: for bounding the global CO2 emissions expressed in GtC;
- CH4-MT: for bounding the global CH4 emissions expressed in Mt;
- N2O-MT: for bounding the global N2O emissions expressed in Mt.

In addition, the user can also bound the CO_2 concentration expressed in GtC, by using CM_MAXCO2C.

3.4.7 Incorporating climate variables in UC constraints

When using the Climate Module extension, one can also refer to the climate variables in user constraints. The UC attribute for that purpose is the following:

UC_CLI(uc_n, side, reg, y, item)

This parameter can be used to define climate variable coefficients in any period-wise user constraints. The UC_GRPTYPE (to be used in UC_ATTR) for this parameter is 'CLI'. The ***item*** index can be any of the following climate variables:

- CO2-GTC - total global CO_2 emissions (or CO_2-eq. GHGs)
- CO2-ATM - CO_2 concentration in the atmosphere
- CO2-UP - CO_2 concentration in the biosphere/upper ocean
- CO2-LO - CO_2 concentration in the deep ocean layer
- FORCING - radiative forcing
- DELTA-ATM - atmospheric temperature
- DELTA-LO - deep oceanic temperature

The attribute can be used for defining custom relationships by each region, between any of the climate variables and e.g. process flows, activities or capacities, or total commodity flows. However, if used in a global constraint, one should normally define the UC_CLI attribute only for one region (e.g. GLB).

3.4.8 Random climate parameters (refer to documentation on stochastic TIMES)

If the stochastic programming version of TIMES is used, several climate parameters may be assumed random. These fall into two categories: the upper bounds on climate quantities discussed in the previous section, and the two climate coefficients, **Cs** and **SIGMA1.**

Regarding the random upper bounds, their multiple values are specified via the stochastic version of the **CM_MAX** parameter, namely **S_CM_MAX(datayear,item,stage,sow)**, where in addition to **datayear** and **item** already explained, **stage** refers to the stage of the event tree and **sow** refers to the state-of-the-world. Note that this single generic parameter will be specified as many times as there are

stages and **sow**'s in the stochastic event tree. If this parameter is specified, the corresponding values of the deterministic parameter **CM_MAX** are superseded.

Regarding the two random coefficients, their multiple values are then declared via the single generic parameter **S_CM_CONST(item,stage,sow)**, where **item** may be equal to **CS** or to **SIGMA1**, **stage** is the stage number, and **sow** is the state-of-the-world. Note that this single generic parameter will be specified as many times as there are stages and sow's in the stochastic event tree. If this parameter is specified, the corresponding values of the deterministic parameter (**LAMBDA** and/or **SIGMA1**) are superseded.

The reader is referred to Chapter 8 of Part I and the documentation of the stochastic programming version of TIMES for the precise meaning of the **stage** and **sow** concepts.

Remark: in addition to the possible values of the random parameters, the user must specify the probabilities attached to each ***sow***. This is also explained in the documentation on stochastic TIMES.

3.4.9 Parameters for extending the Climate Module equations beyond EOH

The main purpose of extending the climate equations beyond EOH is to be able to set climate targets beyond EOH. This is particularly useful for DeltaT targets, because there is a considerable time lag between the decline of emissions and the peak of DeltaT.

The extended climate equations must be explicitly activated by the user. The activation can be done by specifying any non-negative value for the new Climate Module constant **CM_CONST**(**'EXT-EOH')**. Different values of the constant will have the following meaning:

Value	Meaning
-1 (default)	The feature is deactivated.
0	In this case **'EXT-EOH'** will be automatically adjusted to E(*M*), where *M* is the last model year ***m*** for which the end-year **E(*m*)** is specified. The adjusted parameter will then have the same meaning as in the case EXT-EOH > 0 below.
>0	The emissions at EOH will remain constant at the endogenous value in EOH=E(T) (where T=last **milestone year**) until the year MAX(**EXT-EOH**, EOH), and then develop linearly from that value to the first user-defined emission value in a subsequent year.

The setting **EXT-EOH**=**0** may be useful for ensuring that any user-defined target values for the emissions will only be taken into account beyond the last ***model year***, even in model runs where a truncated model horizon is used. In such case, when **EXT-EOH**=**0** is used, the emissions are assumed to remain constant between the truncated EOH and the end of the full model horizon.

A positive value **EXT-EOH**=**y ≤ EOH** means that a linear development of emissions towards the first user-defined value is requested to start immediately at the EOH, regardless of the model horizon being truncated or not. Finally, a positive value **EXT-EOH**=**y > EOH** can be useful if the user wishes the emissions to remain constant at the EOH value until a predefined year y > EOH, before turning into the linear development towards the first user-defined value.

Warning: If **0 < EXT-EOH < E(M) = $\mathbf{MAX_m(E(m))}$**, any user-defined global emission bounds for CO2-GTC, CH4-MT or N2O-MT, which may be inadvertently specified at years between **MAX(EXT-EOH, EOH)** and **E(M)**, will also be taken into account as target values for the emission trajectories.

The global greenhouse gas emissions that can be considered by the extended climate equations are the three main input emissions to the Climate Module:

- CO2-GTC Global CO2 emissions, expressed in GtC
- CH4-MT Global CH4 emissions, expressed in Mt
- N2O-MT Global N2O emissions, expressed in Mt

The user can specify target emission values for these emissions at any year(s) beyond EOH. For simplicity, the target emission values are specified by using the **CM_MAXC** parameter, which is normally used for specifying upper bounds for the global emissions, as well as for the temperature and concentrations.

Starting from the year ***B***= MAX(EOH,EXT-EOH), the emissions will be assumed to develop linearly from the value at EOH to the first user-specified value beyond ***B***. If no target values are specified, the emissions will be assumed to remain constant at the EOH value. If several successive values are specified, the emissions will develop linearly also between the successive target values.

Bounds on the global atmospheric temperature, forcing or GHG concentrations can be specified at any years beyond the EOH, in the normal way. In addition, exogenous forcing can be specified and is interpolated beyond EOH.

The Climate Equations will be calculated beyond EOH at each of the years for which either a user-defined emission target or a temperature or concentration bound is specified. The years considered thus span between the EOH and the last year for which a CM_MAXC is specified. However, as described above, any emission bounds between EOH and MAX(EOH,EXT-EOH) will be ignored.

In addition, by default the Climate Equations will be calculated also at each year having a year value divisible by 20. This default year resolution can be changed by using the new Climate Module constant **'BEOHMOD'**. Accordingly, if the user wishes the Climate Equations to be calculated at 10 years' intervals (in addition to the CM_MAXC years) she can specify the following parameter:

```
PARAMETER CM_CONST / BEOHMOD 10 /;
```

The **reporting years** for the climate variables are the same as the calculation years.

3.5 Internal parameters

- *$CM\_PPM_{cm\_var}$*: The densities of the greenhouse gases are hard coded in TIMES (via the internal parameter), with the following values:
  density of CH_4 : 2.84 Mt / ppbv
  density of N_2O : 7.81 Mt / ppbv
  density of CO_2 : 2.13 Gt / ppm.

- $CM\_PHI_{cm\_var,t,i,j}$: The transition matrix for climate indicator cm_var between reservoirs i and j and successive years t–1 and t;
- $CM\_AA_{cm\_var,t,i,j}$: The transition matrix for climate indicator cm_var between reservoirs i and j and between the milestone years of periods t–1 and t;
- $CM\_BB_{cm\_var,t,i,j}$: The transition matrix for climate indicator cm_var from emissions in period t to reservoir contents in the same period;
- $CM\_CC_{cm\_var,t,i,j}$: The transition matrix for climate indicator cm_var from emissions in period t–1 to reservoir contents in the period t.

3.6 Reporting parameters

There are two reporting parameters, CM_RESULT and CM_MAXC_M, which contain the results on the levels of the climate variables (or reporting quantities) and the dual values of the constraints defined by using CM_MAXC.

CM_RESULT is indexed by year *y* and result type {e.g. CO2-ATM, CO2-PPM, FORCING, DELTA-ATM, DELTA_LO}. The values represent the quantities at the end of year y. The reporting years y include the milestone years plus any years beyond m(T) that either have some CM_MAXC bound defined or are modulo(BEOHMOD).

- CO2-GtC(y): the total global CO2 emissions at the end of year **y**.
- CO2-ATM(y): the value of the atmospheric mass of CO2-equivalent at the end of year **y**, obtained directly from the variable **VAR_CLIBOX('CO2-ATM',y)**.
- CO2-PPM(y): the value of the atmospheric concentration of CO2-equivalent at the end of year **y**.
- FORCING(y): forcing value at end of year y, calculated using the linearized forcing functions as defined by the user.
- FORC+TOT(y): exact forcing value at end of year y, calculated using the logarithmic forcing equation defined in section 2.2 and the CO2-ATM(y) value.
- DELTA_ATM(y): exact atmospheric temperature value at end of year y, calculated using the forcing FORC+TOT(y).
- DELTA_LOW(y): exact lower ocean temperature value at end of year y, calculated using the forcing FORC+TOT(y).

CM_MAXC_M is indexed by year *y* and constraint type. The values are reported for each of the EQ_CLITOT and EQ_CLIMAX equations. The values represent directly the dual values of these constraints at year y.

3.7 Default values of the climate parameters

Table A-2 shows the default values of all parameters of the Climate Module except exogenous forcing. All defaults may be modified by the user.

- CS and SIGMA1 may be assumed random, in which case the default values are not used. The user must specify their values explicitly using the appropriate parameter names described earlier.
- The parameters highlighted blue are upper bounds on five climate variables (in this example, they are set high enough to be inoperative).
- The three parameters highlighted pink concern the extension of emissions beyond EOH, as described in the separate note on this subject.

Table A-3 shows an example of specification of the EXOFORCING time series.

Table A-2. Parameters of the climatic module (default values)

Attribute	Lim	DataYear	Item	Default value
CM_HISTORY		2005	CO2-ATM	807.27
CM_HISTORY		2005	CO2-UP	793
CM_HISTORY		2005	CO2-LO	19217
CM_HISTORY		2005	DELTA-ATM	0.76
CM_HISTORY		2005	DELTA-LO	0.06
CM_HISTORY		2005	CH4-UP	1988
CM_HISTORY		2005	CH4-ATM	3067
CM_HISTORY		2005	N2O-UP	2109
CM_HISTORY		2005	N2O-ATM	390
CM_CONST			GAMMA	3.71
CM_CONST			PHI-UP-AT	0.0453
CM_CONST			PHI-AT-UP	0.0495
CM_CONST			PHI-LO-UP	0.00053
CM_CONST			PHI-UP-LO	0.0146
CM_CONST			LAMBDA	1.41
CM_CONST			CS	2.9
CM_CONST			SIGMA1	0.024
CM_CONST			SIGMA2	0.44
CM_CONST			SIGMA3	0.002
CM_CONST			CO2-PREIND	596.4
CM_CONST			PHI-CH4	0.09158
CM_CONST			PHI-N2O	0.008803
CM_LINFOR	LO	2005	CO2-PPM	375
CM_LINFOR	UP	2005	CO2-PPM	550
CM_LINFOR	N	2005	CH4-PPB	0.00034
CM_LINFOR	FX	2005	CH4-PPB	-0.11000

CM_LINFOR	N	2005	N2O-PPB	0.00292
CM_LINFOR	FX	2005	N2O-PPB	-0.76900
CM_MAXC		2005	CO2-PPM	500
CM_MAXC		2005	CO2-ATM	1000
CM_MAXC		2005	FORCING	10
CM_MAXC		2005	DELTA-ATM	10
CM_MAXC		2005	CO2-GTC	50
CM_CONST			EXT-EOH	2150
CM_CONST			BEOHMOD	20
CM_MAXC		2200	CO2-GTC	0

Table A-3. Example of EXOFORCING (from TIAM-WORLD, 2010 version)

Attribute	DataYear	Value
CM_EXOFORC	2005	-0.25376
CM_EXOFORC	2010	-0.20475
CM_EXOFORC	2015	-0.16055
CM_EXOFORC	2020	-0.11689
CM_EXOFORC	2025	-0.10104
CM_EXOFORC	2030	-0.0774
CM_EXOFORC	2035	-0.06398
CM_EXOFORC	2040	-0.03787
CM_EXOFORC	2045	-0.0354
CM_EXOFORC	2050	-0.04528
CM_EXOFORC	2055	-0.06434
CM_EXOFORC	2060	-0.08634
CM_EXOFORC	2065	-0.09485
CM_EXOFORC	2070	-0.09632
CM_EXOFORC	2075	-0.09254
CM_EXOFORC	2080	-0.08929
CM_EXOFORC	2085	-0.08868
CM_EXOFORC	2090	-0.08273
CM_EXOFORC	2095	-0.0796
CM_EXOFORC	2100	-0.07447

4 Variables

The variables that are used in the Climate Module in TIMES are presented in **Table A-4** below. The climate indicators represented in the Climate Module are grouped according to the following internal sets, which are referred to in the GAMS formulation, presented in Section 5:

- **cm_var**: the set of all climate indicators
- **cm_tkind**: aggregate total indicators (CO2-GtC, CH4-Mt, N2O-Mt, FORCING)
- **cm_emis**: emission indicators (CO2-GtC, CH4-Mt, N2O-Mt)
- **cm_boxmap**$_{\textbf{tkind,cm\_var,cm\_box}}$: mapping between aggregate indicators tkind, reservoir indicators cm_var, and corresponding box labels (ATM/UP/LO);
- **cm_atmap**$_{\textbf{tkind,cm\_var}}$: mapping between aggregate indicators *tkind* and the corresponding boundable atmospheric indicators (CO2-PPM / CH4-PPM / N2O-PPM / DELTA_ATM);
- **cm_atbox** $_{\textbf{tkind,cm\_box}}$: mapping between mapping between aggregate emission indicators *tkind* and the corresponding reservoirs that comprise the atmospheric concentration part; contains the pairs {(CO2-GtC,ATM), (CH4-Mt,ATM),(CH4-Mt,UP),(N2O-Mt,ATM),(N2O-Mt,UP) }

Table A-4. Model variables specific to the Climate Module.

Variable (Indexes)	Variable Description
VAR_CLITOT (cm_var,y)	Represents the total amount of climate indicator *cm_var* in year y, where *cm_var* is one of {CO2-GtC, CH4-Mt, N2O-Mt, FORCING}.
VAR_CLIBOX (cm_var,y)	Represents the amount of reservoir indicator *cm_var* in a single reservoir/box in year y, where *cm_var* is one of {CO2-ATM, CO2-UP, CO2-LO, CH4-ATM, CH4-UP, N2O-ATM, N2O-UP, DELTA-ATM, DELTA-LO}.

4.1 VAR_CLITOT(cm_var,y)

Description: The total amount of aggregate climate indicator in year y.

Purpose and Occurrence: This variable tracks the total amount of an aggregate climate indicator by period. This variable is generated for each main emission type of the Climate Module as well as for the total forcing from all greenhouse gas concentrations.

Units: GtC (for CO2 emissions), Mt (for CH4 and N2O emissions), or W/m2 (for total radiative forcing).

Bounds: This variable can be directly bounded with the CM_MAXC attribute.

4.2 VAR_CLIBOX(cm_var,y)

Description: The amount of climate indicator in a reservoir.

Purpose and Occurrence: This variable tracks the amount of reservoir-specific climate indicator by period. This variable is generated for each of the reservoirs for each of the aggregate indicators: ATM/UP/LO for CO2 emissions, ATM/UP for CH4 and N2O emissions, and ATM/LO for FORCING (connected to the temperature reservoirs).

Units: GtC (for CO2 emissions), Mt (for CH4 and N2O emissions), or °C (for temperature reservoirs).

Bounds: Only the total atmospheric amounts can be bounded with the CM_MAXC attribute (CO2-ATM, CO2-PPM, CH4-PPB, N2O-PPB, DELTA-ATM).

5 Equations

There are three blocks of definitional equations: the first block of equations calculates the global emissions of GHG (either all in CO_2 eq., or separately for CO_2, CH_4 and N_2O) as well as the total (linearized) radiative forcing, the next block calculates the concentrations of the greenhouse gases in the reservoirs, and the third block calculates the atmospheric temperature and lower ocean temperature at period t.

In addition, there is a generic block of equations expressing the upper bounding of the five climate quantities discussed in subsection 3.4.6. This generic equation is generated as many times as an upper bound on any climate variable is specified by the user, and is not generated if no upper bound is specified.

We now give the formulations of these constraints.

> **Reminder**: the Climate Module formulation is activated at run time from the data handling system, which in turn set the $SET CLI YES switch.

General notation:

- *D(t):* duration of period *t*, *t=1 to T*
- *B(t):* first year in period *t*, *t=1 to T*
- *m(t):* milestone year of period t (approximate middle year of period, defined as $m(t) = B(t) + \lfloor (D(t) - 1)/2 \rfloor$
- *y:* designates a year, while *t* designates a period (ranging from 1 to T)
- *Y:* designates the calibration year, which can be chosen by the user to be either *B*(1)–1, *m*(1)–1, or *m*(1), see section 3.2 above.

Table A-5. Climate Module specific constraints (all in the GAMS file equ_ext.cli).

Constraints (Indexes)	**Constraint Description**
EQ_CLITOT (cm_var,t)	Defines the amount of global greenhouse gas emissions in each period; defines the amount of total radiative forcing from the greenhouse gas concentrations in each period *t*.
EQ_CLICONC (cm_var, cm_box,t)	Defines the mass of each greenhouse gas *cm_var* in each reservoir *cm_box* at the end of the milestoneyr **m(t)** of period *t*.
EQ_CLITEMP (cm_box,t)	Defines the temperature increase in the each reservoir *cm_box* (the lower atmosphere and the lower ocean layer) over its pre-industrial temperature measured at the end of milestoneyr **m(t)** of period *t*.
EQ_CLIMAX (y,cm_var)	Imposes an upper bound on any or all of the climate variables *cm_var* (*CO2-GTC, CH4-MT, N2O-MT, CO2-ATM, CO2-PPM, CH4-PPB, N2O-PPB, FORCING, DELTA-ATM)*, at any desired year *y*, according to the user-defined input parameter CM_MAXC.

5.1 EQ_CLITOT(cm_var,t)

Description: Defines the total amount of aggregate climate indicator in period t.

Purpose: This constraint defines the amount of global greenhouse gas emissions in each period and the amount of total radiative forcing from the greenhouse gas concentrations in each period *t*.
This equation is generated in each time period for all indicators considered.

Units: Global emission units (GtC, Mt) or forcing units (W/m2)

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:
Primal: The level of this constraint must be zero in a feasible solution.
Dual variable: The dual variables represent the marginal prices of the global emissions / forcing (when undiscounted).

Remarks:
- For CO2, the linear forcing function parameters $CM\_LINFOR_{t,cm\_emis,'FX'}$ and $CM\_LINFOR_{t,cm\_emis,'N'}$ are automatically calculated by the model generator from any user-defined $CM\_LINFOR_{t,cm\_emis,'LO'}$ and $CM\_LINFOR_{t,cm\_emis,'UP'}$.

Equation:

$$EQ\_CLITOT_{cm\_tkind,t} \ \forall \left[\left(t \in \mathbf{milestonyr}\right)\right]$$

$$\sum_{\substack{cm\_tkind \in \mathbf{cm\_emis} \\ (r,c,s) \in \mathbf{rtcs\_varc}_{r,c,t,s}}} VAR\_COMNET_{r,t,c,s} \times CM\_GHGMAP_{r,c,cm\_tkind}$$

$$\sum_{\mathbf{cm\_emis}_{\mathbf{cm\_tkind}}} \left(\begin{array}{l} CM\_LINFOR_{t,cm\_emis,'N'} \times \\ \left(\sum_{\substack{\mathbf{cm\_atbox}_{\mathbf{cm\_emis,cm\_box}} \\ \mathbf{cm\_boxmap}_{\mathbf{cm\_emis,cm\_var,cm\_box}}}} VAR\_CLIBOX_{cm\_var} \right) \\ CM\_LINFOR_{t,cm\_emis,'FX'} \end{array} \right) +$$

$$+ CM\_EXOFORC_t$$

$$\{=\}$$

$$VAR\_CLITOT_{cm\_tkind,t}$$

5.2 EQ_CLICONC(cm_var,cm_box,t)

Description: Defines the reservoir-specific amounts of concentration indicator in each period.

Purpose: Defines the dynamic relationship between emissions and the concentration in the reservoirs modelled for each greenhouse gas, such that the amount of concentration in reservoir *i* and period *t* may depend on the amounts of concentrations in any reservoir *k* in period *t–1*, and on the emissions in period *t*.

Units: Global emission units (GtC, Mt).

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be zero in a feasible solution.

Dual variable: The dual variable of this constraint in the solution is of little interest.

Remarks:

- See expressions for the transfer matrices on next page.
- The equations beyond the last milestone year m(T) are similar, but omitted here.

Equation:

$$EQ\_CLICONC_{cm\_emis,cm\_box,t} \quad \forall\left[\left(t \in \mathbf{milestonyr}\right)\right]$$

$$\sum_{\mathbf{cm\_boxmap}_{cm\_emis,cm\_var,cm\_box2}} VAR\_CLIBOX_{cm\_var,t-1} \times CM\_AA_{cm\_emis,t,cm\_box,cm\_box2} +$$

$$CM\_BB_{cm\_emis,t,cm\_box} \times VAR\_CLITOT_{cm\_emis,t} +$$

$$CM\_CC_{cm\_emis,t,cm\_box} \times VAR\_CLITOT_{cm\_emis,t-1} +$$

$$\sum_{\substack{\mathbf{miyr\_1}_t \\ \mathbf{cm\_boxmap}_{cm\_emis,cm\_var,cm\_box2}}} CM\_CONST_{cm\_var} \times CM\_AA_{cm\_emis,t,cm\_box,cm\_box2}$$

$$\{=\}$$

$$\sum_{\mathbf{cm\_boxmap}_{cm\_emis,cm\_var,cm\_box}} VAR\_CLIBOX_{cm\_var,t}$$

$$CM\_AA_{cm\_emis,t,i,j} = \left\{A_{ij}(t)\right\} = PHI^{n(t)} \ \ (PHI^0 = I), \text{ where}$$

$$PHI \text{ is the } 3\times 3 \text{ matrix : } \begin{bmatrix} (1-PHI\_AT\_UP) & PHI\_UP\_AT & 0 \\ PHI\_AT\_UP & (1-PHI\_UP\_AT-PHI\_UP\_LO) & PHI\_LO\_UP \\ 0 & PHI\_UP\_LO & (1-PHI\_LO\_UP) \end{bmatrix}$$

$$CM\_BB_{cm\_emis,t,i} = \left\{BB_{i1}(t)\right\} \text{ is the first column of the matrix :}$$

$$BB(t) = \sum_{i=0}^{p(t)-1} PHI^i \qquad if \ \ p(t) \geq 1$$

$$BB(t) = 0 \qquad if \ \ p(t) = 0$$

$$CM\_CC_{cm\_emis,t,i} = \left\{CC_{i1}(t)\right\} \text{ is the first column of the matrix :}$$

$$CC(t) = \sum_{i=p(t)}^{n(t)-1} PHI^i \qquad if \ \ n(t) \geq p(t)+1$$

$$CC(t) = 0 \qquad if \ \ n(t) = p(t)$$

$$p(t) = \left\lfloor \frac{D(t)+1}{2} \right\rfloor, \qquad n(t) = m(t) - m(t-1) \qquad if \ \ t \neq 1,$$

$$p(t) = m(t) - \Upsilon, \qquad n(t) = p(t) \qquad if \ \ t = 1$$

$D(t)$ is the number of years in period t, *and* $m(t)$ is the middle year of period t defined as

$$m(t) = B(t) + \left\lfloor \frac{D(t)-1}{2} \right\rfloor$$

$\lfloor x \rfloor$ denotes the largest integer smaller than or equal to x

5.3 EQ_CLITEMP(cm_var,cm_box,t)

Description: Defines the reservoir-specific amounts of temperature indicator in each period.

Purpose: Defines the dynamic relationship between forcing and the temperature increase in the reservoirs modelled, such that the amount of temperatures increase in reservoir *i* and period *t* may depend on the amounts of temperature increase in any reservoir *k* in period *t–1*, and on the radiative forcing in period *t*.

Units: Global temperature units (°C).

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:
Primal: The level of this constraint must be zero in a feasible solution.
Dual variable: The dual variable of this constraint in the solution is of little interest.

Remarks:
- See expressions for the transfer matrices on next page.
- The equations for years beyond m(T) are similar, but omitted here.

Equation:

$$EQ\_CLITEMP_{cm\_box,t} \quad \forall\left[\left(t \in \mathbf{milestonyr}\right)\right]$$

$$\sum_{\mathbf{cm\_boxmap}_{'FORCING',cm\_var,cm\_box2}} VAR\_CLIBOX_{cm\_var,t-1} \times CM\_AA_{'FORCING',t,cm\_box,cm\_box2} +$$

$$CM\_BB_{'FORCING',t,cm\_box} \times VAR\_CLITOT_{'FORCING',t} +$$

$$CM\_CC_{'FORCING',t,cm\_box} \times VAR\_CLITOT_{'FORCING',t-1} +$$

$$\sum_{\substack{\mathbf{miyr\_1}_t \\ \mathbf{cm\_boxmap}_{'FORCING',cm\_var,cm\_box2}}} CM\_CONST_{cm\_var} \times CM\_AA_{'FORCING',t,cm\_box,cm\_box2}$$

$$\{=\}$$

$$\sum_{\mathbf{cm\_boxmap}_{'FORCING',cm\_var,cm\_box}} VAR\_CLIBOX_{cm\_var,t}$$

$CM\_AA_{'FORCING',t,i,j} = \{A_{ij}(t)\} \quad = PHI^{n(t)} \ \ (PHI^0 = I)$, where

PHI is the 3×3 matrix : $\begin{bmatrix} (1-SIGMA1\times(LAMBDA+SIGMA2) & SIGMA1\times SIGMA2 & 0 \\ SIGMA3 & (1-SIGMA3) & 0 \\ 0 & 0 & 0 \end{bmatrix}$

$CM\_BB_{'FORCING',t,i} = \{BB_{i1}(t)\}$ is the first column of the matrix :

$$BB(t) = SIGMA1\times \sum_{i=0}^{n(t)-1} \frac{n(t)-i}{n(t)} \times PHI^i$$

$CM\_CC_{'FORCING',t,i} = \{CC_{i1}(t)\}$ is the first column of the matrix :

$$CC(t) = SIGMA1\times \sum_{i=0}^{n(t)-1} \frac{i}{n(t)} \times PHI^i$$

$$n(t) \ = \ m(t) - m(t-1) \qquad\qquad if \ \ t \neq 1,$$

$$n(t) \ = m(t) - Y \qquad\qquad\qquad if \ \ t = 1$$

$D(t)$ is the number of years in period t, and $m(t)$ is the middle year of period t defined as

$$m(t) = B(t) + \left\lfloor \frac{D(t)-1}{2} \right\rfloor$$

$\lfloor x \rfloor$ denotes the largest integer smaller than or equal to x

5.4 EQ_CLIMAX(y,cm_var)

Description: Constraint that sets an upper bound on the climate indicator in a give year.

Purpose: To set an upper bound for a climate indicator variable in any desired year y. The variables that can be bounded are the total global emissions and the total radiative forcing (VAR_CLITOT), the atmospheric concentrations of greenhouse gases (sum of VAR_CLIBOX variables), and the increase in atmospheric temperature (VAR_CLIBOX). The bounds can be specified by using the CM_MAXC_{y,cm_var} attribute.

Units: Units of the variable(s) bounded.

Type: *Binding.* The equation is a less than or equal to inequality (≤) constraint.

Interpretation of the results:

Primal: The level of this constraint must be less than or equal to zero in a feasible solution.

Dual variable: The dual variable of this constraint in the solution may be used to derive the marginal price of the climate indicator constrained (when undiscounted; global dual values are, ex officio, reported without undiscounting, as no well-defined "global discount factors" exist, only regional ones).

Remarks:

- The *CM_MAXC* bounds defined on CO2-ATM are automatically converted into equivalent bounds on CO2-PPM.
- The coefficients α_y and β_y in the equations are such that $y = \alpha_y\,(\mathrm{m(t)}-y) + \beta_y\,(y-\mathrm{m(t-1)})$, for all y in the range $\mathrm{m(t-1)} < y \le \mathrm{m(t)}$.

Equation:

$$EQ\_CLIMAX_{y,cm\_var} \quad \forall\left[\left\{(y,cm\_\mathrm{var}) \mid CM\_MAXC_{y,cm\_var}\right\}\right]$$

Case A. For total emissions, up to m(T)

$$\alpha_y \times VAR\_CLITOT_{cm\_emis,t-1} + \beta_y \times VAR\_CLITOT_{cm\_emis,t}$$
$$\le CM\_MAXC_{y,cm\_emis}$$

Case B. For atmospheric GHG concentrations, up to m(T)

$$\sum_{\substack{\mathbf{cm\_atbox}_{\mathbf{cm\_emis,cm\_box}} \\ \mathbf{cm\_boxmap}_{\mathbf{cm\_emis,cm\_var,cm\_box}}}} \alpha_y \times VAR\_CLIBOX_{cm\_var,t-1} + \beta_y \times VAR\_CLIBOX_{cm\_var,t}$$
$$\leq CM\_MAXC_{y,cm\_\mathrm{var}}$$

Case C. For total radiative forcing, up to m(T)

$$\alpha_y \times VAR\_CLITOT_{'FORCING',t-1} + \beta_y \times VAR\_CLITOT_{'FORCING',t}$$
$$\leq CM\_MAXC_{y,'FORCING'}$$

Case D. For increase in global atmospheric temperature, up to m(T):

$$\sum_{\mathbf{cm\_boxmap}_{\mathbf{'FORCING',cm\_var,'ATM'}}} \alpha_y \times VAR\_CLIBOX_{cm\_var,t-1} + \beta_y \times VAR\_CLIBOX_{cm\_var,t}$$
$$\leq CM\_MAXC_{y,cm\_\mathrm{var}}$$

Case E. For atmospheric GHG concentrations, beyond m(T):

$$\sum_{\substack{\mathbf{cm\_atbox}_{\mathbf{cm\_emis,cm\_box}} \\ \mathbf{cm\_boxmap}_{\mathbf{cm\_emis,cm\_var,cm\_box}}}} VAR\_CLIBOX_{cm\_var,y} \quad \leq \quad CM\_MAXC_{y,cm\_\mathrm{var}}$$

Case F. For total radiative forcing, beyond m(T):

$$VAR\_CLITOT_{'FORCING',y} \quad \leq \quad CM\_MAXC_{y,'FORCING'}$$

Case G. For increase in global atmospheric temperature, beyond m(T):

$$\sum_{\mathbf{cm\_boxmap}_{\mathbf{'FORCING',cm\_var,'ATM'}}} VAR\_CLIBOX_{cm\_var,y} \quad \leq \quad CM\_MAXC_{y,cm\_\mathrm{var}}$$

References

Drouet L., Edwards N.R. and A. Haurie (2004). “Coupling Climate and Economic Models in a Cost-Benefit Framework: A Convex Optimization Approach”. Submitted to *Environmental Modeling and Assessment.*

IPCC. 1996. *Climatic change 1995: The Science of Climate Change.* Intergovernmental Panel on Climate Change, Second Assessment Report, Working Group I. Cambridge: Cambridge University Press. p.572.

IPCC. 2001. *Climatic change 2001: The Scientific Basis.* Intergovernmental Panel on Climate Change, Third Assessment Report, Working Group I. Cambridge: Cambridge University Press. p.944.

Nordhaus, W. D. and J. Boyer. 1999. *Roll the DICE Again: Economic Models of Global Warming*. Yale University, manuscript edition.

Wigley, T.M.L., Solomon, M. and S.C.B. Raper. 1994. *Model for the Assessment of Greenhouse-Gas Induced Climate Change*. Version 1.2. Climate Research Unit, University of East Anglia, UK.

Appendix B Damage Cost Functions

1 Introduction

This Appendix contains the documentation on the Damage Cost Function extensions for the TIMES model. The chapter contains 6 sections: section 2 contains the mathematical formulation, section 3 describes the parameters for the Damage Cost Functions, and section 4 gives two examples. Finally, section 5 describes the variables and section 6 describes the equations.

The Damage Cost Function option of TIMES is intended for modelers who wish to evaluate the environmental externalities caused by an energy system. For instance, emissions of toxic or environmentally harmful pollutants from the energy system create social costs linked to impacts of the pollution on human health and the environment. In another example, in global studies of GHG emissions, it may be of interest to evaluate the impact of GHG emissions on concentrations and ultimately on damages created by climate change induced by increased concentration of GHGs.

Until recently, in most studies involving bottom-up models, emission externalities have been modeled in one of two ways: either by introducing an emission tax, or by imposing emission caps. In the first case, the tax is (ideally) supposed to represent the external cost created by one unit of emission. However, using a tax assumes that the cost is a linear function of emissions. In the second approach, it is assumed that such a cost is unknown but that exogenous studies (or regulations, treaties, etc.) have defined a level of acceptable emissions that should not be exceeded. However, using this approach is akin to making the implicit assumption that emissions in excess of the cap have an infinite external cost. Both of these approaches have merit and have been successfully applied to many energy system model studies.

It is however possible to extend these two approaches by introducing an option to better model the cost of damages created by emissions. The damage function option discussed in this document extends the concept of an emission tax by modeling more accurately the assumed cost of damages due to emissions of a pollutant.

2 Mathematical formulation

We now describe the mathematical formulation used for the damage cost functions. With respect to optimization, two distinct approaches to account for damage costs can be distinguished:

1. Environmental damages are computed ex-post, without feedback into the optimization process, and
2. Environmental damages are part of the objective function and therefore taken into account in the optimization process.

In both approaches, a number of assumptions are made:

- Emissions in each region may be assumed to cause damage only in the same region or, due to trans-boundary pollution, also in other regions; however, all damage costs are allocated to the polluters in the source region, in accordance with the Polluter Pays Principle, or Extended Polluter Responsibility;
- Damages in a given time period are linked to emissions in that same period only (damages are not delayed, nor are they cumulative); and
- Damages due to several pollutants are the sum of damages due to each pollutant (no cross impacts).

In a given time period, and for a given pollutant, the damage cost is modeled as follows:

$$DAM(EM) = \alpha \cdot EM^{\beta+1} \tag{1}$$

where:

- EM is the emission in the current period;
- DAM is the damage cost in the current period;
- $\beta \geq 0$ is the elasticity of marginal damage cost to amount of emissions; and
- $\alpha > 0$ is a calibrating parameter, which may be obtained from dose-response studies that allow the computation of the marginal damage cost per unit of emission at some reference level of emissions.

If we denote the marginal cost at the reference level MC_0, the following holds:

$$MC_0 = \alpha \cdot (\beta + 1) \cdot EM_0^{\beta} \tag{2}$$

where EM_0 is the reference amount of emissions. Therefore expression (1) may be re-written as:

$$DAM(EM) = MC_0 \cdot \frac{EM^{\beta+1}}{(\beta + 1) \cdot EM_0^{\beta}} \tag{3}$$

The marginal damage cost is therefore given by the following expression:

$$MC(EM) = MC_0 \cdot \frac{EM^{\beta}}{EM_0^{\beta}} \tag{4}$$

The approach to damage costs described in this section applies more particularly to local pollutants. Extension to global emissions such GHG emissions requires the use of a global TIMES model and a reinterpretation of the equations discussed above.

The modeling of damage costs via equation (3) introduces a non-linear term in the objective function if the $\boldsymbol{\beta}$ parameter is strictly larger than zero. This in turn requires that the model be solved via a Non-Linear Programming (NLP) algorithm rather than a LP algorithm. However, the resulting Non-Linear Program remains convex as long as the elasticity parameter is equal to or larger than zero. For additional details on convex programming, see Nemhauser et al (1989). If linearity is desired (for instance if problem instances are very large), we can approximate expression (3) by a sequence of linear segments with increasing slopes, and thus obtain a Linear Program.

The linearization can be done by choosing a suitable range of emissions, and dividing that range into *m* intervals below the reference level, and *n* intervals above the reference level. We also assume a middle interval centered at the reference emission level. To each interval corresponds one step variable *S*. Thus, we have for emissions:

$$EM = \sum_{i=1}^{m} S_i^{lo} + S^{mid} + \sum_{i=1}^{n} S_i^{up} \tag{5}$$

The damage cost can then be written as follows:

$$DAM(EM) = \sum_{i=1}^{m} MC_i^{lo} \cdot S_i^{lo} + MC_0 \cdot S^{mid} + \sum_{i=1}^{n} MC_i^{up} \cdot S_i^{up} \tag{6}$$

where:

- MC_i^{lo} and MC_i^{up} are the approximate marginal costs at each step below and above the reference level as shown in (7) below; and
- S_i^{lo}, S^{mid} and S_i^{up} are the non-negative step variables for emissions. Apart from the final step, each step variable has an upper bound equal to the width of the interval. In this formulation we choose intervals of uniform width on each side of the reference level. However, the intervals below and above the reference level can have different sizes. The width of the middle interval is always the average of the widths below and above the reference level.

The approximate marginal costs at each step can be assumed to be the marginal costs at the center of each step. If all the steps intervals are of equal size, the marginal costs for the steps below the reference level are obtained by the following formula:

$$MC_i^{lo} = MC_0 \cdot \left(\frac{(i-0.5)}{(m+0.5)} \right)^{\beta} \tag{7}$$

Formulas for the marginal costs of the other steps can be derived similarly.

The TIMES implementation basically follows the equations shown above. Both the non-linear and linearized approaches can be used. However, in order to provide some additional flexibility, the implementation supports also defining a threshold level of emissions, below which the damage costs are zero. This refinement can be taken into account in the balance equation (5) by adding one additional step variable having an upper bound equal to the threshold level, and by adjusting the widths of the other steps accordingly. The threshold level can also easily be taken into account in the formulas for the approximate marginal costs.

In addition, the implementation supports different elasticities and step sizes to be used below and above the reference level. See Section 3 for more details.

3 Switches and Parameters

3.1 Activating the Damage Cost Functions

Like all other aspects of TIMES, the user describes the Damage Cost Functions by means of a Set and the Parameters and Switches described in this chapter.

As discussed in Section 2, the TIMES Damage Cost Function facility permits the assessment of environmental externalities by means of two approaches to determine the impact or cost of damages arising from emissions: ex-post calculation and internalized damage costs. The second approach can be further divided into the non-linear and linear formulations, and therefore the following three approaches are available in Standard TIMES:

1. The environmental damages are computed ex-post, without feedback into the optimization process;
2. The environmental damages are a linearized part of the objective function and therefore taken into account in the optimization process;
3. The environmental damages are a non-linear part of the objective function and therefore taken into account in the optimization process.

The user can control whether or not the damage costs are activated in the objective function by means of the switch $SET DAMAGE LP/NLP/NO. This switch is provided by the data handling system according to how the user wishes the option to be included:

```
$SET DAMAGE LP
$SET DAMAGE NLP
$SET DAMAGE NO
```

The setting $SET DAMAGE LP is the default, and activates the linearized formulation of damage costs, with the costs included in the objective function. The setting $SET DAMAGE NLP activates the non-linear damage cost option, with the costs included in the objective function. The setting $SET DAMAGE NO causes the damage costs only to be computed ex-post, without feedback into the optimization process.

Note that owing to the non-linear nature of the modified objective function that endogenizes the damages, the NLP damage option requires non-linear solution methods that can lead to much larger resource utilization compared to LP models. In addition, the options with an augmented objective function cannot be currently activated with the non-linear TIMES-MACRO model variant. However, the linear option LP can be used together with the decomposed MACRO_MSA option.

3.2 Input parameters

All the parameters for describing damage functions are available in the VEDA-FE shell, where they may be specified. All parameters have a prefix 'DAM_' in the GAMS code of the model generator. The parameters are discussed in more detail below:

1. The parameter **DAM_COST** is used to specify the marginal damage cost at the reference level of emissions. The parameter has a year index, which can be utilized also for turning damage accounting on/off for an emission in a period (by specifying an EPS value for the cost). **DAM_COST** is interpolated/extrapolated by default, but unlike other cost parameters, the interpolation is sparse, and the costs are assumed to be constant within each period.
2. The parameter **DAM_BQTY** is used to specify the reference level of emissions. If not specified or set to zero, the marginal damage costs will be assumed constant, and no emission steps are used.
3. The parameter **DAM_ELAST** is used to specify the elasticity of marginal damage costs to emissions in the lower and upper direction. If specified in one direction only, the elasticity is assumed in both directions. If neither is specified, the marginal damage costs will be constant in both directions.
4. The parameter **DAM_STEP** can be used for specifying the number of emission steps below and above the reference level of emissions. The last step above the reference level will always have an infinite bound. If the number of steps is not provided in either direction, but the elasticity is, one step is assumed in that direction. If a non-zero **DAM_STEP**(r,c,'N') is specified, the damage costs for commodity **c** in region **r** are not included in the objective. If the NLP formulation is used (DAMAGE=NLP), all **DAM_STEP** parameters will be ignored.
5. The parameter **DAM_VOC** can be used for specifying the variation in emissions covered by the emission steps, both in the lower an upper direction. The variation in the lower direction should be less than or equal to the reference level of emissions. If the lower variation is smaller than **DAM_BQTY**, the damage costs.

The input parameters are listed in Table B-1.

Table B-1. Input parameters for the TIMES Damage Cost Functions.

Input parameter (Indexes)[47]	Related parameters[48]	Units / Ranges & Default values & Default inter-/extrapolation[49]	Instances[50] (Required / Omit / Special conditions)	Description	Affected equations or variables[51]
DAM_COST (r,datayear,c,cur)	DAM_BQTY, DAM_ELAST, DAM _STEP, DAM _VOC	TIMES cost unit [0, INF); default value: none Default i/e[52]: standard	Required for each commodity for which damage costs are to be accounted.	Marginal damage cost of emission c at reference emission level.	EQ_OBJDAM
DAM_BQTY (r,c)	See above	TIMES emission unit [0, INF); default value: 0	Only taken into account if DAM_COST has been specified	Reference level of emissions c	EQ_DAMAGE EQ_OBJDAM
DAM_ELAST (r,c,bd)	See above	Dimensionless [0, INF); default value: 0	Only taken into account if DAM_COST has been specified	Elasticity of marginal damage cost to emissions on the lower and upper side of the reference level	EQ_OBJDAM
DAM_STEP (r,c,bd)	See above	Dimensionless [0, INF), integer; default value: 0	Only taken into account if DAM_COST is specified. Non-zero 'N' value excludes costs from the objective.	Number of emission steps for the linearized cost function in the lower/upper direction. Can also be used for excluding the costs from the objective.	EQ_DAMAGE EQ_OBJDAM
DAM_VOC (r,c,bd)	See above	TIMES emission unit (0, INF); ≤ DAM_BQTY; default value: DAM_BQTY	Only taken into account if DAM_COST has been specified	Variation in emissions covered by the emission steps in the lower/upper direction. A threshold emission level can be defined with bd='LO'.	EQ_DAMAGE EQ_OBJDAM

[47] The first row contains the parameter name, the second row contains in brackets the index domain over which the parameter is defined.

[48] This column gives references to related input parameters or sets being used in the context of this parameter as well as internal parameters/sets or result parameters being derived from the input parameter.

[49] This column lists the unit of the parameter, the possible range of its numeric value [in square brackets] and the inter-/extrapolation rules that apply.

[50] An indication of circumstances for which the parameter is to be provided or omitted.

[51] Equations or variables that are directly affected by the parameter.

[52] Abbreviation i/e = inter-/extrapolation

3.3 Reporting parameters

There is only one reporting parameter specifically related to the Damage Cost functions. The parameter represents the undiscounted damage costs by region, period and emission commodity. The parameter has two flavours; the first one is for standard TIMES and the second one for stochastic TIMES:

- **CST_DAM(r,t,c)**: Annual damage costs from emission **c** in region **r**,
- **SCST_DAM(w,r,t,c)**: Annual damage costs from emission **c** in region **r** and stochastic scenario **w**.

However, in addition the standard reporting parameters REG_WOBJ, and REG_ACOST are augmented with damage costs results, using the label 'DAM'/'DAM-EXT' to distinguish damage costs from other cost components.

These parameters are included in the .vdd files that describe the parameters to be transferred to VEDA-BE under standard TIMES and stochastic TIMES. Therefore, the corresponding result parameter is always available in VEDA-BE whenever Damage Cost functions have been defined, even with the setting DAMAGE=NO.

The damage costs are always reported by using the accurate non-linear expressions, even if the linearized formulation is chosen for the augmented objective function.

Table B-2. Reporting parameters for the TIMES Damage cost functions.

Parameter	Description
CST_DAM(r,t,c)	Damage costs by region, period and emission (standard TIMES)
SCST_DAM (w,r,t,c)	Damage costs by region, period and emission (stochastic TIMES)

4 Examples

Assume that we wish to define linearized damage costs for the emission commodity 'EM' so that the cost function has the following properties:

- The reference level of emissions is 80 units;
- The marginal cost at the reference level are 10 cost units per emission unit;
- The cost elasticity is 1 in the lower direction, and 0.7 in the upper direction;

The damage function can be specified with the following parameters:

```
PARAMETER DAM_COST     / REG.2000.EM.CUR 10 /;
PARAMETER DAM_BQTY     / REG.EM 80 /;
PARAMETER DAM_ELAST    / REG.EM.LO 1, REG.EM.UP 0.7 /;
```

As we did not specify the number of steps, but we did specify the elasticities in both directions, the number of steps is assumed to be 1 in both directions. The resulting damage cost function is illustrated in Figure 15. Because the damage function has a very coarse representation, the total costs have notable deviations from the accurate non-linear function. Note that the step size has been automatically determined to be **DAM_BQTY/(DAM_STEP+0.5)** = 80/1.5. However, the last step has no upper bound.

Assume next that we would like to refine the damage function by the following specifications:

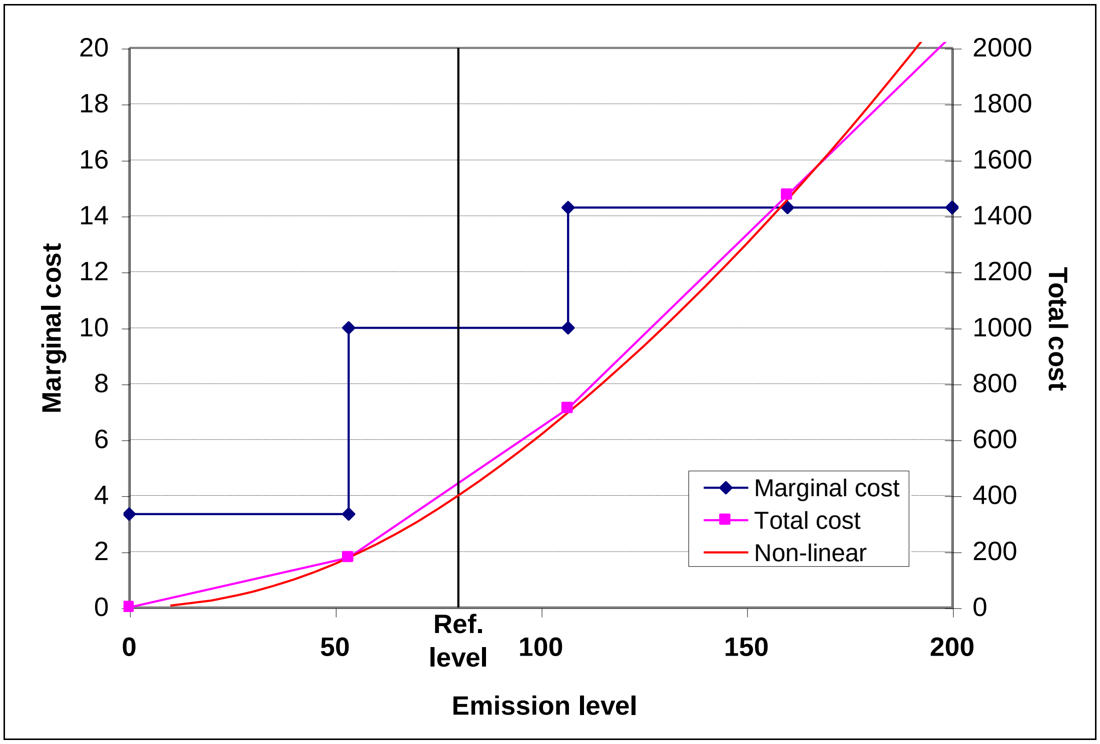


Figure 15. Example of a linearized damage function with 1+1+1 steps (1 lower step, 1 middle step, 1 upper step).

- We want to have 5 steps below the reference, and 3 steps above it;
- The threshold level of damage costs is 20 units of emissions;
- The steps above the reference level should cover 100 units of emissions.

The damage function can be specified with the following parameters

```
PARAMETER DAM_COST     / REG.2000.EM.CUR 10 /;
PARAMETER DAM_BQTY     / REG.EM 80 /;
PARAMETER DAM_ELAST    / REG.EM.LO 1, REG.EM.UP 0.7 /;
PARAMETER DAM_STEP     / REG.EM.LO 5, REG.EM.UP 3 /;
PARAMETER DAM_VOC      / REG.EM.LO 60, REG.EM.UP 100 /;
```

The resulting damage cost function is illustrated in Figure 16. The cost function follows now very closely the accurate non-linear function. Note that the step sizes derived from the VOC specifications are 10 units for the lower steps, 20 for the middle step, and 30 units for the upper steps. However, the last step of course has no upper bound.

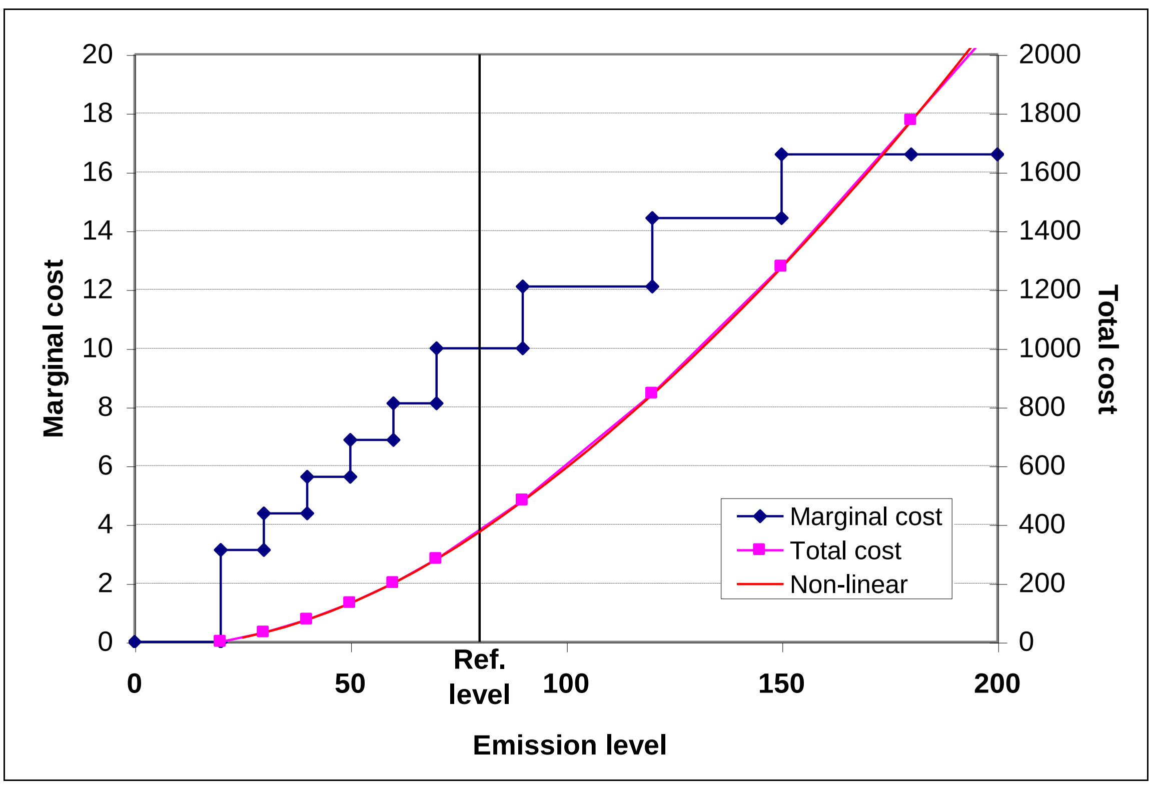


Figure B-16. Example of a linearized damage function with 1+5+1+3 steps (one zero cost step, 5 lower steps, one middle step, 3 upper steps).

5 Variables

There are only two sets of new variables in the damage cost formulation, VAR_DAM and VAR_OBJDAM, which are shown below in Table B-3. The variables VAR_DAM represent the steps in the emissions in each period. In the linearized formulation, there are DAM_STEP(...,'LO') number of step variables on the lower side and DAM_STEP(...'UP') number of step variables on the higher side of emissions. In addition, one step variable of type 'FX' corresponds to the middle step that includes the reference level of emissions, and an optional additional step variable of type 'FX' corresponds to the zero-damage fraction of emissions, as defined by the difference between DAM_BQTY(..) and DAM_VOC(...,'LO').

The variables VAR_OBJDAM represent the total discounted damage costs by region. The undiscounted costs in each period described in Section 2 are discounted and summed over all periods and emissions in each region. As emissions are in TIMES assumed to be constant within each period, damage costs are likewise assumed to be constant within each period.

Table B-3. Model variables specific to the Damage Cost Functions.

Variable (Indexes)	Variable Description
VAR_DAM (r,t,c,bd,j)	The emission step variable for the damage function of commodity **c** in region **r**, for each step **j** in each direction **bd**.
VAR_OBJ (r,'OBJDAM',cur)	The variable is equal to the sum of the total discounted damage costs in each region **r** with currency **cur**.

5.1 VAR_DAMAGE(r,t,c,bd,j)

Description: The amount of emission indicator **c** at cost step **j** in direction **bd**, in period **t**.

Purpose: This variable tracks the amount of an emission indicator by cost step and period, in both the lower and upper direction from the reference level.

Occurrence: The variable is generated for emission indicator that has damage costs specified, whenever the damage cost functions are included in the objective function.

Units: Units of the emission commodity **c**.

Bounds: This variable cannot be directly bounded by the user.

5.2 VAR_OBJ(r,'OBJDAM',cur)

Description: The total present value of damage costs by region.

Purpose: This variable is included in the objective function in order to include damage costs in the objective when requested by the user.

Occurrence: This variable is generated for each region when damage cost functions are included in the objective function

Units: Currency units used for damage functions.

Bounds: This variable cannot be directly bounded by the user.

6 Equations

There are two blocks of equations generated for damage cost functions, whenever they are included in the objective function. The two equations related to the damage functions are listed and briefly described below in Table B-4. The equations include the balance of stepped emissions, the objective component for damage costs, and the augmented total objective function.

In addition, the standard TIMES objective function, **EQ_OBJ**, is augmented by the present value of the damage costs, as defined by the equation EQ_OBJDAM.

We now give the formulations of these constraints.

> Reminder: the Damage Cost Functions are activated at run time from the data handling system, which in turn sets the switch $SET DAMAGE LP/NLP/NO.

Table B-4. Constraints specific to damage costs (in the GAMS file eqdamage.mod).

Constraints (Indexes)	Constraint Description
EQ_DAMAGE (r,t,c)	The balance equation between the stepped emission variables and the total emissions in each period.
EQ_OBJDAM (r,cur)	The total discounted damage costs by region, which will be added as a component to the objective function.

6.1 EQ_DAMAGE(r,t,c)

Description: Allocates the total amount of emission indicator in period **t** to cost steps.

Purpose: This constraint allocates the total amount of emission indicator c to the cost steps of the linearized / non-linear damage cost functions in each period ***t***.
This equation is generated in each time period for all emission indicators considered.

Units: Units of the emission commodity **c**.

Type: *Binding.* The equation is an equality (=) constraint.

Remarks:

- The damage costs can be defined either on the net production (VAR_COMNET) or the gross production (VAR_COMPRD) of the commodity c. By default the damage costs are applied to the NET amount, unless $DAM\_ELAST_{r,c,'N'}$ is also specified. $DAM\_ELAST_{r,c,'N'}$ defines a multiplier for the Base prices to be added to the damage cost function, when it is to be applied to the gross production.
- The internal parameter $DAM\_COEF_{r,t,c,s}$ is set to the base prices, if $DAM\_ELAST_{r,c,'N'}$ is specified, and otherwise to 1.

Equation:

$$EQ\_DAMAGE_{r,t,c} \qquad \ni \left(\mathbf{rtc}_{r,t,c} \wedge \exists(cur): DAM\_COST_{r,t,c,cur}\right)$$

$$\sum_{(jj,bd)\in\mathbf{dam\_num}_{r,c,jj,bd}} \sum_{j\le jj} VAR\_DAM_{r,t,c,bd,j}$$

$$\{=\}$$

$$\sum_{\mathbf{com\_ts}_{r,c,ts}} \left(\begin{array}{l} DAM\_COEF_{r,t,c,ts} \times \\ \left(\begin{array}{ll} VAR\_COMNET_{r,t,c,ts} & if\ DAM\_ELAST_{r,c,'N'}\ not\ given \\ VAR\_COMPRD_{r,t,c,ts} & otherwise \end{array} \right) \end{array} \right)$$

6.2 EQ_OBJDAM(r,cur)

Description: Computes the present value of all damage costs by region and currency.

Purpose: Defines the variable VAR_OBJ(r,'OBJDAM',cur), which represents the total present value of all damage costs in region r, having currency cur. This variable is included in the TIMES objective function.

Units: Currency units.

Type: *Binding.* The equation is an equality (=) constraint.

Remarks:

- The internal parameter $DAM\_SIZE_{r,c,bd}$ represents the sizes of cost steps of the dlinearized damage cost function, for both directions (bd=LO/UP) and for the middle step (bd=FX), as described above in Section 2.

Equation:

$$EQ\_OBJDAM_{r,cur} \quad \ni \left(\mathbf{rdcur_{r,cur}}\right)$$

Case A: Linearized functions

$$\sum_{(t,c)\in\left\{\mathbf{rtc_{r,t,c}}|(DAM\_COST_{r,t,}>0)\right\}} DAM\_COST_{r,t,c,cur} \times OBJ\_PVT_{r,t,cur} \times$$

$$\left[\begin{array}{l}
\sum\limits_{\substack{jj\in\mathbf{dam_num}_{r,c,jj,'LO'} \\ j\le jj}} \left(\begin{array}{l} \dfrac{VAR_DAM_{r,t,c,'LO',j}}{DAM_BQTY_{r,c}^{\;DAM_ELAST_{r,c,'LO'}}} \times \\ \left(\begin{array}{l} DAM_BQTY_{r,c} - DAM_VOC_{r,c,'LO'} + \\ DAM_SIZE_{r,c,'LO'} \times (j-0.5) \end{array} \right)^{DAM_ELAST_{r,c,'LO'}} \end{array} \right) + \\
VAR_DAM_{r,t,c,'FX',1} \\
\sum\limits_{\substack{jj\in\mathbf{dam_num}_{r,c,jj,'UP'} \\ j\le ORD(jj)}} \left(\begin{array}{l} \dfrac{VAR_DAM_{r,t,c,'UP',j}}{DAM_BQTY_{r,c}^{\;DAM_ELAST_{r,c,'UP'}}} \times \\ \left(\begin{array}{l} DAM_BQTY_{r,c} + \dfrac{DAM_SIZE_{r,c,'FX'}}{2} + \\ DAM_SIZE_{r,c,'UP'} \times (j-0.5) \end{array} \right)^{DAM_ELAST_{r,c,'UP'}} \end{array} \right)
\end{array}\right]$$

$$\{=\}$$

$$VAR\_OBJ_{r,'OBJDAM',cur}$$

Case B: Non-linear functions

$$\sum_{(t,c)\in\{\mathbf{rtc_{r,t,c}}|(DAM\_COST_{r,t,}>0)\}} DAM\_COST_{r,t,c,cur} \times OBJ\_PVT_{r,t,cur} \times$$

$$\left[\begin{array}{l}
\dfrac{\left(\left(\begin{array}{l} VAR_DAM_{r,t,c,'LO',j} + \\ DAM_BQTY_{r,c} - DAM_VOC_{r,c,'LO'} \end{array}\right)^{(DAM_ELAST_{r,c,'LO'}+1)} - \left(DAM_BQTY_{r,c} - DAM_VOC_{r,c,'LO'}\right)^{(DAM_ELAST_{r,c,'LO'}+1)}\right)}{DAM_BQTY_{r,c}{}^{DAM_ELAST_{r,c,'LO'}} \times \left(DAM_ELAST_{r,c,'LO'}+1\right)} + \\
\\
\dfrac{\left(\left(VAR_DAM_{r,t,c,'UP',j} + DAM_BQTY_{r,c}\right)^{(DAM_ELAST_{r,c,'UP'}+1)} - \left(DAM_BQTY_{r,c}\right)^{(DAM_ELAST_{r,c,'UP'}+1)}\right)}{DAM_BQTY_{r,c}{}^{DAM_ELAST_{r,c,'UP'}} \times \left(DAM_ELAST_{r,c,'UP'}+1\right)}
\end{array}\right]$$

$$\{=\}$$

$$VAR\_OBJ_{r,'OBJDAM',cur}$$

References

Goldstein, G., Noble, K. & Van Regemorter, D. 2001. *Adaptation to MARKAL for including environmental damages*. MARKAL User Information Note.

Loulou, R., Goldstein, G. & Noble, K. 2004. *Documentation for the MARKAL Family of Models*. October 2004. http://www.iea-etsap.org/web/Documentation.asp

Loulou, R., Remme, U., Kanudia, A., Lehtilä, A. & Goldstein, G. 2005. *Documentation for the TIMES Model*. Energy Technology Systems Ananlysis Programme (ETSAP), April 2005. http://www.iea-etsap.org/web/Documentation.asp

Nemhauser, G.L., Rinnooy Kan, A.H.G. & Todd, M.J. (eds.) 1989. Handbooks in Operations Research and Management Science, Vol I: Optimization. North-Holland.

Appendix C
Endogenous Technological Learning (ETL)

1 Introduction

As discussed in Chapter 11 of Part I, there are situations in which the rate at which a technology's unit investment cost changes over time is a function of cumulative investment in the technology. In these situations, technological learning is called endogenous.

Mixed Integer Programming (MIP) is employed in order to model Endogenous Technological Learning (ETL) in TIMES. As has already been noted in the case of Lumpy Investments, MIP problems are much more difficult to solve than standard LP problems, and so the ETL feature should be applied only where it is deemed necessary to model a limited number of technologies as candidates for Endogenous Technological Learning. This caution is especially required for large-scale TIMES instances. Another important caveat is that ETL is relevant when the modeling scope is broad e.g. when a large portion of (or perhaps the entire) world energy system is being modeled, since the technological learning phenomenon rests on global cumulative capacity of a technology, and not on the capacity implemented in a small portion of the world.

In this chapter we provide the data and modeling details associated with modeling Endogenous Technological Learning (ETL) in TIMES. The implementation of ETL in TIMES is based on the realization in the MARKAL model generator. The major part of the MARKAL code for ETL could be transferred to TIMES. Accordingly the description of ETL presented here follows the MARKAL documentation of ETL. To this end the next three sections will address the Sets, Parameters, Variables, and Equations related to the Endogenous Technological Learning option, including the special clustered learning ETL option where a component common to several technologies learns, thereby benefiting all the related (clustered) technologies.

2 Sets, Switches and Parameters

Like all other aspects of TIMES the user describes the ETL components of the energy system by means of a Set and the Parameters and Switches described in this chapter. Table C-1 and Table C-2 below describe the User Input Parameters, and the Matrix Coefficient and Internal Model Sets and Parameters, respectively, that are associated with the Endogenous Technological Learning option. Note that the special clustered learning ETL option requires one additional User Input Parameter (ETL-CLUSTER), and two additional Matrix Coefficient/Internal Model Parameters (CLUSTER and NTCHTEG).

Besides the basic data described in Table the user controls whether or not the ETL component is activated by means of the $SET ETL 'YES' switch. This switch is provided by the data handling system when the user indicates that the ETL option is to be included in a run. This permits the easy exclusion of the feature if the user does not want to perform a MIP solve without having to remove the ETL data.

Table C-1. Definition of ETL user input parameters

Input Parameter (Indexes)	Alias / Internal Name	Related Parameters	Units/Range & Defaults	Instance (Required/Omit/ Special Conditions)	Description
CCAP0 (r,p)	TL_CCAP0	*PAT* *CCOST0*	• Units of capacity (e.g., GW, PJa). • [open]; no default.	• Required, along with the other ETL input parameters, for each learning technology (TEG).	The initial cumulative capacity (starting point on the learning curve) for a (non-resource) technology that is modeled as one for which endogenous technology learning (ETL) applies. Learning only begins once this level of installed capacity is realized. • The CCAP0 parameter appears as the right-hand-side of the cumulative capacity definition constraint (EQ_CUINV). • Note that if the NCAP_PASTI parameter is specified for an ETL technology, then its value in the first period should match the value of CCAP0, otherwise an infeasibility will occur.
CCAPM (r,p)	TL_CCAPM	*CCOSTM*	• Units of capacity (e.g., GW, PJa). • [open]; no default	• Required, along with the other ETL input parameters, for each learning technology (TEG).	The maximum cumulative capacity (ending point on the learning curve) for a (non-resource) technology that is modeled as one for which endogenous technology learning (ETL) applies. • The parameter CCAPM does not appear in any of the ETL constraints, but its value affects the values of a number of internal parameters that directly contribute to one or more of the ETL constraints.
TEG (p)	TEG	*ETL-CUMCAP0* *ETL-CUMCAPMAX* *ETL-INVCOST0* *ETL-NUMSEG* *ETL-PROGRATIO*	• Indicator. • [1]; no default.	• Required to identify the learning technologies. • For each TEG the other ETL input parameters are	An indicator (always 1) that a process is modeled as one for which endogenous technology learning (ETL) applies. • The set TEG controls the generation of the ETL constraints. Each of the ETL constraints is generated only for those technologies that are in set TEG.

Input Parameter (Indexes)	Alias / Internal Name	Related Parameters	Units/Range & Defaults	Instance (Required/Omit/ Special Conditions)	Description
				required.	
SC0 (r,p)	TL_SC0	*PAT*	• Base year monetary units per unit of capacity (e.g., 2000 M$/GW or PJa). • [open]; no default.	• Required, along with the other ETL input parameters, for each learning technology (TEG).	The investment cost corresponding to the starting point on the learning curve for a technology that is modeled as one for which endogenous technology learning (ETL) applies. • The parameter SC0 does not appear in any of the ETL constraints, but its value affects the values of a number of internal parameters that directly contribute to one or more of the ETL constraints.
SEG (r,p)	TL_SEG	*ALPH* *BETA* *CCAPK* *CCOSTK*	• Number of steps. • [1-6]; no default.	• Required, along with the other ETL input parameters, for each learning technology (TEG).	The number of segments to be used in approximating the learning curve for a technology that is modeled as one for which endogenous technology learning (ETL) applies. • The SEG parameter appears in all of the ETL constraints that are related to piecewise linear approximation of the learning curve (EQ_CC, EQ_COS, EQ_EXPE1, EQ_EXPE2, EQ_LA1, EQ_LA2).
PRAT (r,p)	TL_PRAT	*CCAPK* *CCOST0* *CCOSTM* *PAT* *PBT*	• Decimal fraction. • [0-1]; no default.	• Required, along with the other ETL input parameters, for each learning technology (TEG).	The “progress ratio” for a technology that is modeled as one for which endogenous technology learning (ETL) applies. The progress ratio, which is referred to as the learning rate, is defined as the ratio of the change in unit investment cost each time cumulative investment in an ETL technology doubles. That is, if the initial unit investment cost is SC0 and the progress ratio is PRAT, then after cumulative investment is doubled the unit investment cost will be PRAT * SC0. • The parameter PRAT does not appear in any of the ETL constraints, but its value affects the

Input Parameter (Indexes)	Alias / Internal Name	Related Parameters	Units/Range & Defaults	Instance (Required/Omit/ Special Conditions)	Description
					values of a number of internal parameters (ALPH, BETA, CCAPK, CCOST0) that directly contribute to one or more of the ETL constraints.
CLUSTER (r,p,p)	TL_CLUSTER NCLUSTER	*TL_MRCLUST*	• Decimal fraction. • [0-1]; no default.	• Provided to model clustered endogenous technology learning. • Each of the learning parameters must also be specified for the key learning technology.	The "cluster mapping and coupling factor" for a technology that is modeled as a clustered technology is associated with a key learning technology to which endogenous technology learning (ETL) applies. Clustered technologies use the key ETL technology, and are subject to learning via the key technology. • The first index of the CLUSTER parameter is a key learning technology. • The second index of the CLUSTER parameter is a clustered technology that is associated with this key learning technology. • In general there may be several clustered technologies each of which is associated with the same key learning technology, and hence there may be several instances of the CLUSTER parameter each of which has the same key learning technology as its first index. • The numerical value of the CLUSTER parameter indicates the extent of coupling between the clustered technology and the key learning technology to which it is associated.
TL_MRCLUST (r,teg,reg,p)		*CLUSTER*	• Decimal fraction. • [0-1]; no default.	• See CLUSTER	The multi-region cluster mapping and coupling factor. Similar to CLUSTER, but may be used to map technologies p in multilple regions reg to key components teg in region r. See CLUSTER.

Table C-2. ETL-specific matrix coefficient and internal model parameters[53]

Matrix Controls & Coefficients (indexes)	Type	Description & Calculations
ALPH (r,k,p)	I	ALPH are the intercepts on the vertical axis of the line segments in the piecewise linear approximation of the cumulative cost curve. They are calculated in COEF_ETL.ETL from the starting and ending points of the cumulative cost curve, its assumed form, the number of segments used in its piecewise linear approximation, and the choice of successive interval lengths on the vertical axis to be such that each interval is twice as wide as the preceding one. The parameter ALPH occurs in the ETL equation EQ_COS that defines the piecewise linear approximation to the cumulative cost curve.
BETA (r,k,p)	I	BETA are the slopes of the line segments in the piecewise linear approximation of the cumulative cost curve. They are calculated in COEF_ETL.ETL from the starting and ending points of the cumulative cost curve, its assumed form, the number of segments used in its piecewise linear approximation, and the choice of successive interval lengths on the vertical axis to be such that each interval is twice as wide as the preceding one. The parameter BETA occurs in the ETL equation EQ_COS that defines the piecewise linear approximation to the cumulative cost curve.
CCAP0 (r,p)	A	CCAP0 is the initial cumulative capacity (starting point on the learning curve). The parameter CCAP0 occurs in the ETL equation EQ_CUINV that defines cumulative capacity in each period.
CCAPK (k,p)	I	CCAPK are the break points on the horizontal axis in the piecewise linear approximation of the cumulative cost curve. They are calculated in COEF_ETL.ETL from the starting and ending points of the cumulative cost curve, its assumed form, the number of segments used in its piecewise linear approximation, and the choice of successive interval lengths on the vertical axis to be such that each interval is twice as wide as the preceding one. The parameter CCAPK occurs in the ETL equations EQ_LA1 and EQ_LA2 whose role is to ensure that variable R_LAMB(r,t,k,p) lies in the k^{th} interval, i.e., between CCAPK(r,k-1,p) and CCAPK(r,k,p), when its associated binary variable R_DELTA(r,t,k,p) = 1.
CCOST0 (r,p)	I	CCOST0 is the initial cumulative cost (starting point on the learning curve). It is calculated in COEF_ETL.ETL from the initial cumulative capacity (CCAP0) and corresponding initial investment cost (user input parameter SC0) and the progress ratio (user input parameter PRAT). The parameter CCOST0 occurs in the ETL equation EQ_IC1 that defines first period investment costs (prior to discounting).

[53] Parameters that occur in the ETL-specific equations but that also occur in non-ETL equations (e.g., TCH_LIFE) are not listed in this table.

Matrix Controls & Coefficients (indexes)	Type	Description & Calculations
SEG (r,p)	A	The user input parameter SEG is the number of segments in the cumulative cost curve. The parameter SEG occurs in all of those ETL equations that are related to the piecewise linear approximation of the cumulative cost curve.
TEG (p)	S	TEG is the set of technologies to which endogenous technology learning (ETL) applies. Each of the ETL equations has set TEG as an index.
CLUSTER (r,p,p)	I	The user input parameter CLUSTER (cluster mapping and coupling factor) is only relevant when modeling clustered endogenous technology learning. The parameter occurs in the special ETL cluster equation EQ_CLU that defines investment in new capacity (VAR_NCAP) in the key learning technology as the weighted sum of investments in new capacity of the clustered technologies that are attached to the key technology. (The weights used are the numeric values of the CLUSTER parameter.)
TL_MRCLUST (r,teg,reg,p)	I	The user input parameter TL_MRCLUST is only relevant when modeling clustered endogenous technology learning. The parameter occurs in the special ETL cluster equation EQ_MRCLU that defines investment in new capacity (VAR_NCAP) in the key learning technology as the weighted sum of investments in new capacity of the clustered technologies that are attached to the key technology.
NTCHTEG (r,p)	I	The parameter NTCHTEG is only relevant when modeling clustered endogenous technology learning. If TEG is an ETL technology, then NTCHTEG(R,TEG) is the number of clustered technologies that are attached to key technology TEG. NTCHTEG is calculated in COEF_ETL.ETL from the "cluster mapping and coupling factor" (CLUSTER). It occurs in the special ETL cluster equation EQ_CLU.
PBT (r,p)		The learning index PBT is an internal parameter calculated in COEF_ETL.ETL. It is derived from the progress ratio PRAT using the formula: PBT(r,p) = -log(PRAT(r,p))/log(2). PBT does not occur directly in the equations, but is used in the calculation of equation coefficients.
PAT (r,p)		The internal parameter PAT describes the specific investment costs of the first unit. It is derived in COEF_ETL.ETL using PBT, SC0 and CCAP0. PAT does not occur directly in the equations, but is used in the calculation of equation coefficients.
K		The set K has the members '1'-'6' and is used as indicator for the kink points of the piecewise linear approximation of the cumulative cost curve. The number of elements can be changed in the *run file if desired.
WEIG (r,k,prc)	I	The internal parameter WEIG is calculated in COEF_ETL.ETL and is used as a factor in the calculation of the length of the intervals being used in the piecewise linear approximation of the cumulative cost curve. The interval lengths on the vertical axis are chosen in such a way that each interval is twice as wide as the preceding one.

3 Variables

The variables that are used to model the Endogenous Technological Learning option in TIMES are presented in Table below. As is the case with the modeling of lumpy investments, the primary role of the variables and equations used to model ETL is to control the standard TIMES investment variable (VAR_NCAP) and the associated dynamic cost of these investments, so ETL is rather self-contained. That is the VAR_NCAP variable links the ETL decisions to the rest of the model, and the VAR_IC investment cost variable determines the associated contribution to the regional investment costs (VAR_OBJINV). Note that the special clustered learning ETL option does not require any additional variables, as compared with the modeling of endogenous technology learning when there are no clusters.

Table C-3. ETL-specific model variables

Variable (Indexes)	Variable Description
VAR_CCAP (r,t,p)	The cumulative investment in capacity for an ETL technology. This variable represents the initial cumulative capacity (CCAP0) plus investments in new capacity made up to and including the current period. This variable differs from the total installed capacity for a technology (VAR_CAP) in that it includes all investments in new capacity made up to and including the current period, whereas the latter only includes investments that are still available (i.e. whose life has not expired yet).
VAR_CCOST (r,t,p)	The cumulative cost of investment in capacity for an ETL technology. The cumulative cost is interpolated from the piecewise linear approximation of the cumulative cost curve.
VAR_DELTA (r,t,p,k)	Binary variable (takes the value 0 or 1) used for an ETL technology to indicate in which interval of the piecewise linear approximation of the cumulative cost curve the cumulative investment in capacity (VAR_CCAP) lies. A value of 1 for this variable for exactly one interval k indicates that VAR_CCAP lies in the k^{th} interval.
VAR_IC (r,t,p)	The portion of the cumulative cost of investment in capacity for an ETL technology (VAR_CCOST) that is incurred in period t, and so subject to the same discounting that applies to other period t investment costs. This variable is calculated as the difference between the cumulative costs of investment in capacity for periods t and t-1, and enters the regional investment cost part of the objective function (EQ_OBJINV)
VAR_LAMBD (r,t,p,k)	Continuous variable used for an ETL technology to represent the portion of cumulative investment in capacity (VAR_CCAP) that lies in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve. For a given ETL technology and given time period, ETL model constraints involving this variable and the associated binary variable VAR_DELTA ensure that VAR_LAMBD is positive for exactly one interval k.

3.1 VAR_CCAP(r,t,p)

Description: The cumulative investment in capacity for an ETL technology.

Purpose and Occurrence: This variable tracks the cumulative investment in capacity for an ETL technology which then determines, along with the progress ratio, how much the investment cost is to be adjusted for the learning gains.

This variable is generated for each ETL technology in all time periods beginning from the period that the technology is first available. It appears in the cumulative capacity definition constraint (EQ_CUINV) that defines it as the initial cumulative capacity (CCAP0) plus investments in new capacity (VAR_NCAP) made up to and including the current period. It also appears in the cumulative capacity interpolation constraint (EQ_CC). This constraint equates VAR_CCAP(r,t,p) to the sum over k of the variables VAR_LAMBD(r,t,p,k) used to represent the cumulative investment in capacity lying in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve.

Units: PJ/a, Gw, or Bvkm/a, or any other unit defined by the analyst to represent technology capacity.

Bounds: This variable is not directly bounded. It may be indirectly bounded by specifying a bound (NCAP_BND) on the level of investment in new capacity (VAR_NCAP).

3.2 VAR_CCOST(r,t,p)

Description: The cumulative cost of investment in capacity for an ETL technology.

Purpose and Occurrence: This variable defines the interpolated cumulative cost of investment in capacity in terms of the continuous variables VAR_LAMBD and the binary variables VAR_DELTA, and the internal model parameters ALPH and BETA. ALPH and BETA represent the intercepts on the vertical axis and the slopes, respectively, of the line segments in the piecewise linear approximation of the cumulative cost curve.

This variable is generated for each ETL technology in all time periods beginning from the period that the technology is first available. It appears in the cumulative cost interpolation equation (EQ_COS) that defines it. It also appears in the equations EQ_IC1 and EQ_IC2 that define the

VAR_IC variables that represent the portions of the cumulative cost of investment in capacity that are incurred in period t.

Units: Million 2000 US$, or any other unit in which costs are tracked.

Bounds: None.

3.3 VAR_DELTA(r,t,p,k)

Description: *Binary* variable (takes the value 0 or 1) used for an ETL technology to indicate in which interval of the piecewise linear approximation of the cumulative cost curve the cumulative investment in capacity (VAR_CCAP) lies.

Purpose and Occurrence: To indicate which step on the learning curve a technology achieves. A value of 1 for this variable for interval k, and zero values for intervals ≠ k, imply that the cumulative investment in capacity (VAR_CCAP) lies in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve.

This binary variable, along with the associated continuous variable VAR_LAMBD, are generated for each ETL technology in all time periods beginning from the period that the technology is first available, and for each interval in the piecewise linear approximation. It appears in the constraint EQ_DEL, whose purpose is to ensure that, for each ETL technology in each period, it has a value of 1 for exactly one interval k (with zero values for intervals ≠ k); and in the cumulative cost interpolation constraint (MR_COS). It also appears in the pair of constraints EQ_LA1 and EQ_LA2, whose purpose is to ensure that VAR_LAMBD, if positive for interval k, is between the two break points on the horizontal axis for interval k in the piecewise linear approximation. (See below under "Purpose and Occurrence" for the variable VAR_LAMBD.)

Finally, this binary variable appears in two constraints EQ_EXPE1 and EQ_EXPE2, whose purpose is to reduce the domain of feasibility of the binary variables and thereby improve solution time for the Mixed Integer Program (MIP).

Units: None. This is a binary variable that takes the value 0 or 1.

Bounds: This binary variable is not directly bounded.

3.4 VAR_IC(r,t,p)

Description: The portion of the cumulative cost of investment in capacity for an ETL technology (VAR_CCOST) that is incurred in period t.

Purpose and Occurrence: This variable represents the portion of the cumulative cost of investment in capacity for an ETL technology that is incurred in period t, and so is subject to the same discounting in the investment cost part of the objective function (EQ_OBJINV) that applies to other period t investment costs.

This variable is calculated as the difference between the cumulative costs of investment in capacity for period t and t-1, and is generated for each ETL technology in all time periods beginning from the period that the technology is first available. Apart from its appearance in the objective function, this variable appears in the constraints EQ_IC1 and EQ_IC2 that define it in the first period that the technology is available, and in subsequent periods, respectively. It also appears in the salvage of investments constraint (EQ_OBJSALV), which calculates the amount to be credited back to the objective function for learning capacity remaining past the modeling horizon.

Units: Million 2000 US$, or any other unit in which costs are tracked.

Bounds: None.

3.5 VAR_LAMBD(r,t,p,k)

Description: *Continuous* variable used for an ETL technology to represent the portion of cumulative investment in capacity (VAR_CCAP) that lies in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve.

Purpose and Occurrence: A positive value for this variable for interval k, and zero values for intervals ≠ k, imply that the cumulative investment in capacity (VAR_CCAP) lies in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve. This continuous variable, along with the associated binary variable VAR_DELTA, are generated for each ETL technology in all time periods beginning from the period that the technology is first available (START), and for each interval in the piecewise linear approximation.

Since this variable represents the portion of the cumulative investment in capacity (VAR_CCAP) that lies in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve, the value of

EQ_LAMBD – if positive – is required to be between CCAPK(k-1,p) and CCAP(k,p), where the internal model parameters CCAPK are the break points on the horizontal axis in the piecewise linear approximation of the cumulative cost curve. A zero value for VAR_LAMBD is also allowed. These requirements on the value of VAR_LAMBD are imposed via the pair of constraints EQ_LA1 and EQ_LA2, in which the value for VAR_LAMBD is subject to lower and upper bounds of CCAPK(k-1,p) * VAR_DELTA and CCAP(k,p) * VAR_DELTA respectively, where VAR_DELTA = VAR_DELTA(r,t,p,k) is the binary variable associated with VAR_LAMBD = VAR_LAMBD(r,t,p,k).

This variable also appears in the cumulative capacity interpolation constraint (EQ_CC), and the cumulative cost interpolation constraint (EQ_COS).

Units: PJ/a, Gw, or Bvkm/a, or any other unit defined by the analyst to represent technology capacity.

Bounds: The pair of constraints EQ_LA1 and EQ_LA2 that are discussed above have the effect of either bounding VAR_LAMBD between CCAPK(k-1,p) and CCAP(k,p), or forcing VAR_LAMBD to be zero.

4 Equations

The equations that are used to model the Endogenous Technological Learning option in TIMES are presented in Table C-4 below. Since the primary role of the variables and equations used to model ETL is to control the standard TIMES investment variable (VAR_NCAP) and the associated dynamic cost of these investments, ETL is rather self-contained. That is the VAR_NCAP variable links the ETL decisions to the rest of the model, and the VAR_IC investment cost variable determines the associated contribution to the regional investment cost part objective function (EQ_OBJINV). Note that the special clustered learning ETL option involves one additional equation (EQ_CLU), as compared with the modeling of endogenous technology learning where there are no clusters. IN BOX BELOW, ADD ANSWER or CHANGE TO "system"

> Reminder: the ETL formulation is activated at run time from the data handling system, which in turn sets the $SET ETL 'YES' switch.

Table C-4. ETL-specific model constraints

Constraints (Indexes)	Constraint Description	GAMS Ref
EQ_CC (r,t,p)	The Cumulative Capacity Interpolation constraint for an ETL technology. This constraint defines the cumulative investment in capacity for a technology (VAR_CCAP) in a period as the sum over all intervals k of the continuous variables R_LAMBD(r,t,p,k) that represent cumulative investment in capacity as lying in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve.	EQU_EXT.ETL
EQ_CLU (r,t,p)	Constraint that is generated only for the special clustered learning ETL option (CLUSTER). For a key learning ETL technology it defines investment in new capacity (VAR_NCAP) as the weighted sum of investments in new capacity of the associated clustered technologies.	EQU_EXT.ETL
EQ_COS (r,t,p)	The Cumulative Cost Interpolation constraint for an ETL technology. This constraint defines the interpolated cumulative cost of investment in capacity for a technology (VAR_CCOST) in a period in terms of the binary variables VAR_DELTA and the continuous variables VAR_LAMBD, and the internal model parameters ALPH and BETA.	EQU_EXT.ETL
EQ_CUINV (r,t,p)	The Cumulative Capacity Definition constraint for an ETL technology. Defines the cumulative investment in capacity for a technology in a period as the initial cumulative capacity (CCAP0) plus the sum of investments in new capacity (VAR_NCAP) made up to and including this period.	EQU_EXT.ETL

Constraints (Indexes)	Constraint Description	GAMS Ref
EQ_DEL (r,t,p)	The constraint for an ETL technology that ensures that in each period there is exactly one interval k for which the binary variable R_DELTA(r,t,p,k) has value 1 (with zero values for intervals ≠ k).	EQU_EXT.ETL
EQ_EXPE1 (r,t,p,k)	One of two constraints for an ETL technology to improve MIP solution time by reducing the domain of feasibility of the binary variables VAR_DELTA.	EQU_EXT.ETL
EQ_EXPE2 (r,t,p,k)	Second of two constraints for an ETL technology to improve MIP solution time by reducing the domain of feasibility of the binary variables VAR_DELTA.	EQU_EXT.ETL
EQ_IC1 (r,t,p)	The constraint for an ETL technology that defines the portion of the cumulative cost of investment in capacity (VAR_IC) that is incurred in the first period of the model horizon.	EQU_EXT.ETL
EQ_IC2 (r,t,p)	The constraint for an ETL technology that defines the portion of the cumulative cost of investment in capacity (VAR_IC) that is incurred in each period but the first one.	EQU_EXT.ETL
EQ_LA1 (r,t,p,k)	The constraint for an ETL technology that sets a lower bound on the continuous variable VAR_LAMBD(r,t,p,k).	EQU_EXT.ETL
EQ_LA2 (r,t,p,k)	The constraint for an ETL technology that sets an upper bound on the continuous variable VAR_LAMBD(r,t,p,k).	EQU_EXT.ETL
EQ_MRCLU (r,t,p)	Constraint that is generated only for the special clustered learning ETL option (TL_MRCLUST). For a key learning ETL technology it defines investment in new capacity (VAR_NCAP) as the weighted sum of investments in new capacity of the associated clustered technologies in multilple regions.	EQU_EXT.ETL
EQ_OBJSAL (r,cur)	For an ETL technology in periods appropriately close to the model horizon, part of the investment costs (VAR_IC) exceed the model horizon. This part of the investment cost is reflected in the calculation of the salvage value variable VAR_OBJSAL.	EQOBSALV.MOD
EQ_OBJINV (r,cur)	The endogenously calculated cost of investments for learning technologies (VAR_IC) needs to be discounted and included in the regional investment cost part of the objective function (EQ_OBJINV) in place of the traditional investment calculation using variable VAR_NCAP.	EQOBJINV.MOD

4.1 EQ_CC(r,t,p)

Description: The Cumulative Capacity Interpolation constraint for an ETL technology.

Purpose and Occurrence: This constraint defines the cumulative investment in capacity for a technology in a period (VAR_CCAP) as the sum over all intervals k of the *continuous* variables VAR_LAMBD(r,t,p,k) that represent cumulative investment in capacity as lying in the k^{th} interval of the piecewise linear approximation of the cumulative cost curve. This constraint links the cumulative capacity investment variable (VAR_CCAP) to the variables VAR_LAMBD. In combination with other ETL constraints, it is fundamental to ensuring the validity of the piecewise linear approximation of the cumulative cost curve.

This equation is generated in each time period for which the ETL technology is available.

Units: Technology capacity units.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as discussed in Section 10.3 of PART I.

Equation

$$EQ\_CC_{r,t,p} \; \forall \left[\left(p \in teg \right) \wedge \left(\left(r,t,p \right) \in rtp \right) \right]$$

Cumulative investment in capacity in the current period.

$$VAR\_CCAP_{r,t,p}$$

$$\{=\}$$

Sum over all intervals k (in the piecewise linear approximation of the cumulative cost curve) of the *continuous* variables VAR_LAMBD in the current period t.

$$\sum_{k} VAR\_LAMBD_{r,t,p,k}$$

4.2 EQ_CLU(r,t,p)

Description: For a key learning ETL technology it defines investment in new capacity (VAR_NCAP) as the weighted sum of investments in new capacity of the attached clustered technologies. The weights used are the numeric values of the CLUSTER parameter.

Purpose and Occurrence: Defines the relationship between investment in new capacity for a key learning ETL technology and investment in new capacity for the associated clustered technologies. This equation is generated in each time period for which the ETL technology is available. It is a key learning technology, that is, it has associated clustered technologies.

Units: Money units, e.g., million 2000 US$, or any other unit in which costs are tracked.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be zero in a feasible solution.

Dual variable: The dual variable (DVR_CLU) of this constraint in the MIP solution is of little interest.

Remarks: Activation of the special clustered learning ETL option occurs automatically if data is included for the CLUSTER parameter.

Equation

$$EQ\_CLU_{r,t,p} \ \forall \begin{bmatrix} (p \in teg) \wedge \left(NTCHTEG_{r,p} > 0\right) \wedge \\ ((r,t,p) \in rtp) \end{bmatrix}$$

Investment in new capacity (for key learning technology p $\in$ teg) in period t.

$$VAR\_NCAP_{r,t,p}$$

$$\{=\}$$

The weighted sum of the investments in new capacity in period t of the clustered technologies p' attached to the key learning technology p $\in$ teg, and whose START period is less than or equal to t. The weights used are the numeric values of the CLUSTER parameter.

$$\sum_{\substack{p'\$\left(CLUSTER_{r,p,p'}>0\right) \wedge \\ (r,t,p' \in rtp)}} (CLUSTER_{r,p,p'} * VAR\_NCAP_{r,t,p'})$$

4.3 EQ_COS(r,t,p)

Description: The Cumulative Cost Interpolation constraint for an ETL technology.

Purpose and Occurrence This constraint defines the interpolated cumulative cost of investment in capacity for a technology in a period (VAR_CCOST) in terms of the binary variables VAR_DELTA and the continuous variables VAR_LAMBD, and the internal model parameters ALPH and BETA, where ALPH and BETA represent the intercepts on the vertical axis and the slopes, respectively, of the line segments in the piecewise linear approximation of the cumulative cost curve. For a more precise definition, see "Equation" below. In combination with other ETL constraints, it is fundamental to ensuring the validity of the piecewise linear approximation of the cumulative cost curve. This equation is generated in each time for which the ETL technology is available.

Units: Money units, e.g., million 2000 US$, or any other unit in which costs are tracked.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as discussed in Section 10.3 of PART I.

Equation

$$EQ\_COS_{r,t,p} \; \forall \left[\left(p \in teg \right) \wedge \left(\left(r,t,p \right) \in rtp \right) \right]$$

Interpolated cumulative cost of investment in capacity in the current period.

$$VAR\_CCOST_{r,t,p}$$

$$\{=\}$$

Sum over all intervals k (in the piecewise linear approximation of the cumulative cost curve) of ALPH times the *binary* variable VAR_DELTA plus BETA times the *continuous* variable VAR_LAMBD, for the current period t, where ALPH and BETA represent the intercepts on the vertical axis and the slopes, respectively, of the k^{th} interval.

$$\sum_{k} \left(ALPH_{k,p} * VAR\_DELTA_{r,t,p,k} + BETA_{k,p} * VAR\_LAMBD_{r,t,p,k} \right)$$

4.4 EQ_CUINV(r,t,p)

Description: The Cumulative Capacity Definition constraint for an ETL technology.

Purpose and Occurrence: This constraint defines the cumulative investment in capacity of a technology in a period (VAR_CCAP) as the initial cumulative capacity (CCAP0) plus the sum of investments in new capacity made up to and including this period. This equation is generated in each time period for which the ETL technology is available.

Units: Technology capacity units.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as mentioned above.

Equation

$$EQ\_CUINV_{r,t,p} \ \forall \left[(p \in teg) \wedge ((r,t,p) \in rtp) \right]$$

Cumulative investment in capacity in the current period.

$$VAR\_CCAP_{r,t,p}$$

$$\{=\}$$

Cumulative investment in capacity at the start of the learning process.

$$CCAP0_{r,p} +$$

Sum of the investments made since the technology is first available.

$$\sum_{u \in rtp_{r,u,p} \wedge u \le t} VAR\_NCAP_{r,u,p}$$

4.5 EQ_DEL(r,t,p)

Description: The constraint for an ETL technology that ensures that in each time period there is exactly one interval k for which the *binary* variable VAR_DELTA(r,t,p,k) has value 1 (with zero values for intervals ≠ k).

Purpose and Occurrence: To ensure that only one of the *binary* variable VAR_DELTA(r,t,p,k) has value 1 for each technology. This constraint, in combination with other ETL constraints, is fundamental to ensuring the validity of the piecewise linear approximation of the cumulative cost curve. This equation is generated in each time period for which the ETL technology is available.

Units: None.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be 1 in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as already mentioned.

Equation

$$EQ\_DEL_{r,t,p} \ \forall \left[\left(p \in teg \right) \wedge \left(\left(r,t,p \right) \in rtp \right) \right]$$

Sum over all intervals k (in the piecewise linear approximation of the cumulative cost curve) of the *binary* variables VAR_DELTA in the current period t.

$$\sum_{k} VAR\_DELTA_{r,t,p,k}$$

$$\{=\} \ 1$$

4.6 EQ_EXPE1(r,t,p,k)

Description: One of two constraints for an ETL technology to improve MIP solution time by reducing the domain of feasibility of the binary variables VAR_DELTA.

Purpose and Occurrence: To improve MIP solution time this constraint takes advantage of the observation that cumulative investment is increasing with time, thus ensuring that if the cumulative investment in period t lies in segment k, then it will not lie in segments k-1, k-2, …, 1 in period t+1. This equation is generated for each ETL technology in each time period, for which the technology is available, and excluding the final period (TLAST), and for each interval k in the piecewise linear approximation of the cumulative cost curve.

Units: None.

Type: *Binding.* The equation is a greater than or equal to (≥) constraint.

Interpretation of the results:

Primal: The level of this constraint must be greater than or equal to zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as already mentioned.

Equation

$$EQ\_EXPE1_{r,t,p,k}\ \forall\left[\left(p \in teg\right) \wedge \left(\left(r,t,p\right) \in rtp\right) \wedge \left(t < TLAST\right)\right]$$

Sum over intervals j ≤ k of binary variables VAR_DELTA(r,t,p,j), for the k^th interval, in period t.

$$\sum_{j \le k} (VAR\_DELTA_{r,t,p,j})$$

$$\{\ge\}$$

Sum over intervals j ≤ k of binary variables VAR_DELTA(r,t,p,j), for the k^th interval, in period t+1.

$$\sum_{j \le k} (VAR\_DELTA_{r,t+1,p,j})$$

4.7 EQ_EXPE2(r,t,p,k)

Description: Second of two constraints for an ETL technology to improve MIP solution time by reducing the domain of feasibility of the binary variables VAR_DELTA. Both constraints rely on the observation that cumulative investment is increasing as time goes on.

Purpose and Occurrence: To improve MIP solution times this constraint is derived from the observation that if cumulative investment in period t lies in segment k, then it must lie in segment k or k+1 or k+2 etc … in period t+1.

This equation is generated for each ETL technology in each time period, for which the technology is available, and excluding the final period (TLAST), and for each interval k in the piecewise linear approximation of the cumulative cost curve.

Units: None.

Type: *Binding.* The equation is a less than or equal to (≤) constraint.

Interpretation of the results:

Primal: The level of this constraint must be less than or equal to zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as already mentioned.

Equation

$$EQ\_EXPE2_{r,t,p,k}\ \forall\left[\left(p \in teg\right) \wedge \left(\left(r,t,p\right) \in rtp\right) \wedge \left(t < TLAST\right)\right]$$

Sum over intervals j ≥ k of binary variables VAR_DELTA(r,t,p,j), for the k^th interval, in period t.

$$\sum_{j \geq k} (VAR\_DELTA_{r,t,p,j})$$

$$\{\leq\}$$

Sum over intervals j ≥ k of binary variables VAR_DELTA(r,t,p,j), for the k^th interval, in period t+1.

$$\sum_{j \geq k} (VAR\_DELTA_{r,t+1,p,j})$$

4.8 EQ_IC1(r,t,p)

Description: The constraint for an ETL technology that defines the portion of the cumulative cost of investment in capacity (VAR_IC) that is incurred in period t, where t = first period of model horizon.

Purpose and Occurrence: To determine the variable VAR_IC which represents the current investment cost incurred in the first period a learning technology is available according to the cumulative investments made in that period. VAR_IC then enters the regional investment cost part of the objective function (EQ_OBJINV) subject to the same discounting that applies to other period t investment costs. This equation is generated for the first period of the model horizon.

Units: Money units, e.g., million 2000 US$, or any other unit in which costs are tracked.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as already mentioned.

Equation

$$EQ\_IC1_{r,t,p} \; \forall (p \in teg) \wedge (t = MIYR\_V1)$$

The portion of the cumulative cost of investment in capacity that is incurred in period t, in this case the first period the technology is available.

$$VAR\_IC_{r,t,p}$$

$$\{=\}$$

The cumulative cost of investment in new capacity in the first period t (t = MIYR_V1).

$$VAR\_CCOST_{r,t,p} -$$

The initial cumulative cost of investment in new capacity for a learning technology.

$$CCOST0$$

4.9 EQ_IC2(r,t,p)

Description: The constraint for an ETL technology that defines the portion of the cumulative cost of investment in capacity that is incurred in each period t other than the first period.

Purpose and Occurrence: To determine the variable VAR_IC which represents the current investment cost incurred in period t according to the cumulative investments made thus far, where VAR_IC then enters the regional investment cost part of the objective function (EQ_OBJINV) subject to the same discounting that applies to other period t investment costs. This equation is generated in each time period other than the first period of the model horizon.

Units: Money units, e.g., million 2000 US$, or any other unit in which costs are tracked.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:

Primal: The level of this constraint must be zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as already mentioned.

Equation

$$EQ\_IC2_{r,t,p} \ \forall (p \in teg) \wedge (t > MIYR\_V1)$$

The portion of the cumulative cost of investment in capacity that is incurred in period t.

$$VAR\_IC_{r,t,p}$$

$$\{=\}$$

The cumulative cost of investment in new capacity as of period t.

$$VAR\_CCOST_{r,t,p} -$$

The cumulative cost of investment in new capacity as of the previous period t-1.

$$VAR\_CCOST_{r,t-1,p}$$

4.10 EQ_LA1(r,t,p,k)

Description: The constraint for an ETL technology that sets a lower bound on the continuous variable VAR_LAMBD(r,t,p,k).

Purpose and Occurrence: To set the lower bound for VAR_LAMBD(r,t,p,k) to CCAPK(r,k-1,p) * VAR_DELTA, where CCAPK(r,k-1,p) is the left hand end of the k^{th} interval and VAR_DELTA = VAR_DELTA(r,t,p,k) is the binary variable associated with VAR_LAMBD(r,t,p,k). If binary variable VAR_DELTA = 1, the effect is to set a lower bound on variable VAR_LAMBD(r,t,p,k) of CCAPK(r,k-1,p), whereas if VAR_DELTA = 0 the effect is to set a lower bound of 0. This constraint, in combination with other ETL constraints, is fundamental to ensuring the validity of the piecewise linear approximation of the cumulative cost curve.

This equation is generated in each time period, for which the ETL technology is available, and for each interval k in the piecewise linear approximation of the cumulative cost curve.

Units: Technology capacity units.

Type: *Binding.* The equation is a greater than or equal to (≥) constraint.

Interpretation of the results:

Primal: The level of this constraint must be greater than or equal to zero in a feasible solution.

Dual variable: The dual variables of mixed integer problems have limited usefulness, as already mentioned.

Equation

$$EQ\_LA1_{r,t,p,k}\ \forall\left[\left(p \in teg\right) \wedge \left(\left(r,t,p\right) \in rtp\right)\right]$$

Portion of the cumulative investment in capacity that lies in the k^{th} interval (of the piecewise linear approximation of the cumulative cost curve), in the current period.

$$VAR\_LAMBD_{r,t,p,k}$$

$$\left\{\geq\right\}$$

Left hand end of the k^{th} interval (CCAPK(r,k-1,p)) times binary variable VAR_DELTA(r,t,p,k), in the current period.

$$CCAPK_{r,k-1,p} * VAR\_DELTA_{r,t,p,k}$$

4.11 EQ_LA2(r,t,p,k)

Description: The constraint for an ETL technology that sets an upper bound on the continuous variable VAR_LAMBD(r,t,p,k).

Purpose and Occurrence: To set the upper bound of VAR_LAMBD(r,t,p,k) to CCAPK(r,k,p) * VAR_DELTA, where CCAPK(r,k,p) is the right hand end of the k^{th} interval and VAR_DELTA = VAR_DELTA(r,t,p,k) is the binary variable associated with VAR_LAMBD(r,t,p,k). If binary variable VAR_DELTA = 1, the effect is to set an upper bound on variable VAR_LAMBD(r,t,p,k) of CCAPK(r,k,p), whereas if VAR_DELTA = 0 the effect is to set an upper bound of 0. This constraint, in combination with other ETL constraints, is fundamental to ensuring the validity of the piecewise linear approximation of the cumulative cost curve.

This equation is generated in each time period, for which the ETL technology is available, and for each interval k in the piecewise linear approximation of the cumulative cost curve.

Units: Technology capacity units.

Type: *Binding.* The equation is a less than or equal to (≤) constraint.

Interpretation of the results:

Primal: The level of this constraint must be less than or equal to zero in a feasible solution.

Dual variable: The dual variable (DVR_LA2) of this constraint in the MIP solution is of little interest.

Equation

$$MR\_LA2_{r,t,p,k} \ \forall\left[\left(p \in teg\right) \wedge \left(\left(r,t,p\right) \in rtp\right)\right]$$

Portion of the cumulative investment in capacity that lies in the k^{th} interval (of the piecewise linear approximation of the cumulative cost curve), in the current period.

$$VAR\_LAMBD_{r,t,p,k}$$

$$\{\leq\}$$

Right hand end of the k^{th} interval (CCAPK(r,k,p)) times binary variable R_DELTA(r,t,p,k), in the current period.

$$CCAPK_{r,k,p} * VAR\_DELTA_{r,t,p,k}$$

4.12 EQ_MRCLU(r,t,p)

Description: For a key learning ETL technology it defines investment in new capacity (VAR_NCAP) as the weighted sum of investments in new capacity of the attached clustered technologies in multiple regions. The weights used are the numeric values of the TL_MRCLUST parameter.

Purpose and Occurrence: Defines the relationship between investment in new capacity for a key learning ETL technology and investment in new capacity for the associated clustered technologies. This equation is generated in each time period for which the ETL technology is available. It is a key learning technology, that is, it has associated clustered technologies, possibly in multiple regions.

Units: Money units, e.g., million 2010 US$, or any other unit in which costs are tracked.

Type: *Binding.* The equation is an equality (=) constraint.

Interpretation of the results:
Primal: The level of this constraint must be zero in a feasible solution.
Dual variable: The dual variable of this constraint in the MIP solution is of little interest.

Remarks: Activation of the special clustered learning ETL option occurs automatically if data is included for the TL_MRCLUST parameter.

Equation

$$EQ\_MRCLU_{r,t,p} \ \forall \begin{bmatrix} (p \in teg) \wedge (TL\_RP\_KC_{r,p}) \wedge \\ ((r,t,p) \in rtp) \end{bmatrix}$$

Investment in new capacity (for key learning technology p ∈ teg) in period t.

$$VAR\_NCAP_{r,t,p}$$

$$\{=\}$$

The weighted sum of the investments in new capacity in period t of the clustered technologies p' attached to the key learning technology p ∈ teg, and whose START period is less than or equal to t. The weights used are the numeric values of the CLUSTER parameter.

$$\sum_{(reg,t,prc \in \mathbf{rtp})} \left(TL\_MRCLUST_{reg,p,prc} \times VAR\_NCAP_{reg,t,prc} \right)$$

4.13 EQ_OBJSAL(r,cur)

Description: Regional salvage value part of objective function adjusted to include the salvage value of endogenously determined investments (VAR_IC) in learning technologies. A salvage value for a learning technology investment exists when the technical lifetime of the investment exceeds the model horizon.

Purpose and Ocurrence: The objective function part calculating the salvage value is changed (for learning technologies only) by replacing the traditional calculation of the salvage value of investments with one based on the investment costs of learning technologies (VAR_IC).

Units: Money units, e.g., million 2000 US$, or any other unit in which costs are tracked.

Type: *Binding.* The equation is an equality (=) constraint.

Equation

$$EQ\_OBJSAL_{r,cur}$$

All the basic objective function term for calculating the salvage value (section 5.2.8)

• • •

The calculated salvage value associated with the ETL technologies. The internally derived parameter coefficient OBJSIC describing the portion of the investment costs that has to be salvaged. It takes into account the discounting of the salvage value.

$$+\sum_{t,p \in teg}\left[OBJSIC_{r,t,p} * VAR\_IC_{r,t,p}\right]$$

4.14 EQ_OBJINV(r,cur)

- see EQ_OBJINV in section 5.2.2 for a general description without ETL

Description: Regional investment cost part of objective function adjusted to include the endogenously determined investment cost (VAR_IC) for new investments in learning technologies.

Purpose and Occurrence: The objective function part calculating the investment costs is changed (for learning technologies only) by replacing the traditional calculation of discounted cost of investments in new capacity with that of the endogenously determined value. This equation is generated for each region where the learning investment costs occur in each time period beginning from the period, for which the ETL technology is available.

Equation

$$EQ\_OBJINV_{r,cur}$$

All the basic objective function terms for investment costs (section 5.2.2)

• • •

The calculated investments costs associated with the ETL technologies.

$$+\sum_{t,p \in teg}\left[DISC_{r,t,p} * VAR\_IC_{r,t,p}\right]$$

Energy Technology Systems Analysis Programme

http://www.iea-etsap.org/web/Documentation.asp

Documentation for the TIMES Model

PART III

July 2016

Authors:

Gary Goldstein
Amit Kanudia
Antti Lehtilä
Uwe Remme
Evelyn Wright

General Introduction to the TIMES Documentation

This documentation is composed of five Parts.

Part I provides a general description of the TIMES paradigm, with emphasis on the model's general structure and its economic significance. Part I also includes a simplified mathematical formulation of TIMES, a chapter comparing it to the MARKAL model, pointing to similarities and differences, and chapters describing new model options.

Part II constitutes a comprehensive reference manual intended for the technically minded modeler or programmer looking for an in-depth understanding of the complete model details, in particular the relationship between the input data and the model mathematics, or contemplating making changes to the model's equations. Part II includes a full description of the sets, attributes, variables, and equations of the TIMES model.

Part III describes the organization of the TIMES modeling environment and the GAMS control statements required to run the TIMES model. GAMS is a modeling language that translates a TIMES database into the Linear Programming matrix, and then submits this LP to an optimizer and generates the result files. Part III describes how the routines comprising the TIMES source code guide the model through compilation, execution, solve, and reporting; the files produced by the run process and their use; and the various switches that control the execution of the TIMES code according to the model instance, formulation options, and run options selected by the user. It also includes a section on identifying and resolving errors that may occur during the run process.

Part IV provides a step-by-step introduction to building a TIMES model in the VEDA-Front End (VEDA-FE) model management software. It first offers an orientation to the basic features of VEDA-FE, including software layout, data files and tables, and model management features. It then describes in detail twelve Demo models (available for download from the ETSAP website) that progressively introduce VEDA-TIMES principles and modeling techniques.

Part V describes the VEDA Back-End (VEDA-BE) software, which is widely used for analyzing results from TIMES models. It provides a complete guide to using VEDA-BE, including how to get started, import model results, create and view tables, create and modify user sets, and step through results in the model Reference Energy System. It also describes advanced features and provides suggestions for best practices.

PART III: THE OPERATION OF THE TIMES CODE

TABLE OF CONTENTS FOR PART III

List of Figures

List of Tables

1 Introduction

1.1 Summary of components

The TIMES model environment under VEDA is depicted in Figure 1. For ANSWER the underlying model management flow is very similar, with the addition of a <Case>.ANT file being dumped by the TIMES GAMS report writer for importing of the model results into ANSWER, if desired, though model TIMES users tend to rely on the extra power brought to bear by VEDA-BE.

It is composed of five distinct components described below.

- **The TIMES Model Generator** (as well as **MARKAL**[1]) comprises the GAMS source code that processes each dataset (the model) and generates a matrix with all the coefficients that specify the economic equilibrium model of the energy system as a mathematical programming problem. The model generator also post-processes the optimization to prepare results that are suitable to be read by the model management systems (and other tools). It is shown in Figure 1 labelled as TIMES. The TIMES model generator is available from ETSAP by executing a Letter of Agreement[2].

- **The model** is a set of data files (spreadsheets, databases, simple ASCII files), which fully describes an energy system (technologies, commodities, resources, and demands for energy services) in a format compatible with an associated model management shell. Each set of files comprises one model (perhaps consisting of a number of regional models) and is "owned" by the developer(s). It is shown in Figure 1 as the Data and Assumptions box in the upper left. Instances of global models include the IEA's Energy Technology Perspectives (ETP[3]), the TIMES Integrated Assessment Models (TIAM[4]), and that of the European Fusion Development Agreement (EFDA[5]). Large multi-region models exist in the form of the Pan-European TIMES (PET[6]) model covering EU27 + Norway, Switzerland and Iceland, and the Framework for Analysis of Climate-Energy-Technology Systems (FACETS[7]) for the US. Finally, there are numerous national models assembled by the ETSAP Partner institutions[8], and various national, regional, and municipal models developed by other institutions[9].

[1] MARKAL is the legacy ETSAP model generator that has been superseded by its advanced TIMES successor.

[2] http://www.iea-etsap.org/web/AcquiringETSAP_Tools.asp

[3] http://www.iea.org/etp/etpmodel/

[4] http://www.iea-etsap.org/web/applicationGlobal.asp

[5] https://www.euro-fusion.org/wpcms/wp-content/uploads/2015/02/EFDA-TIMES_Global.pdf

[6] http://www.kanors-emr.org/Website/Models/PET/Mod_PET.asp

[7] http://facets-model.com/overview/

[8] http://iea-etsap.org/web/applicationNational.asp

[9] http://iea-etsap.org/web/Applications.asp

- **A Model Management "shell"** is a user interface that oversees all aspects of working with a model, including handling the input data, invoking the Model Generator, and examining the results. It is shown in Figure 1 labelled VEDA-FE and VEDA-BE for the parts handling the input data and model results respectively. It thereby makes practical the use of robust models (theoretically, simple models can be handled by means of ASCII file editors, if desired). The first shell, MUSS, was developed in 1990 by DecisionWare Inc. for use with MARKAL (and is no longer available). Two shells currently in use for TIMES are ANSWER, originally developed by ABARE and subsequently the property of Noble-Soft Systems Pty Ltd[10], and VEDA, developed by KanORS-EMR. Both ANSWER and VEDA handle MARKAL as well as TIMES. Both shells were partly developed using ETSAP resources, along with substantial contributions of the developers and other projects employing the systems. Note that as shown in Figure 1, VEDA-FE interacts with GAMS by means of the *.RUN/DD files and GAMS interacts with VEDA-BE by processing the GDX file to produce the run VD* files. VEDA-BE can write to XLS or other file types. See Sections IV and V for a description of VEDA, and the separate ANSWER documentation respectively.

- **The General Algebraic Modeling System (GAMS)**[11] is the computer programming language in which the MARKAL and TIMES Model Generators are written. GAMS is a two-pass language (first compiling the input data and source code, then executing for the data provided) designed explicitly to facilitate the formulation of complex mathematically programming models. GAMS integrates smoothly with various solvers to generate the mathematic programming problem and seamlessly pass it to the solvers for optimization, then post-process the optimization to produce the TIMES results report for the "shells." It is shown in Figure 1 GAMS together with the final component, Solvers. During a run, GAMS produces a LST file with an echo of the model run steps and solution. The LOG file in the figure is actually produced by TIMES, listing the quality assurance checks. GAMS is the property of GAMS Development Corporation, Washington D.C. Information on GAMS may be found at www.gams.com. More specific GAMS - ETSAP information can be obtained from the ETSAP Liaison Officer, Gary Goldstein.

- **A solver** is a software package integrated with GAMS which solves the mathematical programming problem produced by the Model Generator for a particular instance of the

[10] Note that as of December 2016 ANSWER will no longer be actively developed and only limited support will be provided, so although mentioned here in Part III, users should carefully consider their longer-term needs when considering using ANSWER as their TIMES model management platform going forward.

[11] Anthony Brooke, David Kendrick, Alexander Meeraus, and Ramesh Raman, GAMS – A User's Guide, December 1998, www.gams.com.

TIMES model. Solvers are discussed further in Section 1.4. More information on solvers may be found at www.gams.com.

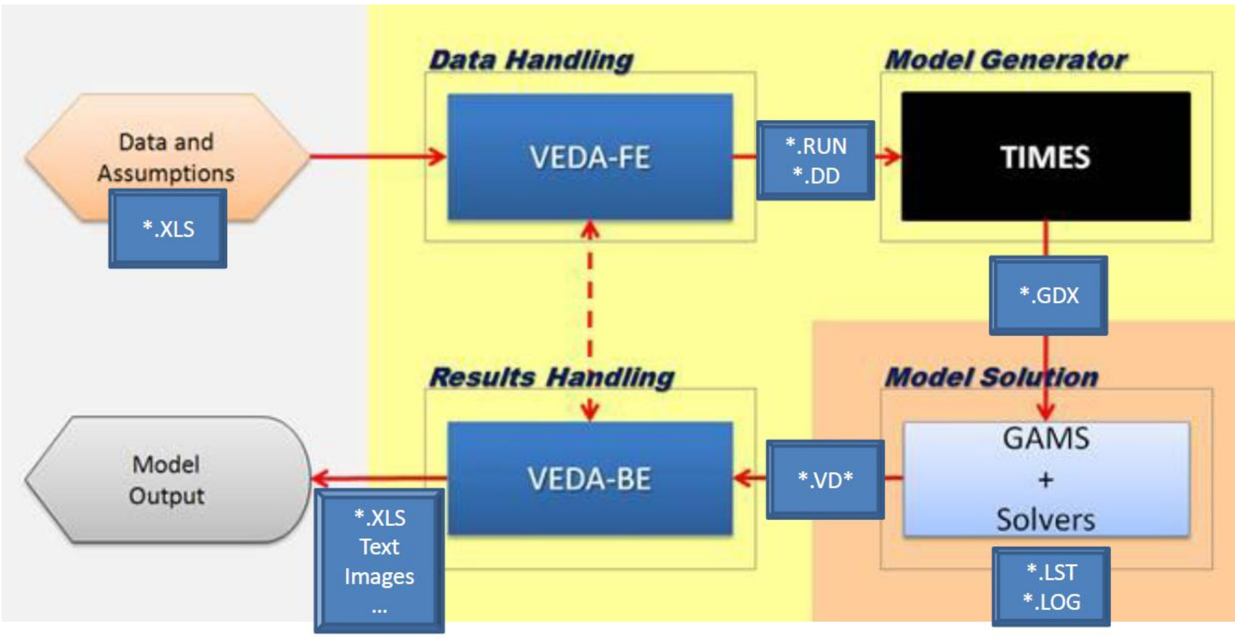


Figure 1: Components of the TIMES Modeling Platform Under VEDA

The rest of this Part describes in more detail how the computer environment is organized and operates to make working with TIMES viable and effective.

1.2 Minimum computer requirements

The minimum basic software requirements consist of the GAMS modeling language and an associated solver, a model management "shell" (which while technically optional has been used for every application of TIMES to date) comprised of either VEDA (Front-End (FE) for handling input data and Back-End (BE) for processing results) or ANSWER (where ANSWER users often also employ VEDA-BE for results). The "shells" are Windows based Visual-Basic turnkey applications that are distributed as part of a TIMES installation package, see Parts IV and V for VEDA and the separate ANSWER documentation for a discussion on the model management systems.

In terms of the Windows operating system, any version (32 or 64 bit) from Version 7 on is supported by both ANSWER and VEDA. Both ANSWER and VEDA require that a properly licensed version of Microsoft Excel be installed on the computer. Both shells may be run on Apple computers within a Windows emulator; however, they are not supported on Linux/Unix platforms.

For hardware, a "high-end" personal computer with a minimum of 8GB RAM (16GB or more for larger models), ideally a multi-core/CPU processor (dual quad core for large models), and up to 250GB (depending upon the size of the model and studies to be undertaken) of hard disk storage for the modeling is recommended.

1.3 General layout of the software

Each of the components mentioned above – GAMS, VEDA, and ANSWER – reside in their own Windows folder of the **ROOT** on whatever drive the user wishes. When installing the software, the user is strongly encouraged to follow this "install in the root" recommendation, as the complex nature of the software systems and their interdependencies are most smoothly handled when the system is setup in this manner (rather than installing under Program Files for example).

The various components discussed above "talk" with each other primarily by means of ASCII text files deposited in common locations (folders) and passed between said components. The specific folder layout for each component is discussed below and later in the Section a look at the specific files involved with the inter-component communication is provided. This handshaking is virtually seamless from the users' perspective, as long as all the component paths are properly identified for each component.

For GAMS, the system is self-contained in a \GAMS\<os>\<version> system folder (if installed in the default location, as recommended) and is connected to VEDA-FE and ANSWER through the Windows Path Environment Variable. This GAMS path is either set during installation automatically (by requesting Advanced Installation Mode and requesting Add GAMS Directory to Path Environment Variable) or manually via the Windows Control Panel. Full (simple) instructions are provided for installing and properly configuring GAMS for use with TIMES with the software distribution notification email and are summarized in Section 1.4 below.

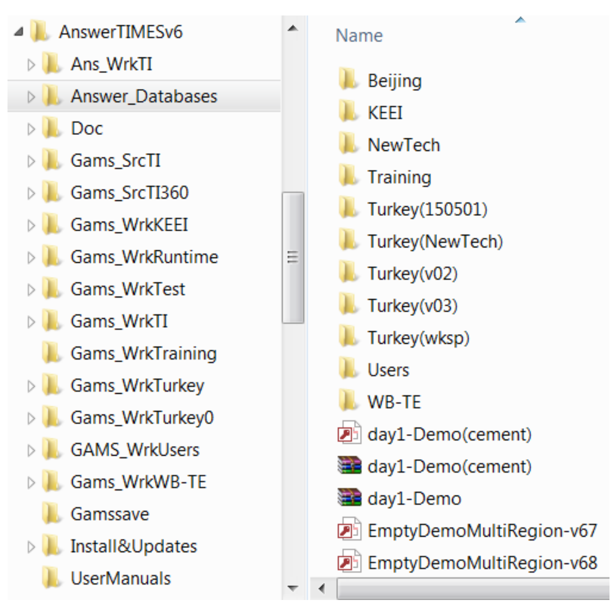


Figure 2: Layout of the ANSWER Folders

For ANSWER, the core of the system must reside in a single folder \AnswerTIMESv6 (encouraged to be right off the root). A full description of the folder structure that ANSWER employs may be found in the separate ANSWER documentation, with the basic layout shown in Figure 2 below. From the perspective of connecting ANSWER with GAMS and VEDA-BE (if used) the key subfolders the user needs to be aware of are the GAMS_SrcTI and GAMS_WrkTI default TIMES source code and model run folders respectively. Upon initiating a model run, ANSWER needs to inform GAMS where the TIMES model source code is, that being GAMS_SrcTI (or any variant the user chooses to setup). For the model results to find their way to VEDA-BE, it must be informed of the model run folder, that being GAMS_WrkTI (or any variant the user chooses to set up, say for different projects), through the VEDA-BE Import Results/Manage Input File Location operation. The location of these folders for each model is set within ANSWER, through the Tools/File Locations option. (In the example shown in Figure 2, several GAMS_Wrk<model> folders have been created so that different models (or projects) are run in distinct folders.)The other folder the user will interact with is the Answer_Databases where by default the user's ANSWER TIMES database (MDB) and usually Excel input templates would reside. In this regard the user is encouraged to make subfolders under Answer_Databases (or any other location they wish) for each of their models or project, as shown in Figure 2.

For VEDA, the core of the system must reside in a single folder \VEDA (encouraged to be right off the root), with the basic required folder structure shown in Figure 3. From the perspective of connecting VEDA-FE with GAMS, the key subfolders the user needs to be particularly aware of are the GAMS_SrcTIMESv### and GAMS_WrkTIMES (or other run folders for each project if desired), the default TIMES source code and model run folders respectively. Both reside in the \VEDA\VEDA_FE folder. Upon initiating a model run, VEDA-FE needs to inform GAMS where the TIMES model source code is, that being GAMS_SrcTIMES### (or any variant the user chooses to setup). For the model results to find their way to VEDA-BE it must be informed of the model run folder, that being GAMS_WrkTIMES (or any variant the user chooses to set up, say for different projects or model instances), through the VEDA-BE Import Results/Manage Input File Location operation. The location of these folders for each model is set within VEDA-FE, through Tools/User Options settings.

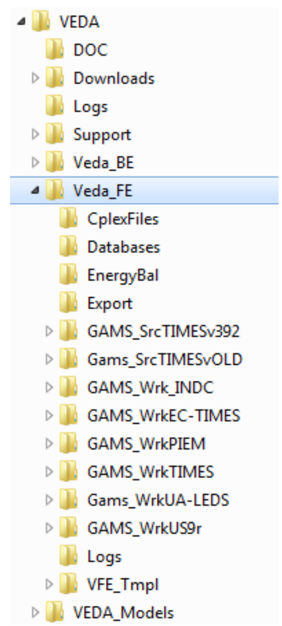


Figure 3: Layout of the VEDA-FE Folders

To complete the inter-connection picture between components of VEDA, VEDA-FE maintains each model instance in the VEDA_Models folder where the user assembles the model input Excel templates. The other folder the user will need to be aware of is the VEDA_BE\Databases\<project> where the user's the VEDA-BE results databases reside. In order for VEDA-FE to use Sets defined in VEDA-BE for user constraint and/or scenario specifications, the former must be pointed to the latter – by means of clicking on the VEDA-BE database reference up at the top of the VEDA-FE application window.

1.4 Software installation

This section provides an overview of the installation process for GAMS, which is required for all TIMES installations[12]

GAMS employs "soft" licensing. That is, each system is licensed for a certain Windows PC or Server or Linux and requested solvers to a particular institution for a requested number of users. The license is not to be shared outside the authorized institution and the number of users is to be adhered to – all based upon trust (and the very active GAMS and MARKAL/TIMES user community).

Note that GAMS provides two kinds of licenses for working with TIMES, the conventional license which provides the user with the actual TIMES GAMS source code, and a Runtime license where the source code is precompiled and therefore may not be changed. The Runtime license ONLY permits GAMS to be used in conjunction with TIMES. That is, no other GAMS models may be run using a ETSAP TIMES Runtime license. The Runtime license is sold at half the price of a corresponding full license. To obtain GAMS for use with MARKAL/TIMES contact the ETSAP Liaison Officer, Gary Goldstein.

The basic procedure for installing GAMS is:

1. Copy your GAMS license file, GAMSLICE.txt, provided as part of the licensing process by the Liaison Officer, someplace on your computer.
2. Head to http://www.gams.com/download/ and select the Windows download option for either Win-64/32, as appropriate.
3. Run Setup by clicking on it in Windows Explorer
    a) Check "Use advanced installation mode" at the bottom of the GAMS Setup form.
    b) Let GAMS get installed into the default folder (\GAMS\<Win#>\<ver>.
    c) Check the Add GAMS directory to PATH environment variable.
    d) Have the GAMSLICE.TXT copied from wherever it currently resides.

If you are using a non-default solver (e.g., CPLEX is the default for LP and MIP models and CONOPT for NLP) then there is one further step that must be carried out to complete the setup procedure:

4. If using a non-default solver, upon completion use Windows Explorer to go to the GAMS system folder and run the GAMSINST program to set the default solver for each type to the solvers supported by your license GAMSLICE file by entering the associated number in the list and hitting return or just hitting return (if your solver is the default or not listed).

Which solver to use is a function of the TIMES model variant to be solved and the solver(s) purchased by the user with GAMS. Basically the solvers used for TIMES fall into three

[12] Instructions for installing VEDA are available at http://support.kanors-emr.org/VEDAInstallation. Instructions for ANSWER can be obtained from Noble-Soft Systems.

categories, linear (LP), mixed integer (MIP) and non-linear (NLP), where Table 1 provides a partial list of the GAMS solvers generally used with TIMES for each main model variant. For a complete list of the solvers available refer to the GAMS website. The various Model Instances mentioned in the table are discussed further in Section 3.6, as well as in Parts I and II.

Table 1: TIMES Model Variants and GAMS Solvers

Model Instance	Nature of Model	Viable Solvers
Basic TIMES (including Elastic Demand, Climate, Stochastic, etc.)	Linear (LP)	For full-blown TIMES models power solvers are recommended (CPLEX/XPRESS/GUROBI) For modest size models MINOS or the “free” public solvers may suffice
Discrete Investment, Discrete Retirement, Discrete Dispatching and Endogenous Technology Learning Extensions	Mixed Integer (MIP)	Power solvers are recommended (CPLEX/XPRESS/GUROBI) For modest size models the “free” public CBC solvers may suffice
MACRO (integrated or MSA/CSA), Damage (NLP option)	Non-linear (NLP)	CONOPT recommended, MINOS an option (but no longer being developed)

1.5 Organization of Part III

Section 2 lays out more specifically how the various components of the TIMES modeling platform interact and accomplish their tasks. Following an overview of the run process, the routines comprising the TIMES source code are described, discussing how they guide the model through compilation, execution, solve, and reporting. Section 2.3 then describes the various files produced by the run process and their use. Finally, Section 2.4 discusses identifying and resolving errors that may occur during the run process. Section 3 then details the various switches that control the execution of the TIMES code.

2 The TIMES source code and the model run process

As discussed in the previous section, the heart of TIMES is embodied in the GAMS source code, comprised of the matrix generator that digests the input data prepared from ANSWER or VEDA-FE and prepares the mathematical representation of the model instance, an optimizer to solve the resulting mathematical programming problem, and a report writer that post-processes the solution and prepares results files for ANSWER and VEDA-BE. It is this collection of GAMS source code routines that correspond to the TIMES model, where each TIMES model run proceeds through the appropriate path in the source code based upon the user specified runtime switches, described in Section 3, and the provided input data.

For the most part, this process is seamless to the user, as the model management shells extract the scenario data and prepare ASCII text files in the layout required by GAMS, set up the top level GAMS control file, and initiate the model run (in a Command Prompt box). GAMS then compiles the source and data, constructs the model, invokes the solvers, and dumps the model results for importing back into the model management environment. However, knowledge of the run process and the files produced along the way can be helpful in diagnosing model errors (e.g., a division by zero may necessitate turning on the source code listing ($ONLISTING/$OFFLIST at the top/bottom of the routine where the error is reported) to determine which parameter is causing the problem) or if a user is considering modifying the model formulation to, say, add a new kind of constraint (note that any such undertaking should be closely coordinated with ETSAP).

2.1 Overview of the model run process

Once a model is readied, a run can be initiated from the model management systems by means of the Case Manager in VEDA-FE or the Run Model option on the ANSWER Home screen. In both systems the user assembles the list of scenarios to comprise the model run, taking care to ensure that the order of the scenarios is such that all RES components are first declared (that is their item name and set membership specified) and then assigned data.

In the model management shells, the user can also adjust the TIMES model variant, run control switches, and solver options. For VEDA-FE this is done through the Case Manager, via the Control Panel, RUNFile template, and solver settings. Figure 4 shows an example of the VEDA-FE Control Panel. See Part IV for more on the use of the Case Manager in VEDA-FE.

In ANSWER, the Run Model button brings up the Run Model Form, from which the model variant and most run control switches can be set. (The others need to be set using the Edit GAMS Control Options feature.) However the <Solver>.OPT file needs to be handled manually outside of ANSWER. See the separate ANSWER documentation for more details on these ANSWER facilities.

When a model run is initiated, three kinds of files are created by VEDA and ANSWER. The first is a Windows command script file VTRUN/ANSRUN.CMD (for VEDA/ANSWER respectively), which just identifies the run name, indicates where the source code resides, and perhaps any restart (see Section 3.8), and then calls the VEDA/ANSWER driver command script (VT_GAMS/ANS_RUN.CMD). The second is the top-level GAMS command file <Case>.RUN/GEN (for VEDA/ANSWER), which is passed to GAMS to initiate and control the model run. It sets the model variant, identifies the Milestone (run) years, lists the scenario data files (DD/DDS) to include, and invokes the main GAMS routine to have the model actually assembled mathematically, solved, and reported upon. It is discussed further in Section 2.3.1. The third group of files comprise the data dictionary <scenario>.DD/DDS file(s), which contain the user input sets and parameters in the format required by GAMS to fully describe the energy system to be analyzed.

GAMS is a two pass language, first compiling and then executing. In the first pass, GAMS reads the input data prepared by ANSWER or VEDA-FE, and then proceeds to compile the data as well as the actual TIMES source code to ready it for execution (unless a Runtime license is employed in which case only the data is compiled).

In the second pass, GAMS then proceeds to execute the complied data and code to declare the equations and variables that are to make up this particular TIMES incarnation and generate the appropriate coefficients for matrix intersection, that is the multiplier for the individual variables comprising each equation. With the matrix assembled GAMS then turns over the problem to the solver.

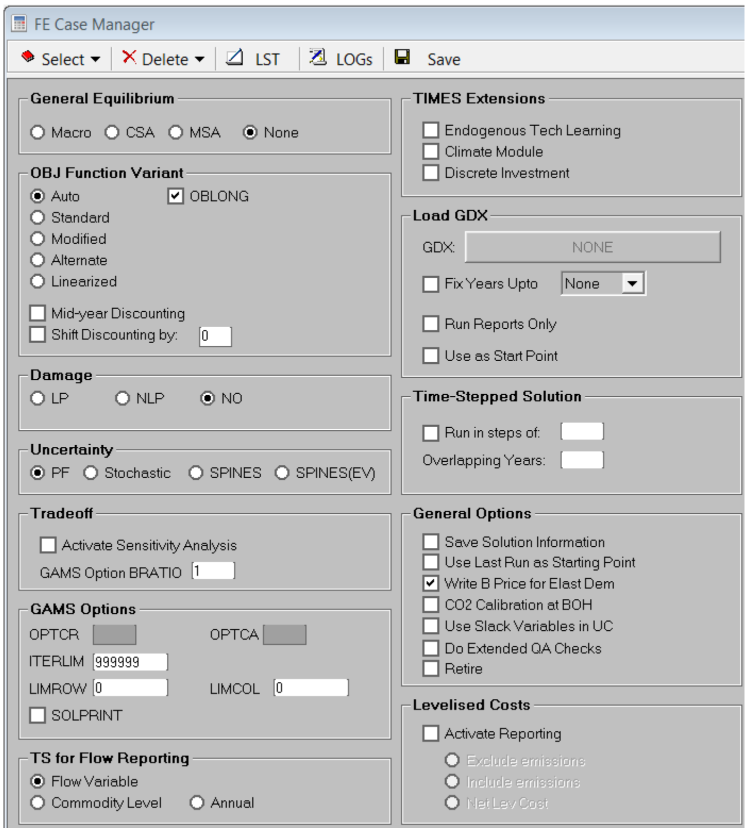


Figure 4: VEDA-FE Case Manager Control Form

As a result of a model run a listing file (<Case>.LST), and a <case>.GDX file (GAMS dynamic data exchange file with all the model data and results) are created. The <Case>.LST file may contain compilation calls and execution path through the code, an echo print of the GAMS source code and the input data, a listing of the concrete model equations and variables, error messages, model statistics, model status, and solution dump. The amount of information displayed in the listing file can be adjusted by the user through GAMS options in the <Case>.RUN file.

The <Case>.GDX file is an internal GAMS file. It is processed according to the information provided in the TIMES2VEDA.VDD to create results input files for the VEDA-BE software to analyze the model results in the <case>.VD* text files. A dump of the solution results is also done to the <case>.ANT file for importing into ANSWER, if desired. At this point, model results can be imported into VEDA-BE and ANSWER respectively for post-process and analysis. More information on VEDA-BE and ANSWER results processing can be found in Part V and the separate ANSWER documentation respectively.

In addition to these output files, TIMES may create a file called QA_CHECK.LOG to inform the user of possible errors or inconsistencies in the model formulation. The QA_CHECK file should be examined by the user on a regular basis to make sure no “surprises” have crept into a model. The content and use of each of these files is discussed further in Section 2.3.

For the ETSAP Runtime GAMS license, which does not allow for adjustments to the TIMES source code by users (which in general is not encouraged anyway), a special TIMES.g00 file is used that contains the declaration of each variable and equation that is part of the model definition, thereby initializing the basic model structure.

2.2 The TIMES source code

The TIMES model generator is comprised of a host of GAMS source code routines, which are simple text files that reside in the user's \VEDA\VEDA-FE\GAMS_SrcTIMESv### folder, as discussed in Section 1.3 (or \AnswerTIMESv6\GAMS_SrcTI). Careful naming conventions are employed for all the source code routines. These conventions are characterized, for the most part, by prefixes and extensions corresponding to collections of files handling each aspect of the code (e.g., set bounds, prepare coefficients, specify equations), as summarized in Table 2.

Table 2: TIMES Routines Naming Conventions

Type	Nature of the Routine
Prefix	
ans	ANSWER TIMES specific pre-processor code
bnd	set bounds on model variables
cal	calculations performed in support of the preprocessor and report writer
coef	prepare the actual matrix intersection coefficients
eq	equations specification (that is the actual assembling of the coefficients of the matrix)
err	Error trapping and handling
fil	handles the fundamental interpolation/extrapolation/normalization of the original input data
init	initialize all sets and parameters potentially involved in assembling a TIMES model
main	Top level routines according to the model variant to be solved
mod	the declaration of the equations and variables for each model variant

Type	Nature of the Routine
timeslices	Handles the aggregation / inheritance of timeslices to the various levels
pp	preprocess responsible for preparing the TIMES internal parameters by assembling, interpolating, normalizing, and processing the input data to prepare the data structures needed to produce the model coefficients
put	components of the results report writer that actually writes the output lines
qa	Quality assurance checking and reporting
rpt	main reporting components performing the calculations needed and assembling the relevant parameters from the model results
solve	manage the actual call to solve the model (that is the call to invoke the optimizer)
uc	handles the user constraints
Extension	
ANS	ANSWER specific code
CLI	climate module routines
CMD	Windows command scripts to invoke GAMS/GDX2VEDA in order to solve and afterwards dump the model results
DEF	setting of defaults
DSC	discrete (lumpy) investment routines
ETL	endogenous technology learning routines
GMS	lower level GAMS routines to perform interpolation, apply shaping of input parameters, etc.
RUN/GEN	VEDA-FE/ANSWER specific GAMS TIMES command templates for dynamic substitution of the switches and parameters needed at run submission to identify the model variant and other options that will guide the current model run
IER	routines and extensions prepared by the University of Stuttgart (Institute for the Rational Use of Energy, IER) (e.g., for more advanced modeling of CHPs)
LIN	routines related to the alternative objective formulations
MOD	core TIMES routines preparing the actual model
MSA	code related to the MSA implementation enabling TIMES and MACRO to iterate in harmony
RED	reduction algorithm routines
RPT	report writer routines
STC	code related to stochastics
STP	code related to time-stepped or partially fixed-horizon solution
TM	the core TIMES MACRO code
VDA	routines related to new TIMES features implemented under the VDA extension

Note that these don't cover every single routine in the TIMES source code folder, but do cover most all of the core routines involved in the construction and reporting of the model. They guide the steps of the run process as follows:

- **GAMS Compile:** As mentioned above, GAMS operates as a two-phase compile then execute system. As such it first reads and assembles all the control, data, and code files into a ready executable; substituting user and/or shell provided values for all GAMS environment switches and subroutine parameter references (the %EnvVar% and %Param% references in the source code) that determine the path through the code for the requested model instance and options desired. If there are inconsistencies in input data they may result in compile-time errors (e.g., $170 for a domain definition error), causing a run to stop. See Section 2.4 for more on identifying the source of such errors.
- **Initialization:** Upon completion of the compile step, all possible GAMS sets and parameters of the TIMES model generator are declared and initialized, then established for this instance of the model from the user's data dictionary file(s) (<Case>.DD[13]). Model units are also initialized using the UNITS.DEF file, which contains the short names for the most common sets of units that are normally used in TIMES models, and which can be adjusted by the user.
- **Execution:** After the run has been prepared, the maindrv.mod routine controls all the remaining tasks of the model run. The basic steps are as follows.
    - **Pre-processing:** One major task is the pre-processing of the model input data. During pre-processing control sets defining the valid domain of parameters, equations and variables are generated (e.g., for which periods each process is available, at what timeslice level (after inheritance) is each commodity tracked and does each process operate), input parameters are inter-/extrapolated, and time-slice specific input parameters are inherited/aggregated to the correct timeslice level as required by the model generator.
    - **Preparation of coefficients:** A core activity of the model generator is the proper derivation of the actual coefficients used in the model equations. In some cases coefficients correspond directly to input data (e.g., FLO_SHAR to the flow variables), but in other cases they must be transformed. For example, the investment

[13] For simplicity, it has been assumed in this description that the name of the <case>.run/gen (for VEDA-FE/ANSWER respectively) file and the *.dd files are the same (<case_name>). The names of the two files can be different, and usually are with BASE.dd the main dataset with non-Base scenarios included in a run having <scenario>.dd/dds names (for VEDA-FE/ANSWER respectively). The listing file generated by GAMS always has the same name of the <case>.run/gen file. The name of the gdx files can be chosen by the user on the command line calling GAMS (e.g. gams mymodel.run gdx = myresults will result in a file called myresults.gdx), however, out of VEDA-FE/ANSWER the files are <case>.gdx.

cost (NCAP_COST) must be annualized, spread for the economic lifetime, and discounted before being applied to the investment variable (VAR_NCAP) in the objective function (EQ_OBJ), and based upon the technical lifetime the coefficients in the capacity transfer constraint (EQ_CPT**)** are determined to make sure that new investment are accounted for and retired appropriately.

- **Generation of model equations:** Once all the coefficients are prepared, the file eqmain.mod controls the generation of the model equations. It calls the individual GAMS routines responsible for the actual generation of the equations of this particular instance of the TIMES model. The generation of the equations is controlled by sets, parameters, and switches carefully assembled by the pre-processor to ensure that no superfluous equations or matrix intersections are generated.
- **Setting variable bounds:** The task of applying bounds to the model variables corresponding to user input parameters is handled by the bndmain.mod file. In some cases it is not appropriate to apply bounds directly to individual variables, but instead applying a bound may require the generation of an equation (e.g. the equation EQ(l)_ACTBND is created when an annual activity bound is specified for a process having a diurnal timeslice resolution).
- **Solving the model:** After construction of the actual matrix (rows, columns, intersections and bounds) the problem is passed to an optimizing solver employing the appropriate technique (LP, MIP, or NLP). The solver returns the solution of the optimization back to GAMS. The information regarding the solver status is written by TIMES in a text file called END_GAMS, which allows the user to quickly check whether the optimisation run was successful or not without having to go through the listing file. Information from this file is displayed by VEDA-FE and ANSWER at the completion of the run.
- **Reporting:** Based on the optimal solution the reporting routines calculate result parameters, e.g. annual cost information by type, year and technology or commodity. These result parameters together with the solution values of the variables and equations (both primal and dual), as well as selected input data, are assembled in the <case>.GDX file. The gdx file is then processed by the GAMS GDX2VEDA.EXE utility according to the directives contained in TIMES2VEDA.VDD control file to generate files for the result analysis software VEDA-BE[14]. The <case>.ANT file for providing results for import into ANSWER may also be produced, if desired.

2.3 Files produced during the run process

Several files are produced by the run process. These include the files produced by the shell for model initiation, the .LST listing file, which echoes the GAMS compilation and execution

[14] The basics of the TIMES2VEDA.VDD control file and the use of the result analysis software VEDA-BE are described in Part V.

process and reports on any errors encountered during solve, results files, and the QAcheck.log file. These files are summarized in Table 2 and discussed in this section.

Table 3: Files Produced by a TIMES Model Run

Extension	Produced By	Nature of the Output
ant	TIMES report writer	ANSWER model results dump
gdx	GAMS	Internal (binary) GAMS Data eXchange file with all the information associated with a model run
log	TIMES quality check routine	List of quality assurance checks (warnings and possible errors)
lst	GAMS	The basic echo of the model run, including indication of the version of TIMES being run, the compilation and execution steps, model summary statistics and error (if encountered), along with optionally an equation listing and/or solution print
vdd	GDX2VEDA utility	The core model results dump of the solution including the variable/equations levels/slack and marginals, along with cost and other post-processing calculations
vde	GDX2VEDA utility	The elements of the model sets (and the definition of the attributes)
vds	GDX2VEDA utility	The set membership of the elements of the model
vdt	VEDA-FE or ANSWER	The RES topology information for the model

2.3.1 Files produced by model initiation

As discussed in Section 2.1, three sets of files are created by VEDA and ANSWER upon run initiation, the command script file VTRUN/ANSRUN.CMD, the top-level GAMS command file <Case>.RUN/GEN (for VEDA/ANSWER), and the data dictionary <scenario>.DD/DDS text file(s) that contain all the model data to be used in the run.

The VTRUN/ANSRUN.CMD script file calls GAMS, referring to the <case> file and identifying the location of the TIMES source code and gdx file. For VEDA-FE the CMD file consists of the line:

```
Call ..\<source_code_folder>\vt_gams <case>.run <source_code_folder> gamssave\<case>
```

along with a 2nd line to call the GDX2VEDA utility to process the TIMES2VEDA.VDD file to prepare the <case>.VD* result files for VEDA-BE. The ANSRUN.CMD file has a similar setup calling ANS_GAMS.CMD in the source code folder which invokes GAMS and subsequently the GDX2VEDA utility.

The <case>.RUN/GEN file is the key file controlling the model run. It instructs the TIMES code what data to grab, what model variant to employ, how to handle the objective function, and other aspects of the model run controlled by the switches discussed in Section 3. An example .RUN file is displayed in Figure 5. Rows beginning with an asterisk (*) are comment lines for the user's convenience and are ignored by the code. Rows beginning with a dollar-sign ($) are switches that can be set by the user (usually by means of VEDA/ANSWER).

Both VEDA and ANSWER have facilities to allow the user to tailor the content of the RUN/GEN files, though somewhat differently. In VEDA-FE the Case Manager RUNFile_Tmpl button allows the basic RUN template to be brought up, and if desired carefully edited. However, the Case Manager also has a Control Panel, shown in Figure 4, where many of the more common switches can be set.

At the beginning of a <case>.RUN file the version of the TIMES code being used is identified and some option control statements that influence the information output (e.g., SOLPRINT ON/OFF to see a dump of the solution, OFF recommended) are provided. The LIMROW/LIMCOL options allow the user to turn on equation listing in the .LST file (discussed in the next section) by setting the number of rows/columns of each type to be shown.

Then compile-time dollar control options indicating which solver to use (if not the default to the particular solution algorithm), whether to echo the source code ($ON/OFFLISTING) by printing it to the LST file, and that multiple definitions of sets and parameters ($ONMULTI) are permitted (that is they can appear more than one time, which TIMES requires since first there are empty declarations for every possible parameter followed by the actual data provided by the user). Further possible dollar control options are also described in the GAMS manual.

Afterwards the content of several so-called TIMES dollar control (or environment) switches are specified. Within the source code the use of these control switches in combination with queries enables the model to skip or activate specific parts of the code. Thus it is possible to turn-on/off variants of the code, e.g. the use of the reduction algorithm, without changing the input data. The meaning and use of the different control switches is discussed in Section 3. Again these are generally set using the Case Manager/Run form in VEDA/ANSWER.

After the basic control switches, the definition of the set of all timeslices is established by means of the call to the <case>_TS.DD file before any other declarations carried out in the initialization file INITSYS.MOD. This is necessary to ensure the correct ordering of the timeslices for seasonal, weekly, or daynite storage processes. After the definition of the timeslices, the files INITSYS.COM and INITMTY.MOD, which are responsible for the declaration and initialization of all sets and parameters of the model generator, are included.

```
$TITLE  TIMES -- VERSION 4.1.0
OPTION RESLIM=50000, PROFILE=1, SOLVEOPT=REPLACE;
OPTION ITERLIM=999999, LIMROW=0, LIMCOL=0, SOLPRINT=OFF;

option LP=cplex;

*--If you want to use an optimizer other than cplex/xpress, enter it here:
*OPTION LP=MyOptimizer;

$OFFLISTING
*$ONLISTING

* activate validation to force VAR_CAP/COMPRD
$SET VALIDATE 'NO'
* reduction of equation system
$SET REDUCE   'YES'
*--------------------------------------------------------------*
* BATINCLUDE calls should all be with lower case file names!!! *
*--------------------------------------------------------------*

* initialize the environment variables
$ SET DSCAUTO YES
$   SET VDA YES
$   SET DEBUG                           'NO'
$   SET DUMPSOL                         'NO'
$   SET SOLVE_NOW                       'YES'
$   SET MODEL_NAME                      'TIMES'
$   IF DECLARED REG      $SET STARTRUN 'RESTART'
$   IF NOT DECLARED REG  $SET STARTRUN 'SCRATCH'
$SET XTQA YES
* VAR_UC being set so that non-binding constraints appear in results
$SET VAR_UC YES
 OPTION BRATIO=1;
$ SET OBJ AUTO
$ SET OBLONG YES
$SET DAMAGE NO
$ SET DSC YES
$ SET FIXBOH 2012
$ SET LPOINT B_M0000
$ SET STAGES NO
$SET SOLVEDA 'YES'
$SET VARCOST LIN

* merge declarations & data
$   ONMULTI

* the times-slices MUST come 1st to ensure ordering OK
$BATINCLUDE s_m0tthx_ts.dd
```

{continued on next page}

```
* perform fixed declarations
$SET BOTIME 1970
$BATINCLUDE initsys.mod

* declare the (system/user) empties
$   BATINCLUDE initmty.mod
*$   BATINCLUDE initmty.mod DSC
$IF NOT DECLARED REG_BNDCST $Abort "You need to use TIMES v2.3.1 or higher"

$BATINCLUDE base.dd
$BATINCLUDE b-newtechs.dd
$BATINCLUDE syssettings.dd
$BATINCLUDE base_hfcs_emi.dd
$BATINCLUDE base_delivcost.dd
$BATINCLUDE base_exim_fp.dd
$BATINCLUDE base_feedin.dd
$BATINCLUDE base_minutilall.dd
$BATINCLUDE base_ref.dd
$BATINCLUDE base_rsd-com.dd
$BATINCLUDE base_elccap.dd
$BATINCLUDE base_stock.dd
$BATINCLUDE base_heat-ex.dd
$BATINCLUDE base_hrates.dd
$BATINCLUDE flex_tfc_structure.dd
$BATINCLUDE ee_measures.dd
$BATINCLUDE re_measures.dd
$BATINCLUDE re_targets.dd
$BATINCLUDE ee_targets.dd
$BATINCLUDE co2_tax_high.dd

SET MILESTONYR /2005,2006,2007,2008,2009,2010,2011,2012,2015,2020,2025,2030,2035,2040,2045,2050/
$SET RUN_NAME 'S_MOTTHx'

*GG* Add the LevelizedCost/Cost_NPV switches
$SET ANNCOST LEV
RPT_OPT('OBJ','1') = 1;

$ SET VEDAVDD 'YES'

* do the rest
$ BATINCLUDE maindrv.mod mod
```

Figure 5: Example of a VEDA-FE TIMES <case>.RUN file[15]

The line containing the include command for the file initmty.mod can be supplemented by calls for additional user extensions that trigger the use of additional special equations or report routines. The use of these extension options are described in more detail in Section 0.

Afterwards the data dictionary file(s) (BASE.DD, …, CO2_TAX_HIGH.DD in Figure 5) containing the user input sets and parameters are included, inserted automatically by VEDA-FE/ANSWER according to the list of scenarios in the Case Manager/Run forms by means of the $INCLUDE statements. It is normally advisable to segregate user data into “packets” as scenarios, where there may be a single Base scenario containing the core descriptions of the energy system being studied and a series of alternate scenario depicting other aspects of the system. For example, one <scenario>.DD file may contain the description of the energy system for a reference scenario, and additional <alt_scenario>.DD files (.DDS for ANSWER) may be

[15] The ANSWER GEN file will have similar content though with some syntax and perhaps slightly augmented scripts.

included containing additions or changes relative to the reference file, for example CO_2 mitigation targets for a reduction scenario, or alternative technology specifications.

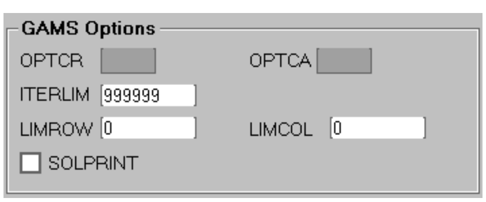
The SET MILESTONYR declaration identifies years for this model run based upon those years identified in in VEDA via the Period Defs selected on the Case Manager (and maintained in SysSettings) and the Milestone Years button on the ANSWER run form. The dollar control switch RUN_NAME contains the short name of the scenario, and is used for the name of the results files (<case>.VD*) passed to VEDA-BE.

Next in the example shown in Figure 5, some runtime switches are activated to request levelized cost reporting and splitting of investment costs into core and the incremental additional cost arising from any technology based discount rate specified in the data. See Section 3 for the full description of these and other control switches.

The last line of the <case>.RUN file invokes the file main driver routine (maindrv.mod) that initiates all the remaining tasks related to the model run (pre-processing, coefficient calculation, setting of bounds, equation generation, solution, reporting). Thus any information provided after the inclusion of the maindrv.mod file will not be considered in the main model solve request, though if the user wishes to introduce specialized post-processing of the result that could be added (or better yet handled externally by GAMS code that processes the GDX file).

2.3.2 GAMS listing file (.LST)

The GAMS listing file echoes the model run. In this file GAMS reports the compile and execution steps, presents a summary of the model statistics and objective function results, and reports any errors, if incurred. Optionally the user can request that the equation listing be turned on by specifying the LIMROW/LIMCOL (number of rows/columns of each type to be shown) and/or the solution dumped (via SOLPRINT) by means of the VEDA-FE CaseManager settings, as shown here.




```
* GEN file for use with ANSWER-TIMES GAMS code
*
* ANSWER will replace the keywords below enclosed in <> brackets.
* Replacement does not occur if line starts with *.
* Keywords are case-sensitive. Possible keywords include:
*     <Run>, <IncludeDDLines>, <TPList>, <ParameterBandParameterE>,
*     <IncludeRATRULEDDLines>, <SetModelVariantLines>,
*     <SetObjectiveFunctionLines>, <FIXBOHValue>, <TimeStep>, <Overlap>,
*     <ImportResultsIntoAnswer>, <SuppressPureZeroTSResults>
*     <SavepointLoadpointLines>, <G_DYEARValue>
*
$TITLE  NOBLE-SOFT ANSWER-TIMES VERSION 6.9.7 -- TIMES VERSION 3.9.0
OPTION LIMROW=0, LIMCOL=0, SOLPRINT=OFF, ITERLIM=50000;
OPTION RESLIM=50000, PROFILE=1, SOLVEOPT=REPLACE;
```

Figure 6: Requesting Equation Listing and Solution Print

For ANSWER these are handled by manually editing these entries in the GEN file either at runtime, or via the Run menu if the change is to be retrained as the default,

When GAMS takes its 1st pass the <Case>.LST file will report each of the individual source code modules compiled. [Note that for any particular TIMES model instance, according to the Run Switch settings, only the routines needed are invoked, as discussed in Section 3.]

A small snippet from the LST file compilation trace from an ANSWER-TIMES model run of is shown in Figure 7, where the "..." shows the nesting as one GAMS routine calls another with the appropriate parameters needed.

```
586     47887 BATINCLUDE   585      26  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
587     47967 BATINCLUDE   585      38  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
588     48038 BATINCLUDE   495     127  ...C:\AnswerTIMESv6\Gams_SrcTI\eqfloshr.mod
589     48064 BATINCLUDE   588      26  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
590     48144 BATINCLUDE   588      38  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
591     48219 BATINCLUDE   495     132  ...C:\AnswerTIMESv6\Gams_SrcTI\eqfloshr.mod
592     48245 BATINCLUDE   591      26  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
593     48325 BATINCLUDE   591      38  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
594     48396 BATINCLUDE   495     133  ...C:\AnswerTIMESv6\Gams_SrcTI\eqfloshr.mod
595     48422 BATINCLUDE   594      26  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
596     48502 BATINCLUDE   594      38  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
597     48573 BATINCLUDE   495     134  ...C:\AnswerTIMESv6\Gams_SrcTI\eqfloshr.mod
598     48599 BATINCLUDE   597      26  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
599     48679 BATINCLUDE   597      38  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
600     48754 BATINCLUDE   495     139  ...C:\AnswerTIMESv6\Gams_SrcTI\eqflomrk.mod
601     48782 BATINCLUDE   600      69  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
602     48882 BATINCLUDE   495     140  ...C:\AnswerTIMESv6\Gams_SrcTI\eqflomrk.mod
603     48910 BATINCLUDE   602      69  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
604     49010 BATINCLUDE   495     141  ...C:\AnswerTIMESv6\Gams_SrcTI\eqflomrk.mod
605     49038 BATINCLUDE   604      69  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
606     49142 BATINCLUDE   495     146  ...C:\AnswerTIMESv6\Gams_SrcTI\eqire.mod
607     49199 BATINCLUDE   495     152  ...C:\AnswerTIMESv6\Gams_SrcTI\eqirebnd.mod
608     49266 BATINCLUDE   495     153  ...C:\AnswerTIMESv6\Gams_SrcTI\eqirebnd.mod
609     49333 BATINCLUDE   495     154  ...C:\AnswerTIMESv6\Gams_SrcTI\eqirebnd.mod
610     49404 BATINCLUDE   495     159  ...C:\AnswerTIMESv6\Gams_SrcTI\eqpeak.mod
611     49432 BATINCLUDE   610      28  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_ire.mod
612     49476 BATINCLUDE   610      29  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_ire.mod
613     49530 BATINCLUDE   610      40  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_stgn.mod
614     49552 BATINCLUDE   610      43  ....C:\AnswerTIMESv6\Gams_SrcTI\cal_fflo.mod
615     49567 BATINCLUDE   614      37  .....C:\AnswerTIMESv6\Gams_SrcTI\cal_red.red
```

Figure 7: GAMS Compilation of the TIMES Source Code

If an error is encountered during the compilation operation GAMS will tag where the error occurred and report an error code. Most common in this regard is a Domain Error ($170) where perhaps there was a typo in an item name, as in the example shown in Figure 8, or the scenarios were not in the proper order and data was attempted to be assigned to a process before it was declared. Further discussion of errors encountered at this and other stages of the run process is found in Section 2.4.

```
TIMES -- VERSION 3.9.1 -- Restart (v3.9)
C o m p i l a t i o n


4392    'CCK'              'Commercial Cooking                           '
****        $170
**** LINE     38 INCLUDE     C:\AnswerTIMESv6\Gams_WrkStarter\BY-COM+STARTER.DDS
**** LINE    104 INPUT       C:\AnswerTIMESv6\Gams_WrkStarter\REF-01Z.GEN
20183  $ABORT  "*** ERRORS IN INPUT DATA/COMPILE ***"
****        $343
**** LINE     41 BATINCLUDE  C:\AnswerTIMESv6\Gams_SrcTI\err_stat.mod
                             %3  *** ERRORS IN INPUT DATA/COMPILE ***
                             %2  ABORT
                             %1  $IF NOT ERRORFREE
**** LINE     34 BATINCLUDE  C:\AnswerTIMESv6\Gams_SrcTI\maindrv.mod
                             %1  mod
**** LINE    170 INPUT       C:\AnswerTIMESv6\Gams_WrkStarter\REF-01Z.GEN

GAMS 24.4.1  r50296 Released Dec 20, 2014 WEX-WEI x86 64bit/MS Windows
03/20/16 12:22:49 Page 9
TIMES -- VERSION 3.9.1 -- Restart (v3.9)
Error Messages


170  Domain violation for element
343  Abort triggered by above statement

**** 2 ERROR(S)    0 WARNING(S)
```

Figure 8: GAMS Compilation Error

Once the data and code have been successfully complied, execution takes place, with GAMS calling each TIMES routine needed according to the switches and data for this particular run. Again the LST file echoes this execution phase, as shown in Figure 9. It is possible, though unlikely, to encounter a GAMS Execution error. The most common cause of this is the explicit specification of zero (0) as the efficiency of a process. An execution error is reported in the <Case>.LST file in a manner similar to a compilation error, tagged by "Error" at the point that the problem was encountered.

```
----  63012 Assignment FLO_CUM       0.000      0.983 SECS     49 MB      8
----  63014 Assignment FLO_CUM       0.000      0.983 SECS     49 MB      8
----  63017 Loop                     0.000      0.983 SECS     49 MB
----  63021 Assignment VAR_CUMFLO    0.000      0.983 SECS     49 MB      0
----  63022 Assignment VAR_CUMFLO    0.000      0.983 SECS     49 MB      8
----  63025 Assignment COM_CUM       0.000      0.983 SECS     49 MB      0
----  63026 Assignment COM_CUM       0.000      0.983 SECS     49 MB      0
----  63027 Assignment COM_CUM       0.000      0.983 SECS     49 MB      0
----  63029 Assignment COM_CUM       0.000      0.983 SECS     49 MB      0
----  63032 Loop                     0.000      0.983 SECS     49 MB
----  63036 Assignment VAR_CUMCOM    0.000      0.983 SECS     49 MB      0
----  63037 Assignment VAR_CUMCOM    0.000      0.983 SECS     49 MB      0
----  63041 Loop                     0.000      0.983 SECS     49 MB
----  63043 Assignment VAR_CUMCST    0.000      0.983 SECS     49 MB      0
----  63044 Assignment VAR_CUMCST    0.000      0.983 SECS     49 MB      0
----  63045 Assignment VAR_CUMCST    0.000      0.983 SECS     49 MB      0
----  63057 Assignment CNT           0.000      0.983 SECS     49 MB      1
----  63058 Assignment UC_T_EACH     0.000      0.983 SECS     49 MB   1520
----  63077 Assignment VAR_UC        0.000      0.983 SECS     49 MB      0
----  63080 Loop                     0.000      0.983 SECS     49 MB
----  63087 Assignment UC_RHS        0.000      0.983 SECS     49 MB      0
----  63108 Assignment VAR_UCR       0.000      0.983 SECS     49 MB      0
----  63112 Assignment VAR_UCR       0.000      0.983 SECS     49 MB      0
----  63113 Assignment VAR_UCR       0.000      0.983 SECS     49 MB      0
----  63114 Assignment VAR_UCR       0.000      0.983 SECS     49 MB      0
----  63118 Assignment UC_RHSR       0.000      0.983 SECS     49 MB      0
----  63139 Assignment VAR_UCT       0.000      0.983 SECS     49 MB      0
----  63142 Loop                     0.000      0.983 SECS     49 MB
----  63149 Assignment UC_RHST       0.000      0.983 SECS     49 MB      0
----  63170 Assignment VAR_UCRT      0.000      0.983 SECS     49 MB      0
```

Figure 9: GAMS Execution of the TIMES Source Code

Once execution of the matrix generator has completed GAMS reports the model run statistics (Figure 10), and automatically invokes the solver.

```
GAMS 24.4.1  r50296 Released Dec 20, 2014 WEX-WEI x86 64bit/MS Windows
05/13/15 13:44:59 Page 11
TIMES -- VERSION 3.6.0 -- Restart (v3.6)
Model Statistics    SOLVE TIMES Using MIP From line 63932


MODEL STATISTICS

BLOCKS OF EQUATIONS           93     SINGLE EQUATIONS      109,278
BLOCKS OF VARIABLES           13     SINGLE VARIABLES       58,035  4 projected
NON ZERO ELEMENTS        419,203


----   63932 Solve Fini TIMES         0.109     2.527 SECS     86 MB  419203
GENERATION TIME      =        1.544 SECONDS     86 MB  24.4.1 r50296 WEX-WEI


EXECUTION TIME       =        2.527 SECONDS     86 MB  24.4.1 r50296 WEX-WEI
----   63932 GAMS Fini                0.047     0.047 SECS     86 MB
----       1 ExecInit                 0.000     0.000 SECS     46 MB
----   63932 Solve Alg  TIMES         0.000     0.000 SECS     46 MB
```

Figure 10: CPLEX Solver Statistics

If the OPTION LIMROW/LIMCOL is set to non-0 the equation mathematics are displayed in the list file, by equation block and/or column intersection, as shown in Figure 11 and Figure 12 respectively.

```
---- EQG_COMBAL  =G=  Commodity Balance  (=G=)

EQG_COMBAL(STARTER,2013,ELCD,FAD)..  VAR_ACT(STARTER,2013,2013,GRD-ELCD-1,FAD) - VAR_ACT(STARTER,2013,2013,XAGRELC00,FAD) - VAR_ACT
(STARTER,2013,2013,XCOMELC00,FAD) - VAR_ACT(STARTER,2013,2013,XRSDELC00,FAD) - VAR_ACT(STARTER,2013,2013,XINDELC00,FAD)

      - VAR_ACT(STARTER,2013,2013,XTRNELC00,FAD) + VAR_ACT(STARTER,2013,2013,ZZBCKELC,FAD) =G= 0 ; (LHS = 0)

EQG_COMBAL(STARTER,2013,ELCD,FAN)..  VAR_ACT(STARTER,2013,2013,GRD-ELCD-1,FAN) - VAR_ACT(STARTER,2013,2013,XAGRELC00,FAN) - VAR_ACT
(STARTER,2013,2013,XCOMELC00,FAN) - VAR_ACT(STARTER,2013,2013,XRSDELC00,FAN) - VAR_ACT(STARTER,2013,2013,XINDELC00,FAN)

      - VAR_ACT(STARTER,2013,2013,XTRNELC00,FAN) + VAR_ACT(STARTER,2013,2013,ZZBCKELC,FAN) =G= 0 ; (LHS = 0)

EQG_COMBAL(STARTER,2013,ELCD,FAP)..  VAR_ACT(STARTER,2013,2013,GRD-ELCD-1,FAP) - VAR_ACT(STARTER,2013,2013,XAGRELC00,FAP) - VAR_ACT
(STARTER,2013,2013,XCOMELC00,FAP) - VAR_ACT(STARTER,2013,2013,XRSDELC00,FAP) - VAR_ACT(STARTER,2013,2013,XINDELC00,FAP)

      - VAR_ACT(STARTER,2013,2013,XTRNELC00,FAP) + VAR_ACT(STARTER,2013,2013,ZZBCKELC,FAP) =G= 0 ; (LHS = 0)

EQG_COMBAL(STARTER,2013,ELCD,SPD)..  VAR_ACT(STARTER,2013,2013,GRD-ELCD-1,SPD) - VAR_ACT(STARTER,2013,2013,XAGRELC00,SPD) - VAR_ACT
(STARTER,2013,2013,XCOMELC00,SPD) - VAR_ACT(STARTER,2013,2013,XRSDELC00,SPD) - VAR_ACT(STARTER,2013,2013,XINDELC00,SPD)

      - VAR_ACT(STARTER,2013,2013,XTRNELC00,SPD) + VAR_ACT(STARTER,2013,2013,ZZBCKELC,SPD) =G= 0 ; (LHS = 0)

EQG_COMBAL(STARTER,2013,ELCD,SPN)..  VAR_ACT(STARTER,2013,2013,GRD-ELCD-1,SPN) - VAR_ACT(STARTER,2013,2013,XAGRELC00,SPN) - VAR_ACT
(STARTER,2013,2013,XCOMELC00,SPN) - VAR_ACT(STARTER,2013,2013,XRSDELC00,SPN) - VAR_ACT(STARTER,2013,2013,XINDELC00,SPN)

      - VAR_ACT(STARTER,2013,2013,XTRNELC00,SPN) + VAR_ACT(STARTER,2013,2013,ZZBCKELC,SPN) =G= 0 ; (LHS = 0)
```

Figure 11: Equation Listing Example

```
VAR_ACT(STARTER,2015,2015,EEBIOGAS-ST-X0,WIP)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
        0.2859  EQ_OBJVAR(STARTER,CUR)
        1       EQG_ACTBND(STARTER,2015,EEBIOGAS-ST-X0,ANNUAL)
        1       EQL_CAPACT(STARTER,2015,2015,EEBIOGAS-ST-X0,WIP)
        1       EQG_COMBAL(STARTER,2015,ELCT,WIP)
       -4       EQG_COMBAL(STARTER,2015,PWRBIOGAS,ANNUAL)

VAR_ACT(STARTER,2015,2015,EEBIOMSW-ST-X0,FAD)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
        0.2859  EQ_OBJVAR(STARTER,CUR)
        1       EQG_ACTBND(STARTER,2015,EEBIOMSW-ST-X0,ANNUAL)
        1       EQL_CAPACT(STARTER,2015,2015,EEBIOMSW-ST-X0,FAD)
        1       EQG_COMBAL(STARTER,2015,ELCT,FAD)
       -5.56    EQG_COMBAL(STARTER,2015,PWRBIOMSW,ANNUAL)

VAR_ACT(STARTER,2015,2015,EEBIOMSW-ST-X0,FAN)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
        0.2859  EQ_OBJVAR(STARTER,CUR)
        1       EQG_ACTBND(STARTER,2015,EEBIOMSW-ST-X0,ANNUAL)
        1       EQL_CAPACT(STARTER,2015,2015,EEBIOMSW-ST-X0,FAN)
        1       EQG_COMBAL(STARTER,2015,ELCT,FAN)
       -5.56    EQG_COMBAL(STARTER,2015,PWRBIOMSW,ANNUAL)

VAR_ACT(STARTER,2015,2015,EEBIOMSW-ST-X0,FAP)
                (.LO, .L, .UP, .M = 0, 0, +INF, 0)
        0.2859  EQ_OBJVAR(STARTER,CUR)
        1       EQG_ACTBND(STARTER,2015,EEBIOMSW-ST-X0,ANNUAL)
        1       EQL_CAPACT(STARTER,2015,2015,EEBIOMSW-ST-X0,FAP)
        1       EQG_COMBAL(STARTER,2015,ELCT,FAP)
       -5.56    EQG_COMBAL(STARTER,2015,PWRBIOMSW,ANNUAL)
```

Figure 12: Variable Listing Example

And if the SOLPRINT=ON option is activated then the level and marginals are reported as shown in Figure 13.

```
---- EQU EQL_FLOMRK  Process market-share (=L=)

                                              LOWER          LEVEL          UPPER         MARGINAL

STARTER.2015.CHB-F-COA-UP   .CHB.ANNUAL        -INF          -4.7599           .              .
STARTER.2015.CHB-F-GEO-UP   .CHB.ANNUAL        -INF          -0.5895           .              .
STARTER.2015.CHB-F-LTH-UP   .CHB.ANNUAL        -INF             .              .           -0.2796
STARTER.2015.RHB-F-BIOPSF-UP.RHB.ANNUAL        -INF          -0.6310           .              .
STARTER.2015.RHB-F-COALIG-UP.RHB.ANNUAL        -INF             .              .           -0.0070
STARTER.2015.RHB-F-LTH-UP   .RHB.ANNUAL        -INF          -1.7232           .              .
STARTER.2015.RHB-Q-AD-UP    .RHB.ANNUAL        -INF         -37.9473           .              .
STARTER.2015.RHB-Q-BE-UP    .RHB.ANNUAL        -INF             .              .           -0.0906
STARTER.2015.RHB-Q-IM-UP    .RHB.ANNUAL        -INF             .              .           -0.0739
STARTER.2015.TBU-F-GASNAT-UP.TBU.ANNUAL        -INF          -0.0287           .              .
STARTER.2015.THS-F-GASNAT-UP.THS.ANNUAL        -INF          -0.0385           .              .
STARTER.2015.TLD-S-CP-UP    .TLD.ANNUAL        -INF             .              .         -193.7121
STARTER.2015.TLD-S-FS-UP    .TLD.ANNUAL        -INF             .              .         -177.4296
STARTER.2015.TLD-S-LS-UP    .TLD.ANNUAL        -INF          -0.3822           .              .
STARTER.2015.TLD-S-MC-UP    .TLD.ANNUAL        -INF             .              .           -2.9796
STARTER.2015.TLD-S-MV-UP    .TLD.ANNUAL        -INF             .              .          -75.3503
STARTER.2015.TLD-S-PU-UP    .TLD.ANNUAL        -INF             .              .          -66.5300
STARTER.2015.TLD-S-SS-UP    .TLD.ANNUAL        -INF             .              .         -146.9064
STARTER.2015.TLD-T-FX-UP    .TLD.ANNUAL        -INF             .              .           -1.0684
STARTER.2015.TLD-T-HY-UP    .TLD.ANNUAL        -INF             .              .          -18.6543
STARTER.2015.TLD-T-PH-UP    .TLD.ANNUAL        -INF             .              .          -15.3928
STARTER.2015.TMD-F-GASNAT-UP.TMD.ANNUAL        -INF          -0.0498           .              .
STARTER.2020.CHB-F-COA-UP   .CHB.ANNUAL        -INF          -5.2941           .              .
```

Figure 13: Solution Dump Example

Upon successful solving the model the solution statistics are reported (Figure 14), where in this case CPLEX was used to solve a MIP model variant (in this example), and the report writer invoked to finish up by preparing the report. If the solver is not able to find an optimal solution, a non-Normal solve status will be reported, and the user can search the LST file for the string "INFES" for an indication of which equations are preventing model solution. Again, further information on the possible causes and resolution of such errors is found in Section 2.4.

```
Solution Report     SOLVE TIMES Using MIP From line 63932


               S O L V E      S U M M A R Y

     MODEL   TIMES                OBJECTIVE  OBJZ
     TYPE    MIP                  DIRECTION  MINIMIZE
     SOLVER  CPLEX                FROM LINE  63932

**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      1 Optimal
**** OBJECTIVE VALUE          6543356.6761

 RESOURCE USAGE, LIMIT        240.226     50000.000
 ITERATION COUNT, LIMIT      1202         50000

**** Reading with SOLVEOPT=REPLACE (0) ****

IBM ILOG CPLEX   24.4.1 r50296 Released Dec 20, 2014 WEI x86 64bit/MS Windows
--- GAMS/Cplex licensed for continuous and discrete problems.
Cplex 12.6.1.0
```

Figure 14: Solver Solution Summary

The actual production of the dump of the model results is performed by the report writer for ANSWER resulting in a <Case>.ANT file which is imported back into ANSWER after the run complete and/or the GDX2VEDA utility prepared by GAMS and DecisionWare to facilitate the exchange of information from GAMS to VEDA-BE, which may be used with both VEDA-FE and ANSWER.

2.3.3 Results files

The TIMES report writing routine produces two sets of results-related outputs (along with the quality control LOG discussed in the next section). The <case>.ANT file is an ASCII text file, with results ready for import into ANSWER. The GAMS Data eXchange file (GDX) contains all the information associated with a model run [input data, intermediate parameters, model results (primal and dual)] in binary form. The GDX file may be examined by means of the GAMSIDE, available from the Windows Start Menu in the GAMS folder (or as a shortcut from the desktop if put there), if one really wants to dig into what's happening inside of a TIMES run (that is, the set members, preprocessor calculations, the model solution and the reporting parameters calculated).

A more powerful feature within the GAMSIDE is a GDXDIFF facility under Utilities. As seen in Figure 15, the utility shows the differences between all components, comparing two model runs. Within the GDXDIFF utility, the user identifies the GDX files from the two runs and requests the resulting comparison GDX be prepared. The display then shows any differences between the two runs. The GDXDIFF is most effectively used by instructing VEDA to Create DD for the two runs via the Options and Case Manager forms, as shown in Figure 16. Once the comparison GDX has been created, it is viewed in the GAMSIDE. By sorting by Type and scanning down one Symbol at a time, one can determine exactly what input data being sent to GAMS for the two runs is different.

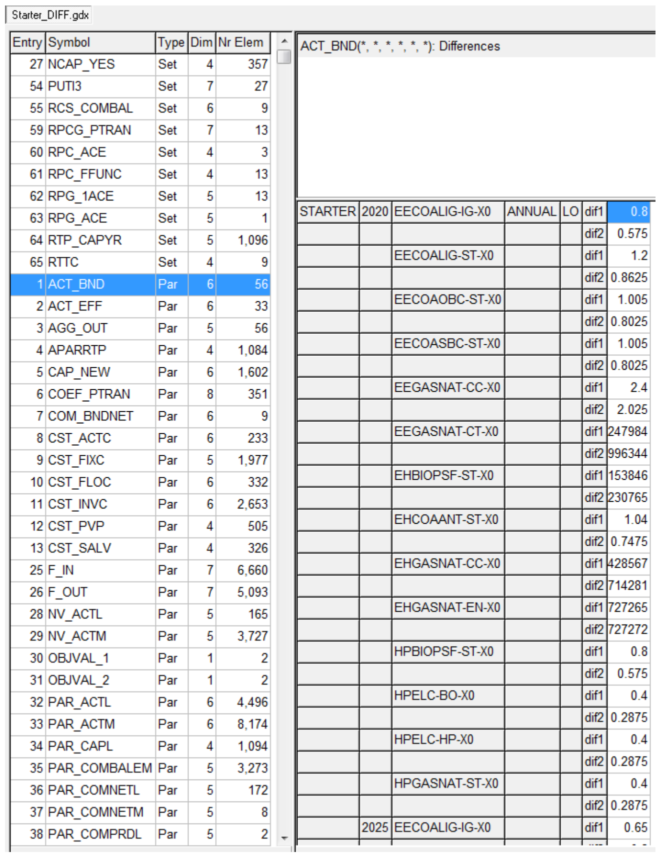


Figure 15: GAMSIDE View of the GDXDIFF Run Comparison

However, the most common use of the GDX is its further processing to generate files for the result analysis software VEDA-BE[16]. ETSAP worked with GAMS a number of years ago to develop a standalone utility (GDX2VEDA) to process the GAMS GDX file and produce the files read into VEDA-BE. The GDX2VEDA utility process a directives file (TIMES2VEDA.VDD) to determine which sets and model results are to be included and prepare said information for VEDA-BE. A general default version of the VDD is distributed with TIMES in the source code folder (for core TIMES, Stochastics, and MACRO), but may be augmented by the user if other information is desired from the solution. However, the process of changing the VDD should be done in consultation with someone fully familiar with the GAMS GDX file for TIMES and the

[16] The basics of the TIMES2VEDA.VDD control file and the use of the result analysis software VEDA-BE are described in Part V.

basics of the GDX2VEDA utility. See Part V, Appendix B, for further information on the GDX2VEDA utility and VDD directives file.

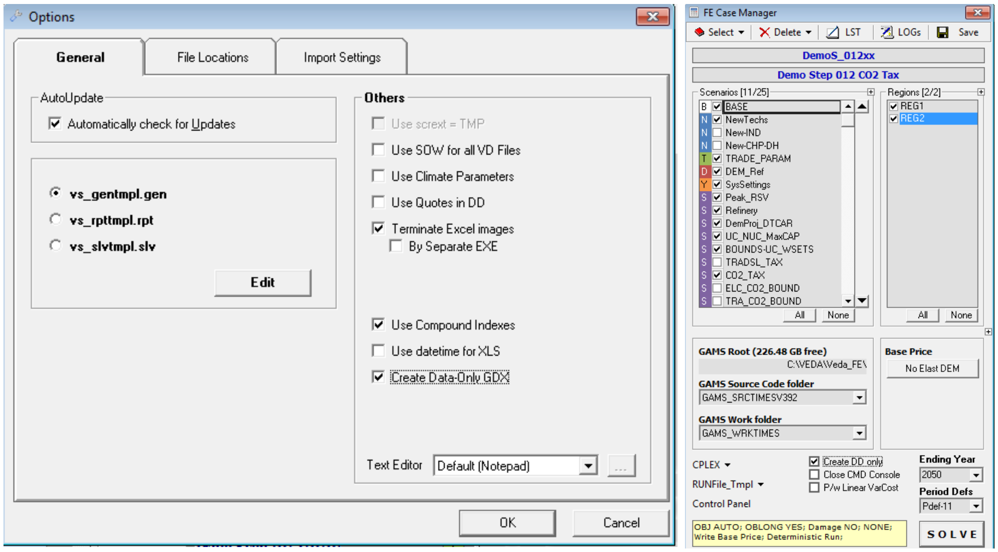


Figure 16: VEDA Setup for Data Only GDX Request

The call to the GDX2VEDA routine is embedded in the VTRUN/ANSRUN.CMD command routines. There are three files produced for VEDA-BE by the GDX2VEDA utility: the <Case>.VD data dump with the attributes, and associated VDE (set elements), VDS (sets definition). In addition, VEDA-FE and ANSWER produce a <Case>.VDT (topology) file with the RES connectivity information. These files never require user intervention, though users wishing to post-process the GDX2VEDA results with their own tailored software, rather than VEDA-BE, might choose to parse the VD* files to extract the desired information.

Note that for both ANSWER and VEDA-BE, for the most part low-level (that is commodity/process) results are reported, along with some aggregate cost numbers (such as regional and overall objective function). It is left up to the user to construct relevant sets and tables in VEDA-BE to organize and aggregate the results into meaningful tables. Refer to Part V for a discussion of how to go about assembling report tables in VEDA-BE. For ANSWER the user is left with only the raw results and thereby needs to come up with their own approach to producing useful usable reporting tables, or use VEDA-BE.

In addition, as discussed in Section 3.10, there are a number of switches that control the report writer itself in terms of how it calculates certain outputs and prepares the results as part of the post-processing. Collectively these mechanisms provide the user with a wide range of reporting results and tools for dissecting and assembling the modeling results as part of effectively using TIMES to conduct energy policy analyses.

2.3.4 QA check report (LOG)

In order to assist the user with identifying accidental modelling errors, a number of sanity checks are done by the model generator. If incorrect or suspicious specifications are found in these checks, a message is written in a text file named QA_CHECK.LOG, in the working folder. The checks implemented in TIMES Version 3.9.3 are listed in Table 5. The "Log entry" column shows the identification given for each suspicious specification.

Table 4: TIMES Quality Assurance Checks (as of Version 3.9.3)

Type[17]	Message / Description	Severity	Log entry
STD	Delayed Process but PAST Investment: Process availability has been delayed by using PRC_NOFF or NCAP_START, but also has existing capacity.	warning	region, process
STD	Commodities/processes defined at non-existing TSLVL: PRC_TSL or COM_TSL has been specified on a timeslice level not used in the model.	severe error	number of COM/PRC reset to ANNUAL
STD	NCAP_TLIFE out of feasible range: NCAP_TLIFE specified has a value of either less than 0.5 or greater than 200. Values less than 0.5 are reset to 1.	warning	region, process, vintage
STD	Flow OFF TS level below variable TS level: A PRC_FOFF attribute with a timeslice below the flow level has been specified; the OFF specification is ignored.	warning	region, process, commodity, timeslice
STD	COM_FR does not sum to unity (T=first year): The sum of COM_FR over all timeslices at the COM_TSL level is not equal to 1, and is therefore normalized to 1.	warning	region, commodity, milestone
STD	Unsupported diverging trade topology: The model generator detects an unsupported complex topology of an IRE process, which cannot be properly handled	error	region, process
STD	FLO_EMIS with no members of source group in process: A FLO_EMIS with a source group that has no members for the process has been specified. The parameter is ignored.	severe error	region, process, group, commodity
STD	Unsupported FLO_SHAR: C not in RPC or CG: The commodity in FLO_SHAR is either not in the process topology or not a member of the group specified	error	region, process, commodity, group
STD	FLO_SHAR conflict: Both FX + LO/UP specified, latter ignored: Too many FLO_SHAR bounds are specified, if both FX and LO/UP are specified at the same time.	warning	region, process, vintage, commodity, group

[17] STD=standard QA check (always done), XTD=extended QA check (activate with XTQA)

Type[17]	Message / Description	Severity	Log entry
STD	Inconsistent sum of fixed FLO_SHARs in Group: All flows in a group have a fixed share, but the sum of the fixed FLO_SHAR values is not equal to 1.	warning	region, process, vintage, group
STD	Defective sum of FX and UP FLO_SHARs in Group: All flows in a group have either a fixed or an upper share, but the sum of the FLO_SHAR values is less than 1.	warning	region, process, vintage, group
STD	Excessive sum of FX and LO FLO_SHARs in Group: All flows in a group have either a fixed or a lower share, but the sum of the FLO_SHAR values is greater than 1.	warning	region, process, vintage, group
STD	NCAP_AF/ACT_BND Bounds conflict: Value at PRC_TS level and below, latter ignored	warning	region, process, vintage, timeslice
STD	NCAP_AF Bounds conflict: FX + LO/UP at same TS-level, latter ignored	warning	region, process, vintage, timeslice
STD	FLO_SHAR/FLO_FR Bounds conflict: Value at RPCS_VAR level and below, latter ignored	warning	region, process, vintage, timeslice
STD	FLO_SHAR Bounds conflict: FX + LO/UP at same TS-level, latter ignored	warning	region, process, vintage, commodity, group
STD	COM_BNDNET/COM_BNDPRD/IRE_BND Bounds conflict: Value at COM_TS level and below, latter ignored	warning	region, milestone, commodity, timeslice
STD	IRE_FLO import commodity not in TOP_IRE: An invalid IRE_FLO with the imported commodity not in the process topology has been specified	error	region, process, commodity
STD	CHP process with zero CEH but only upper bound on CHPR: A CHP process has only an upper bound on NCAP_CHPR, but a zero or missing NCAP_CEH, which indicates a modelling error	error	region, process
STD	Year Fraction G_YRFR is ZERO! A timeslice with G_YRFR is within the timeslice tree. This should actually never happen, because TIMES automatically removes timeslices with a zero year fraction from the active timeslices.	fatal	region, timeslice
STD	Illegal system commodity in topology: ACT / ACTGRP is a reserved name which should never be used as a commodity in the model topology.	fatal	region, process

Type[17]	Message / Description	Severity	Log entry
STD	Commodity in CG of process P but not in topology: A commodity group assigned to a process contains members not in the process topology (or no members in the process topology).	severe error	region, process, commodity, group
STD	Elastic Demand but either COM_BPRICE/ELAST/VOC missing: Either a demand that has COM_ELAST or COM_STEP defined but does not have COM_BPRICE, COM_ELAST, or COM_VOC, defined.	warning	region, commodity, LO/UP
STD	Commodity type is also a commodity: Commodity types are reserved names that cannot be used as commodity names.	fatal	region, commodity type
STD	Commodity has ambiguous base type: The base type is either for some commodity is not uniquely defined. All commodities should have a unique base type defined (NRG/MAT/DEM/ENV/FIN).	severe error	region, commodity
STD	Demand: DEM commodity with missing COM_PROJ Projection: A demand commodity without any demand projection is found.	warning	region, commodity
STD	Demand: COM_PROJ specified for non-DEM commodity: Demand is projected for a non-demand commodity.	warning	region, commodity
STD	Phantom entries found in topology (process/commodity not in SET PRC/COM): This error is usually triggered when a GAMS domain violation has occurred, which may cause unexpected behaviour of the GAMS code.	fatal	region, process, commodity
STD	Process with missing or mismatched CG/PRC_ACTUNIT: Process with missing or several PRC_ACTUNT entries.	fatal	region, process
STD	Illegal dependency of substituted auxiliary commodities C1 and C2 in FLO_SUM: This error should not occur, because the GAMS code should make sure that C1 and C2 are not both substituted. If this error is issued, contact TIMES maintenance.	fatal	region, process, commodity1, commodity2
STD	NCAP_AFX defined for NON-vintaged dispatchable process with ACT_MINLD: Shaping of NCAP_AF is not supported for non-vintaged processes having ACT_MINLD defined.	warning	region, process
STD	Internal Region without Discount Rate: TIMES requires a discount rate defined for all internal regions.	fatal	region
STD	Active Currency without Discount Rate: A currency is being used without discount rate, or conversion to another currency that has a discount rate.	fatal	region, currency
STD	Process with zero PRC_ACTFLO for C in PG: A zero PRC_ACTFLO has been specified for a commodity in the primary group.	fatal	region, process, commodity
XTD	Same Commodity IN and OUT of non-STG process: A process has been defined to have the same commodity as an input and an output, and it is not a storage process; that is not supported.	severe error	region, process, commodity
XTD	IRE Process with invalid Parameters:	error	region,

Type[17]	Message / Description	Severity	Log entry
	Some FLO_FUNC, FLO_SUM, FLO_SHAR or UC_FLO parameter not supported for IRE processes has been specified.		process, com-group
XTD	Invalid Commodity / Group used in ACT_EFF - parameter ignored: An invalid ACT_EFF attribute with a CG not containing members on the shadow side or in the PG has been specified.	error	region, process, group
XTD	FLO_SUM Commodity Not in RPC - parameter ignored: An invalid FLO_SUM has been defined where the commodity is not in the process topology.	error	region, process, group, commodity
XTD	FLO_SUM Commodity Not in CG1 - parameter ignored: An invalid FLO_SUM has been defined where the commodity is not a member of the first group, CG1.	error	region, process, group, commodity
XTD	PTRANS between CG1 and CG2 in both directions: A FLO_FUNC or FLO_SUM between groups CG1 and CG2 has been specified in both directions.	severe error	region, process, group1, group2
XTD	RPC in TOP not found in any ACTFLO / FLO_SHAR / FLO_FUNC / FLO_SUM: Some commodity in the topology does not seem to be tied to anything, at least by means of any of the most common attributes; the user is advised to check that this is not a modelling error.	warning	region, process, commodity, IN/OUT
XTD	Empty Group in FLO_SUM/FLO_FUNC/FLO_SHAR: A group that has no members in the process topology has been used for a process attribute. Detects also an empty primary group.	severe error	region, process, group
XTD	Both NCAP_AF and NCAP_AFA specified for same process: Specifying both NCAP_AF(bd) and NCAP_AFA(bd) for an ANNUAL level process is ambiguous and should be avoided.	warning	region, process, vintage
XTD	Too Long Commodity Lead Time: A value of NCAP_CLED > NCAP_ILED has been specified	warning	region, process, commodity
XTD	CHP parameter specified for Non-CHP process: An NCAP_BPME, NCAP_CHPR or NCAP_CEH parameter has been specified for a process that is not defined to be CHP.	error	region, process, vintage
XTD	PG of CHP process consists of single commodity yet has a CHP-ratio: A CHP process has a NCAP_CHPR specified but has only a single commodity in the primary group.	warning	region, process
XTD	Found CHP processes without CHP-ratio defined: A CHP process has no NCAP_CHPR defined	warning	number of such processes
XTD	Found CHP processes with PG commodity efficiencies - unsupported: Specifying ACT_EFF on some flow(s) in the PG is not supported for CHP processes, and may lead to unexpected results.	warning	region, process
XTD	Found CHP processes without electricity in the PG: A CHP process is found with no electricity commodity in the PG.	warning	region, process

2.4 Errors and their resolution

Errors may be encountered during the compilation, execution (rarely), or solve stages of a TIMES model run. During the compilation step, if GAMS encounters any improperly defined item the run will be halted with a Domain or similar error and the user will need to examine the TIMES quality control LOG or GAMS listing (LST) files to ascertain the cause of the problem. While such problems are not normally encountered, some that might occur include:

- an item name was mistyped and therefore not defined;
- an item was previously defined in one scenario but defined differently in another;
- an item was not properly declared for a particular parameters (e.g., a non-trade process using an IRE parameter), and
- scenarios were specified for the run in the wrong order so a data reference is encountered before the declaration (e.g., a bound on a new technology option is provided before it has been identified).

During the execution phase, if GAMS encounters any runtime errors it will halt and report where the error occurred in the LST file. While such problems are not normally encountered some causes of an execution error might be:

- an explicit 0 is provided for an efficiency resulting in a divide by 0, and
- there is a conflict between a lower and upper bound.

Most commonly errors are encountered during the solve process, resulting in an infeasibility. Some causes of the model not being able to solve might be:

- due to bounds, the energy system cannot be configured in such a way as to meet the limit;
- owing to mis-specifying the demand serviced by a device, there is no or not enough capacity to satisfy said demand, and
- the RES is not properly connected so a needed commodity is not able to reach the processes needing it.

To identify the cause of a solve error, if using CPLEX the user can activate the Infeasibility Finder (set in the CPLEX.OPT as default (via the IIS command) in VEDA-FE Case Manager or said file distributed with ANSWER). The CPLEX Infeasibility Finder will identify the explicit row/columns corresponding to the first infeasibility encountered and list the conflict involved in the <Case>.LST file, such as shown here where the electricity balance equation can’t be satisfied (due to a limit being imposed on the first year electric grid capacity that is too small).

```
Implied bounds make row 'EQG_COMBAL('STARTER'.2013.'ELCD'.'FAD')' infeasible.
```

This helps with tracking down the culprit, but the user still needs to figure out why the problem occurred. When using a solver other than CPLEX, or if the Infeasibility Finder is not activated, then the solution dump will be tagged for all the potentially interrelated model variables/equations that were not in equilibrium at the time the solve stopped. The user can find these by searching the LST file for the string "INFES".

As a last resort, the model can be run with the equation listing turned on by setting LIMROW/LIMCOL to, say, 1000 in the <case>.RUN (via the Case Manager) / GEN (via Edit the GEN from the Run form) file, although the equations in this form can be challenging to interpret.

3 Switches to control the execution of the TIMES code

This Section describes the various GAMS control variables available in TIMES as control switches that can be set by the user in the model <case>.RUN/GEN file for VEDA-FE/ANSWER respectively. As discussed in Section 2, VEDA-FE and ANSWER, in most cases automatically take care of inserting the proper switches into the run file, so the user normally does not have to modify the run file at all. The switches are set in the highly user-friendly GUI interface of the user shell, which uses a run file template and inserts all run-specific switches correctly into the run file of each model run. These are managed by the user via the CaseManager (Control Panel) and Run/Edit GEN Template parts of VEDA-FE and ANSWER respectively.

In the sub-sections that follow, unless otherwise stated, the basic syntax for the inclusion of control switch options in the main <case>.RUN/GEN GAMS directive file is $SET <option> <value>. The various options are grouped according to the nature of their usage by sub-section.

3.1 Run name case identification

The use of the RUN_NAME control variable is practically mandatory when running TIMES. By setting the RUN_NAME control variable, the user gives a name to the model run, which will be used when generating various output files and/or loading information from a previously generated file that has the same name. The control variable is used in the following way:

```
$SET RUN_NAME runname
```

Here the ***runname*** identifier (corresponding to the run <case> name) is a string of letters, numbers and other characters (excluding spaces), such that the name complies with the rules for the base name of files. It will be used to construct names for the various files comprising a model run, as listed in Table 5.

Table 5: RUN_NAME TIMES Files

Extension*	Description
ANT	ANSWER results dump
GDX	GAMS data exchange file (for GAMS2VEDA processing)
*_BP.GDX	Base prices to seed a TIMES elastic demand policy run
LOG	Optional GAMS file producing a trace of the model resource usage (activated by lo=1 on the GAMS call line in ANS_GAMS/VT_GAMS.CMD, which needs to be added by the user manual if needed)
LST	GAMS output file with the compile/execute/solve trace, and optional solution dump (via SOLPRINT=YES on the OPTIONS line at the top of the RUN command script)
*_P.GDX	Save/Load point GAMS restart files
RUN	Top level routine calling GAMS (and for VEDA-FE the GDX2VEDA routine)
*_RunSummary.DD	Run summary for the associated model run
*_TS.DD	Timeslices declaration for the associated model run
VD*	Suite of results/Set definition(S)/Element description(E)/topology(T) for VEDA-BE

3.2 Controls affecting equilibrium mode

TIMES has a number of variants or model instances embedded in the within the full set of GAMS source code files. Which path through the code is taken is determined mainly the activation (or not) of various control switches, as summarized in this section.

3.2.1 Endogenous elastic demands [TIMESED]

The TIMESED control variable is one of the most important TIMES control variables. It has to be used whenever the full partial equilibrium features of TIMES (that is, employing elastic demands) are to be utilized. For running a baseline scenario to be subsequently used as the reference scenario for partial equilibrium analyses with elastic demands, the following setting should be used:

```
$SET TIMESED NO
```

This setting indicates that the user plans to use the resulting price levels from the current run as reference prices in subsequent runs with elastic demands. The setting causes the model generator to create the following (identical) two files from the Baseline run (the second file is a backup copy):

- Com_Bprice.gdx
- %RUN_NAME%_DP.gdx

For running any policy scenarios with elastic demands, using price levels from a previous run as reference prices, one must use the following setting:

```
$SET TIMESED YES
```

The reference price levels are read from a file named 'Com_Bprice.gdx', which is expected to reside in the current directory folder where the model run takes place. Therefore, the Baseline scenario using the setting $SET TIMESED NO has to be run before running the policy scenarios, or the correct 'Com_Bprice.gdx' be otherwise restored from some backup copy.

The VEDA-FE CaseManager(BasePrice) and ANSWER Run(ModelVariant) buttons allow the user to easily control setting of the TIMESED switch and thereby creating/including the Com_Bprice.GDX as appropriate. If neither the base prices are to be written out, nor a policy scenario with elastic demands to be run, the user should not set the TIMESED control variable.

3.2.2 General equilibrium [MACRO]

The general equilibrium mode of TIMES can be activated in two different ways, using the following MACRO control switch settings:

`$SET MACRO YES`	– activate the standard MACRO formulation
`$SET MACRO MSA`	– activate the MACRO decomposition formulation

For further information about the standard MACRO formulation, see the TIMES-MACRO documentation available at the ETSAP site: http://www.iea-etsap.org/web/documentation.asp

In both formulations, the use of the MACRO mode for evaluating policy scenarios requires that so-called demand decoupling factors (DDF) and labor growth rates first have to be calibrated for the Baseline scenario and corresponding GDP growth projections. The calibration produces a file containing the calibrated parameters, which must then be included in the policy scenarios to be evaluated.

Until TIMES v3.3.9, the standard MACRO formulation included a separate utility for calibrating the DDF factors and labor growth rates (see the TIMES-MACRO documentation for details). However, the calibration is much easier using the new MACRO decomposition formulation, where you can use the following MACRO control switch setting for carrying out the Baseline calibration:

`$SET MACRO CSA` – calibration with the MACRO decomposition method

The “CSA” calibration facility produces a file called MSADDF.DD, which is automatically included in any subsequent policy run activated by the MACRO=MSA control switch. In order to carry out the calibration, one must also include the necessary MACRO parameters in the model input data (see the TIMES-MACRO documentation for a description of the MACRO parameters). The only mandatory parameters are the initial GDP and GDP growth parameters.

The DDF file produced by CSA can be used for the original TIMES-MACRO formulation, where one may re-calibrate it a few times more with the Baseline scenario to verify the calibration for TIMES-MACRO. The re-calibration is automatically done by TIMES-MACRO at the end of each run, whereupon a new file DDFNEW.DD is written, which can then be renamed for inclusion in the subsequent TIMES-MACRO policy runs.

When using the MACRO decomposition formulation (with MACRO=MSA or MACRO=CSA), and when the partial equilibrium runs of the Baseline and policy scenarios have already been made, using the LPOINT/RPOINT control settings (in combination with SPOINT) may also be useful (see Section 3.8 and 0). If the RPOINT setting is used, the initial solution for the decomposition algorithm is taken directly from the GDX file without having to re-run the previously solved LP model again.

3.3 Controls affecting the objective function

3.3.1 Objective function cost accounting [OBJ]

The user can choose to use several alternative objective function formulations instead of the standard objective function. See Part I, Section 5.3.4 and the documentation for the Objective Function Variants for details. The alternative objective formulations can be activated using the $SET OBJ <option> as described in Table 6.

Table 6: Objective Function Formulation Options

OBJ Option	Description
ALT	Uses modified capacity transfer coefficients that improve the independency of investment costs on period definitions.
AUTO (default)	TIMES automatically selects the objective function among the standard formulation or the ‘MOD’ alternative formulation according to the B(t) and E(t) parameters specified by the user. If those parameters comply with the assumptions used in the standard formulation, then the standard formulation is used, but if not then the alternative formulation ‘MOD’ is used.
LIN	Assumes linear evolution of flows and activities between Milestone years, but is otherwise similar to the ALT formulation.

OBJ Option	Description
MOD	Period boundaries B(t) and E(t) are internally set to be halfway between Milestone years, giving flexibility to set Milestone years to be other than the middle of each period. Investments in Cases I.1.a and I.1.b only of the objective function investment decision are spread somewhat differently across years.
STD	To ensure that the standard formulation is unconditionally used, even if the B(t) and E(t) parameters do not comply with the standard assumptions.

3.3.2 Objective function components

In addition to controlling how the objective function is assembled, as described in the previous section, the user has control of the handling of specific components of the objective functions, as described in Table 7.

Table 7: Objective Function Component Options

Option <value>	Description
DAMAGE <LP(default) /NLP/NO>	The TIMES model generator supports the inclusion of so-called damage costs in the objective function. By default, if such damage costs have been defined in the model input data, they are also automatically included in the objective function in linearized form (LP). However, if the user wishes the damage costs to be included in the solution reporting only, the DAMAGE control variable can be set to 'NO'. Non-linear damage functions can be requested by setting the control variable to 'NLP'. See Part II, Appendix B, for more on the damage cost function extension.
OBJANN <YES>	Used for requesting a period-wise objective formulation, which can be used e.g. together with the MACRO decomposition method for enabling the iterative update of the period-wise discount factors (See the documentation titled *Macro MSA*, on the MACRO Decomposition Algorithm, for details).

Option <value>	**Description**
OBLONG <YES/NO>	In the STD (standard) and MOD (alternative) objective function formulations discussed in Table 6 the capacity-related costs are not completely synchronized with the corresponding activities, which may cause distortions in the accounting of costs. This switch causes all capacity-related cost to be synchronized with the process activities (which are assumed to have oblong shapes), thereby eliminating also the small problems in salvaging that exist in the STD and MOD formulations. Due to the obvious advantages of using this setting, the OBLONG setting is activated by default whenever the MOD formulation is used. However, for backwards compatibility, one can disable it by adding the explicit setting $SET OBLONG NO in the run file. Using the OBLONG setting can be recommended also with the STD and AUTO settings. It can even be used with the ALT and LIN settings, but that is not recommended.
MIDYEAR <YES>	In the standard objective formulation, both the investment payments and the operating cost payments are assumed to occur at the beginning of each year within the economic/technical lifetime of technologies. This also means that the so-called annuities of investment costs are calculated using the following formula, where r is the discount rate (see Part II, Section 6.2 for more on the objective function): $CRF = (1-(1+r)^{-1})/(1-(1+r)^{-L})$ According to this formula, the interest costs are zero if the lifetime L of the technology is only one year, because the payments are assumed to occur at the beginning of each year. This approach is often called as ***beginning-of-year*** discounting. However, it leads to an underestimation of the costs, because in reality the investments can be paid back only after getting some income from the investment. To avoid such underestimation, the following formula for annuities is perhaps more commonly used: $CRF = r/(1-(1+r)^{-L})$ This second formula effectively assumes that the annual investment payments occur at the end of each year. This approach is often called as ***end-of-year*** discounting. As a good compromise between these two approaches, and highly recommended by many guidelines on good practices in cost evaluations[18], so-called ***mid-year discounting*** can additionally be used. See Section 6.2.12 of Part II for more information about mid-year discounting.

[18] For example, by the U.S. government: http://www.whitehouse.gov/omb/circulars/a094/a094.html

Option <value>	Description
DISCSHIFT	As a generalization to the MID_YEAR setting, alternate time-of-year discounting, including the end-of-year discounting mentioned above, can be achieved by using the DISCSHIFT control variable. The control variable should be set to correspond to the amount of time (in years) by which the discounting of continuous streams of payments should be shifted forward in time, with respect to the beginning of operation. Setting it to the value of 0.5 would be equal to the setting $SET MID_YEAR YES, and setting it to the value of 1.0 would be equal to end-of-year discounting, as follows: $SET DISCSHIFT 1
VARCOST <LIN>	The standard dense interpolation and extrapolation of all cost parameters in TIMES may consume considerable amounts of memory resources in very large models. In particular, the variable costs, which may also be to a large extent levelized onto a number of timeslices, usually account for the largest amount of cost data in the GAMS working memory. If desired, TIMES can be advised to interpolate and extrapolate the variable cost parameters only sparsely for the Milestone years. The values at the intermediate years will then be derived "*on the fly*", by piecewise linear interpolation, and will not be stored in the GAMS memory. This option may thus be useful when running very large models on computers with limited memory.

3.4 Stochastic and sensitivity analysis controls

3.4.1 Stochastics [STAGES]

The stochastic mode of TIMES can be activated with the STAGES control variable, by using the following setting:

```
$SET STAGES YES
```

This setting is required for using the multi-stage stochastic programming features of TIMES. It can also be used for enabling sensitivity and tradeoff analysis features. See Part I for more details on stochastic programming and tradeoff analysis in TIMES.

3.4.2 Sensitivity [SENSIS]

Many useful sensitivity and tradeoff analysis features are available in TIMES, and they can be enables by activating the stochastic mode of TIMES (see above). However, such sensitivity and tradeoff analyses are often based on running the model in a series of cases that differ from each other in only in a few parameter values. In such cases the so-called warm start features can usually significantly speed up the model solution in the successive runs.

The use of the warm start facilities can be automatically enabled in sensitivity and tradeoff analysis by using the following setting instead of $SET STAGES YES:

```
$SET SENSIS YES
```

As the variables then remain the same, warm start is automatically enabled by GAMS according to the BRATIO value (BRATIO can be set in VEDA-FE, or added manually to the ANSWER GEN template as a $OPTION BRATIO=1; as well).

See the documentation on stochastic programming and tradeoff analysis in TIMES for more information on the use of this switch. The documentation is available at the ETSAP site: http://www.iea-etsap.org/web/documentation.asp

3.4.3 Hedging recurring uncertainties [SPINES]

For modeling recurring uncertainties, such as hydrological conditions or fuel-price volatilities, the stochastic mode can be activated also in such a way that the SOW index will be inactive for all capacity-related variables (VAR_NCAP, VAR_CAP, VAR_RCAP, VAR_SCAP, VAR_DRCAP, VAR_DNCAP). This modification to the standard multi-stage stochastic formulation makes it possible to use the stochastic mode for hedging against recurring uncertainties, and for finding the corresponding optimal investment strategy.

This variant of the stochastic mode can be activated by using the following control variable setting:

```
$SET SPINES YES
```

In addition, under the SPINES option all the remaining equations that define dynamic or cumulative relationships between variables can additionally be requested to be based on the expected values instead of imposing the inter-period equations separately for each SOW. Doing so will ensure that the uncertainties represented by the SOW-indexed variables will be independent in successive periods. This further model simplification can be requested by using the SOLVEDA switch, as follows:

```
$SET SOLVEDA 1
```

In addition, under the SPINES option this SOLVEDA setting will, for now, also cause all the results for the activities and flows to be reported on the basis of the expected values only, and not separately for each SOW. After all, the recurring uncertainties are rather aleatory by nature, and therefore the user should probably be most interested in the optimal investment strategy, and only in the average or normal year results for the activities and flows. If requested, an option to produce results for all SOWs even under the period-independent variant can later be added.

Unlike the basic stochastic option STAGES, the SPINES option may be used also together with the time-stepped mode (see Section 3.5.2).

The SPINES control variable is available only in TIMES versions 3.3.0 and above, and should currently be considered ***experimental*** only.

3.5 Controls for time-stepped model solution

3.5.1 Fixing initial periods [FIXBOH]

The purpose of the FIXBOH option is to bind the first years of a model run to the same values determined during a previous optimization. The approach first requires that a reference case be run, and then by using FIXBOH the model generator sets fixed bounds for a subsequent run according to the solution values from the reference case up to the last Milestone year less than or equal to the year specified by the FIXBOH control variable. The FIXBOH control has to be used together with the LPOINT control variable, in the following way:

```
$SET FIXBOH 2050
$SET LPOINT <run_name>
```

Here, the value of FIXBOH (2050) specifies the year, up to which the model solution will be fixed to the previous solution, and the value of LPOINT (run_name) specifies the name of the previous run, from which the previous solution is to be retrieved. Consequently, either a full GDX file or a GAMS "point file" (see section 3.8) from the previous run should be available. If no such GDX file is found, a compiler error is issued. The Milestone years of the previous run must match those in the current run.

As a generalization to the basic scheme described above, the user can also request fixing to the previous solution different amounts of first years according to region. The region-specific years up to which the model solution will be fixed can be specified by using the TIMES REG_FIXT(reg) parameter. The FIXBOH control variable is in this case treated as a default value for REG_FIXT.

Example: Assume that you would like to analyze the 15-region ETSAP TIAM model with some shocks after the year 2030, and you are interested in differences in the model solution only in regions that have notable gas or LNG trade with the EU. Therefore, you would like to fix the regions AUS, CAN, CHI, IND, JPN, MEX, ODA and SKO completely to the previous solution, and all other regions to the previous solution up to 2030.

In the RUN file you should specify the control switches described above:

```
$SET FIXBOH 2030
$SET LPOINT <run_name>
```

In a model DD file you should include the values for the REG_FIXT parameter:

```
PARAMETER REG_FIXT /
AUS    2200,  CAN  2200, CHI   2200, IND   2200
JPN    2200, MEX   2200, ODA   2200, SKO   2200
/;
```

3.5.2 Limit foresight stepwise solving [TIMESTEP]

The purpose of the TIMESTEP option is to run the model in a stepwise manner with increasing model horizon and limited foresight. The TIMESTEP control variable specifies the number of years that should be optimized in each solution step. The total model horizon will be solved by successive steps, so that in each step the periods to be optimized are advanced further in the future, and all periods before them are fixed to the solution of the previous step. Figure 17 illustrates the step-wise solution approach.

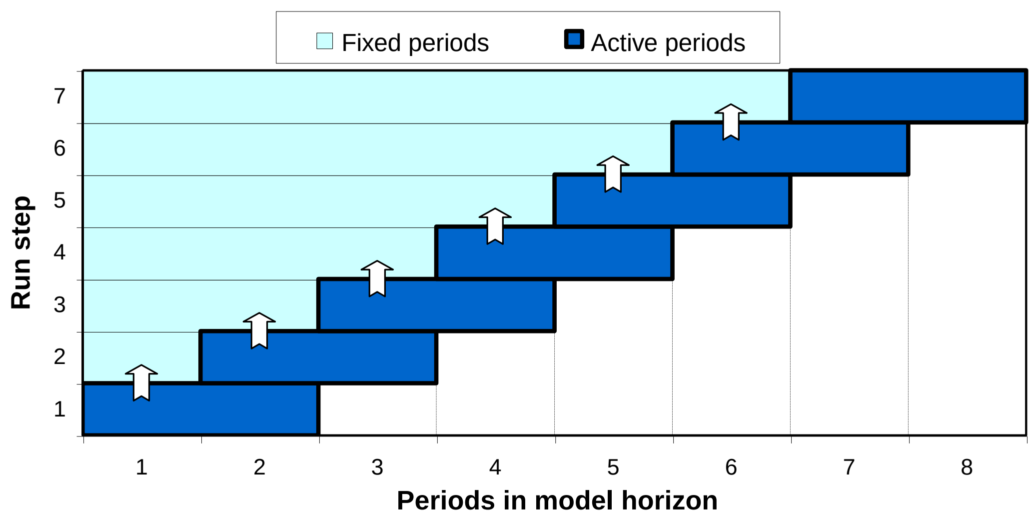


Figure 17: Sequence of Optimized Periods in Time-stepped Solution

The amount of overlapping years between successive steps is by default half of the active step length (the value of TIMESTEP), but it can be controlled by the user by using the TIMES G_OVERLAP parameter. Consequently, the specifications that can be used to control a stepped TIMES solution are the following:

```
$SET TIMESTEP 20              (specified in the run file)
PARAMETER G_OVERLAP / 10 /;   (specified in a DD file)
```

In this example, the TIMESTEP control variable specifies the active step length of each successive solution step (20 years), and the G_OVERLAP parameter specifies the amount of years, by which the successive steps should overlap (10 years). They may be set in the VEDA-FE Control Panel, like almost all control switches, or in ANSWER by manually adding the parameter to the Run GEN Template.

Because the time periods used in the model may be variable and may not always exactly match with the step-length and overlap, the actual active step-lengths and overlaps may somewhat differ from the values specified. At each step the model generator tries to make a best match between the remaining available periods and the prescribed step length. However, at each step at least one of the previously solved periods is fixed, and at least one remaining new period is taken into the active optimization in the current step.

3.6 TIMES extensions

3.6.1 Major formulation extensions

There are several powerful extensions to the core TIMES code that introduce advanced modeling features. The extension options allow the user to link in additional equations or reporting routines to the standard TIMES code, e.g. the DSC extension for using lumpy investments. The entire information relevant to the extensions is isolated in separate files from the standard TIMES code. These files are identified by their extensions, e.g. ***.DSC** for lumpy investments or ***.CLI** for the climate module. The extension mechanism allows the TIMES programmer to add new features to the model generator, and test them, with only minimal hooks provided in the standard TIMES code. It is also possible to have different variants of an equation type, for example of the market share equation, or to choose between different reporting routines, for example adding detailed cost reporting. The extension options currently available in TIMES are summarized in Table 8.

VEDA-FE Case Manager and ANSWER Run Model Options form along with the GEN template will both set the appropriate switches and augment the initialization calls, as described in Table 8 (unless noted otherwise), with the user being fully responsible to provide the necessary data for each extension option employed in a run.

Table 8: TIMES Extension Options

Extension	Description
CLI	The climate module estimates change in CO_2 concentrations in the atmosphere, the upper ocean including the biosphere and the lower ocean, and calculates the change in radiative forcing and the induced change in global mean surface temperature. It is activated with the following setting in the <case>.run file: `$SET CLI YES` See Parts I–II for more information on the use of the Climate Module and this switch.

<table>
<tr><th>Extension</th><th>Description</th></tr>
<tr><td>DSC</td><td>Option to use lumpy investment formulation. Since the usage of the discrete investment options leads to a Mixed-Integer Programming (MIP) problem, the solve statement in the file solve.mod is automatically altered by the user shell. To activate this extension manually, the following control switch needs to be provided in the <Case>.run file :
<code>$SET DSC YES</code>
See Part I, Chapter 10 for more information on the use of Lumpy Investment.</td></tr>
<tr><td>ETL</td><td>Option to use endogenous technology learning formulation. Since the usage of this option leads to a Mixed-Integer Programming (MIP) problem, the solve statement in the file solve.mod is automatically altered by TIMES. To activate this extension manually, the following control switch needs to be provided in the <Case>.run file, as follows:
<code>$SET ETL YES</code>
See Parts I–II for more information on the use of Endogenous Technology Learning.</td></tr>
<tr><td>MACRO</td><td>Option to use the MACRO formulation. Since the usage of the MACRO options leads to a Non-linear Programming (NLP) problem, the solve statement in the file solve.mod has to be altered. To activate this extension manually, the $SET MACRO <value> control switch needs to be provided in the <Case>.run file, with the following valid values:
• YES – activate the integrated MACRO algorithm
• MSA – activate Macro decomposition algorithm (MSA)
• CSA – activate the calibration algorithm for MSA
See the separate MACRO documentation for more on using this option.</td></tr>
<tr><td>RETIRE</td><td>The RETIRE control variable can be used for enabling early and lumpy retirements of process capacities. The valid switch values for this control variable are shown below:
• NO - Disables all early and lumpy retirements;
• LP - Enables continuous early retirements for all those processes that are included in the set PRC_RCAP(r,p);
• MIP - Enables early retirements for the processes that are included in the set PRC_RCAP(r,p), and additionally enables the retirements to be lumpy for those of these processes that also have RCAP_BLK (the lumpy block size) defined, and
• YES - Enables early retirements for any processes that have at least one instance of the parameter RCAP_BND defined. In this variant, activating lumpy retirements for those processes that have also RCAP_BLK defined</td></tr>
</table>

Extension	Description
	requires that the setting $SET DSC YES is used as well. Consequently, when using the $SET RETIRE YES switch, using the set PRC_RCAP is not needed at all (and it will have no effect). See Part II for more information on the use of the Early Retirement feature.
VDA	The VDA control variable can be used to enable the VDA pre-processor extension of TIMES, which implements new features and handles advanced parameters specified by VEDA-FE/ANSWER that are transformed into their equivalent TIMES core parameters to make specification easier (e.g., VDA_FLOP becomes FLO_FUNC/FLO_SUM), with the following setting: `$SET VDA YES` The VDA extension is always automatically enabled by both VEDA-FE and ANSWER. See separate documentation titled *New Parameters under VEDA* (TIMES-VDA.pdf) on the new attributes available under the VDA extension.

3.6.2 User extensions

Besides the core extensions discussed in the previous section, the model management system allows user extensions to be introduced for extended pre-processing of advanced parameter specifications that make the specification of (complex) input parameters much simpler, and to refine features describing technology operations (e.g., for CHPs).

The user extension(s) that are to be included in the current model run need to be activated in the <case>.run file, and passed to inimty.mod, e.g.:

```
$BATINCLUDE initmty.mod IER FIA
```

As shown in this example, it is possible to add several extension in the $BATINCLUDE line above at the same time. In this case two user extensions, IER and FIA, are incorporated with the standard TIMES code.

The GAMS source code related to an extension <ext> has to be structured by using the following file structure in order to allow the model generator to recognize the extension[19] (see also Section 2.2). The placeholder <ext> stands for the extension name, e.g. CLI in case of the climate module extension.

- **initmty.<ext>**: contains the declaration of new sets and parameters, which are only used in the context of the extension;

[19] This structure is only of interest for those modellers who want to programme their own extensions. The modeller who uses an extension in his model does not need to know these programming details.

- **init_ext.<ext>**: contains the initialization and assignment of default values for the new sets and parameters defined in initmty.<ext>;
- **prep_ext.<ext>**: contains primarily calls to the inter-/extrapolation routines (prepparm.mod, fillparm.mod);
- **pp_prelv.<ext>**: contains any preprocessing after inter-/extrapolation but before levelizing;
- **ppm_ext.<ext>**: contains any preprocessing after levelizing of standard parameters but before calculation of equation coefficients; it might contain calls to levelizing routines for the new input parameters implemented in the extension;
- **coef_ext.<ext>**: contains coefficient calculations used in the equations or reporting routines of the extension;
- **mod_vars.<ext>**: contains the declaration of new variables;
- **equ_ext.<ext>**: contains new equations of the extension;
- **mod_ext.<ext>**: adds the new defined equations to the model;
- **rpt_ext.<ext>**: contains new reporting routines.

Not of all these files have to be provided when developing a new extension. If for example no new variables or no new report routines are needed, these files can be omitted.

An example of a user extension is the IER extension included in the TIMES distribution. It contains several extensions to the equation system introduced specifically for the modelling needs by Institute for Energy Economics and the Rational Use of Energy (IER, University of Stuttgart), such as market/product share constraints, and backpressure/condensing mode full load hours.

3.7 The TIMES reduction algorithm

The motivation of the reduction algorithm is to reduce the number of equations and variables generated by the TIMES model, reducing memory usage and solution time. Since there is no downside the having the TIMES reduction algorithm applied by the pre/post-processors, it is the default set via VEDA/ANSWER.

An example for a situation where model size can be reduced is a process with one input and one output flow, where the output flow variable can be replaced by the input variable times the efficiency. Thus the model can be reduced by one variable (output flow variable) and one equation (transformation equation relating input and output flow).

3.7.1 Reduction measures

The effects arising from activating the reduction algorithm are each described below.

1. Process without capacity related parameters does not need capacity variables:

- No capacity variables **VAR_CAP** and **VAR_NCAP** created.
- No **EQL_CAPACT** equation created.

2. Primary commodity group consists of only one commodity:
   - Flow variable **VAR_FLO** of primary commodity is replaced by activity variable.
   - No **EQ_ACTFLO** equation defining the activity variable created.
3. Exchange process imports/exports only one commodity:
   - Import/Export flow **VAR_IRE** can be replaced by activity variable (might not be true if exchange process has an efficiency).
   - No **EQ_ACTFLO** equation defining the activity variable created.
4. Process with one input and one output commodity:
   - One of the two flows has to define the activity variable. The other flow variable can be replaced by the activity variable multiplied/divided by the efficiency.
   - No **EQ_PTRANS** equation created.
5. An emission flow of a process can be replaced by the sum of the fossil flows multiplied by the corresponding emission factor:
   - No flow variables for the emissions created
   - No **EQ_PTRANS** equation for the emission factor.
6. Upper/fixed activity bound **ACT_BND** of zero on a higher timeslice level than the process timeslice level is replaced by activity bounds on the process timeslice level. Thus no **EQG/E_ACTBND** equation is created.
7. Process with upper/fixed activity bound of zero cannot be used in current period. Hence, all flow variables of this process are forced to zero and need not be generated in the current period. Also **EQ_ACTFLO** and **EQx_CAPACT** are not generated. If the output commodities of this process can only be produced by this process, also the processes consuming these commodities are forced to be idle, when no other input fuel alternative exists.
8. When a **FLO_FUNC** parameter between two commodities is defined and one of these two commodities defines the activity of the process, the other flow variable can be replaced by the activity variable being multiplied/divided by the **FLO_FUNC** parameter.
   - One flow variable is replaced.
   - No **EQ_PTRANS** equation for the **FLO_FUNC** parameter is created.

3.7.2 Implementation

To make use of the reduction algorithm one has to define the environment variable

```
$SET REDUCE YES/NO
```

in order to turn on/off the reduction. This environment variable controls in each equation where the flow variable occurs whether it should be replaced by some other term or not. If the control variable is not defined at all, the default is to make partial model reduction by eliminating unnecessary capacity variables and substituting emission flows only. This third option can thus be more useful if full model reduction is not wanted.

The possibility of reduction measures is checked in the file pp_reduce.red. If reduction is turned on, flow variables that can be replaced are substituted by a term defined in cal_red.red. The substitution expression for the import/export variable VAR_IRE is directly given in the corresponding equations. In addition the $control statement controlling the generation of the equations EQ_PTRANS, EQ_ACTFLO, EQx_CAPACT has been altered. Also bnd_act.mod has been changed to implement point 6 above.

To recover the solution values of the substituted variables, corresponding parameters are calculated in the reporting routines and are then written to the VEDA-BE file.

3.7.3 Results

The main solution and solver statistics for model runs of a USEPA9r-TIMES model with and without reduction algorithm are given in Table 9 for CPLEX (GAMSv24.4.1), using a call to the solver for Barrier for initial solve and Primal Simplex crossover to finish up.

Table 9: Reduction Model Comparison

Statistic	Reduce Not Set	Reduce=NO	Reduce=YES
Block / Single Equations	92 / 1,652,677	92 / 1,796,525	92 / 870,814
Block / Single Variables	14 / 2,429,348	14 / 2,564,039	14 / 1,645,631
Total Non-Zeros	7,432,490	8,048,546	5,853,371
Generation	49.499 SECONDS	62.681 SECONDS	45.802 SECONDS
Execution	102.462 SECONDS	115.394 SECONDS	95.972 SECONDS
Memory	2,075 MB	2,180 MB	1,957 MB
Iteration Count	126	116	110
Objective Value	88503425.2162	88151566.0679	88503425.2162
Resource Usage / Solution Time	1323.824	2656.557	1320.111

Comparing the non-setting of REDUCE vs. REDUCE=YES the number of equations and variables in the reduction is around 47% lower than in the non-reduced case. Since the smaller number of equations and variables require less memory, the memory usage in the reduction run decreases by 6.4%. The solution time is only reduced slightly compared to the non-reduced model run.

Issues Using the Reduction Algorithm

- In some cases the reduced problem may produce an "optimal solution with unscaled infeasibilities".
- Shadow price of non-generated EQ_PTRANS equations are lost.
- Reduced cost of upper/fixed ACT_BND of zero are lost. If one needs this information, one should use a very small number instead, e.g. 1.e-5, as value for the activity bound.

3.8 GAMS savepoint / loadpoint controls

TIMES includes GAMS control variables that can be used to utilize the GAMS savepoint and loadpoint facilities. The savepoint facility makes it possible to save the basis information (levels and dual values of variables and equations) into a GDX file after model solution. The loadpoint facility makes it possible to load previously saved basis information from a GDX file and utilize it for a so-called warm start to speed up model solution.

The GAMS control variables that can be used for the savepoint and loadpoint features in TIMES models are SPOINT and LPOINT. These control variables are ***completely optional***, but can be set in the following ways as described in Table 10 if desired:

Table 10: Save/Load Restart Switches

Option	Description
SPOINT	
Not provided (default)	Does not save or load a restart point.
1 (or YES)	The final solution point from the model run should be saved in the file %RUN_NAME%_p.gdx, where %RUN_NAME% is the GAMS control variable that should always be set to contain the name of the current TIMES model run in the run file for the model.
2	The model generator should make an attempt to load the solution point from the file %RUN_NAME%_p.gdx, where %RUN_NAME% is the GAMS control variable that should always be set to contain the name of the current TIMES model run in the run file for the model. If the control variable LPOINT has additionally been set as well, this attempt will be made only if the loading from the file %LPOINT%_p.gdx fails.
3	Combines both of the functionalities of the settings 1 and 2.

LPOINT	
LPOINT filename	Indicates that the model generator should load the solution point from the file %LPOINT%_p.gdx. If the control variable SPOINT has additionally been set to 2 or 3, a subsequent attempt to load from %RUN_NAME%_p.gdx is also made if the loading from the file %LPOINT%_p.gdx fails.

In VEDA-FE the LPOINT can be set from the Case Manager by requesting the loading of a previously GDX, and in ANSWER by means of Run Model Restart files specifications, as shown in Figure 18.

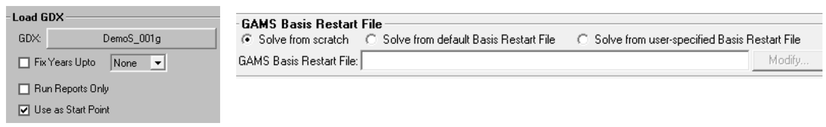


Figure 18 - Setting LPOINT

3.9 Debugging controls

By using the DEBUG control, the user can request dumping out all user/system data structures into a file, and turn on extended quality assurance checks. The switch is activated by means of:

$SET DEBUG YES

with actions performed according to the settings described in Table 11.

Table 11: Debug Switches

Switch <value>	Description
DUMPSOL YES	Dump out selected solution results into a text file (levels and marginals of VAR_NCAP, VAR_CAP, VAR_ACT, VAR_FLO, VAR_IRE, VAR_SIN, VAR_SOUT, VAR_COMPRD, EQG_COMBAL, EQE_COMBAL, EQE_COMPRD).
SOLVE_NOW NO	Only check the input data and compile the source code, but do not solve the model.
XTQA YES	Turn on extended quality assurance checks [this setting is automatically enabled whenever $SET DEBUG YES is used].

3.10 Controls affecting solution reporting

The various $<switch> <value> switches controlling reporting of the model results are summarized in Table 12.

Table 12: Solution Reporting Switches

Switch <value>	Description
ANNCOST LEV	Until TIMES v3.4.9, the values reported for each of these cost components have been calculated strictly for the associated Milestone year of a period. However this can result in investments made in other years within a period not being reflected, and for longer periods may not properly reflect changes in the other annual expenditures over that timeframe. A consequence of this is that it has not been possible to reconstruct the objective function value from the annualized costs reported. Additionally, these reported costs cannot be thought of as "representative" of the entire period, but only of the Milestone year. To redress this, from TIMES v3.5.0 the annual costs based upon the levelized costs over process lifetimes or periods can be requested. The various annualized cost report parameters are found in Table 13. In this way all expenditures during the period are captured and the total objective function can be reconstructed from the levelized annual costs with a very high accuracy (when using $SET OBLONG YES). There is also a new attribute Time_NPV, which gives the period-wise discount factors, and a UC tag = LEVCOST/COST indicating whether the annual costs reported for each scenario are levelized or not. That is, when said Attribute = LEVCOST for a scenario, then the annualized costs for said scenario represent the levelized average annual values.
BENCOST YES	TIMES includes also a basic benefit-cost reporting for new technologies. When the benefit-cost reporting is requested, the TIMES reporting attribute VAR_NCAPR includes the benefit-cost indicators listed in Table 14.
RPT_FLOTS COM / ANNUAL	Used for controlling the timeslices that will be used for reporting the levels of the TIMES flow variables. By default, the timeslices of the original TIMES flow variables are used also for reporting. However, in many cases it may be more desirable to have all the flow levels reported at the commodity timeslices (COM), or, for very large models, at the ANNUAL timeslice only. The RPT_FLOTS setting has no effect on the reporting of marginal costs for flows.
SOLANS YES	Produce the solution reports that can be imported into the ANSWER.

Switch <value>	Description
SOLVEDA YES / 1	Prepare the solution reporting values that are to be imported into the VEDA-BE. The standard setting is $SET SOLVEDA YES, which works with all TIMES extensions. Sometimes it may be useful to request that TIMES reports also the results from non-stochastic runs with an extra dummy SOW index ‘1’, such that the results can be imported into a database that contains results from both deterministic and stochastic runs. The inclusion of the extra index can be activated by the setting $SET SOLVEDA 1.
XTQA YES	Turn on extended quality assurance checks [this setting is automatically enabled whenever $SET DEBUG YES is used].

Table 13: Solution Cost Reporting Attributes

Attribute	Description
Cost_Act	Annual activity costs
Cost_Comx	Annual commodity taxes/subsides
Cost_Els	Annual loss of consumer surplus (for elastic demand)
Cost_Flo	Annual flow costs (including import/export prices)
Cost_Flox	Annual flow taxes/subsidies
Cost_Fixx	Annual fixed operating and maintenance taxes/subsidies
Cost_Fom	Annual fixed operating and maintenance costs
Cost_Inv	Annual investment costs
Cost_Invx	Annual investment taxes/subsidies
Cost_ire	Annual implied costs of endogenous trade
Cost_Salv	Salvage values of capacities at EOH+1
Reg_ACost	Regional annual costs by component

Table 14: BENCOST Reporting Attributes

Attribute	Description
COST	the total unit costs of VAR_NCAP (in terms of investment costs)
COST	the total unit costs of VAR_NCAP (in terms of investment costs)
CGAP	competitiveness gap (in terms of investment costs), obtained directly from the VAR_NCAP marginals (and optional ranging information)
GGAP	competitiveness gap (in terms of investment costs), obtained by checking also

Attribute	Description
	the VAR_ACT, VAR_FLO and VAR_CAP marginals, in case VAR_NCAP happens to be basic at zero
RATIO	benefit / cost ratio, based on CGAP
GRATIO	benefit / cost ratio, based on GGAP
RNGLO	ranging information (LO) for VAR_NCAP (when CPLEX ranging is activated; in terms of investment costs)
RNGUP	ranging information (UP) for VAR_NCAP (when CPLEX ranging is activated; in terms of investment costs

For the BENCOST report, all of the absolute indicators are expressed in terms of undiscounted investment costs (like those specified by NCAP_COST). For example, the competitiveness gap represents the amount of change in investment costs that would bring the technology competitive (the VAR_NCAP variable would enter the solution basis). Ranging information can only be reported when the CPLEX ranging option has been used. The ranging option can be activated by adding the following two lines into the CPLEX options file (CPLEX.OPT):

```
objrng VAR_NCAP
rngrestart timesrng.inc
```

When available, the LO ranging information is also used for calculating the competitiveness gap indicators, because the VAR_NCAP variables can occasionally be basic at zero, making the reduced cost information useless. In such cases the LO ranging value can be used to derive the amount of change required in the VAR_NACP cost coefficient to cause a change in the basis.

Various reporting options can also be set by specifying values for the RPT_OPT parameter. Although it is actually not a GAMS control variable, for completeness it is described here. Like the control switches, these options can be specified in the RUN file, but they can also be included in the DD files, if the user shell implements their use that way. Specifying the options in the RUN file can be done with any of the three following alternative ways:

- $SET RPT_OPT KEY1.N1 <value1>, KEY2.N2 <value2>, ….
- PARAMETER RPT_OPT / KEY1.N1 <value1>, KEY2.N2 <value2>, …. /;
- RPT_OPT('KEY1','N1') = <value1>; RPT_OPT('KEY2','N2') = <value2>; …

Here, KEY1, KEY2, … refer to the main option group and N1, N2, … refer to sub-groups within that group, as indicated in Table 15.

Table 15: RPT_OPT Options Settings

Option group	Sub-group	Value	Description
ACT	2	<0	Suppress reporting of activity marginals
FLO	1	>0	Report process flows at commodity TS level
FLO	3	>0	Report value flows by process (implies (FLO,1)=1)
FLO	5	>0	Report electricity supply by energy source
FLO	7	>0	Report process topology indicators
COMPRD	1	>0	Report VAR_COMPRD for all commodities
NCAP	1	<>0	Activate levelised cost calculation (see separate documentation for details)
OBJ	1	<>0	Split investment costs according to hurdle rate

3.11 Miscellaneous controls

Various other $<option> switches control miscellaneous aspects of a TIMES model run, as described Table 16.

Table 16: Miscellaneous Control Options Settings

Option <value>	Description
BOTIME / EOTIME <year>	These controls can be used for adjusting the total available time span of years available in the model. All years related to the data and model must lie between BOTIME and EOTIME, inclusive. The default for BOTIME ('Beginning of Time') is 1850 and the default for EOTIME ('End of Time') is 2200. [A large model may see slightly faster runtimes if the BO/EOTIME horizon is narrowed to that actually needed for the model run.]

Option <value>	Description
GDX_IREBND / GDX_IPRIC <file>	These control flags can be used to import bounds and prices on exogenous imports/exports from a previous run, and thereby override any user-defined bounds/prices. Only bounds and prices for such imports and exports flows are imported that were endogenous in the previous run but are exogenous for the current run. The first setting tells TIMES to import the flow-levels of imports and exports from the file 'boundfile.gdx', and use these levels as fixed bounds on the imports and exports in the current run (if they are exogenous in the current run and were endogenous in the earlier run). The second setting tells TIMES to import the marginal prices of imports and exports from the file 'pricefile.gdx', and define these prices on the imports and exports in the current run (if they are exogenous in the current run and were endogenous in the earlier run). The earlier run may have different Milestone years than the current run.
RELAX_PRC_CG <YES>	Used to relax the requirement that all genuine commodity groups that are used in process-related attributes have to be explicitly associated with the processes, using the set PRC_CG. All PRC_CG definitions can be omitted in the model when the setting is enabled.
RPOINT <YES>	Used for reproducing the solution of a previous run, without actually solving the model at all. It should be used together with the LPOINT control, which specifies the GDX file where the previous solution is retrieved. The model generator then only loads the solution and generates the reports.
SHELL <ANSWER>	Indicates the ANSWER-TIMES user shell is being used for running this TIMES models.
VALIDATE <YES>	A greatly simplified formulation of the objective function and capacity constraints, emulating the MARKAL model generator, may be requested – however, use of the VALIDATE control switch is discouraged.
VAR_UC <YES>	Used to enable or disable the explicit use of slack variables in user constraints. By default, no explicit slack variables are used and all the user constraints are either equalities or inequalities, depending on the bound type specified. However, if the slack variables are enabled, all the user constraints are defined as equality constraints, using bounds on the slack variables to define the actual type of the constraint. This can be useful for e.g. more efficient specification of ranges, and is required when using the stochastic or sensitivity modes.

Option <value>	Description
VINTOPT <1 / 2>	Any technology characteristics defined for a vintaged process describe the characteristics of new capacity installed in the year specified. However, in TIMES the characteristics at the Milestone year are by default used for all the capacity installed in the corresponding period, which can lead to accelerated technology development, depending on the lengths of periods. To avoid such distortions caused merely by period length definitions setting VINTOPT 1 is used, all vintaged characteristics of technologies are automatically adjusted so that the average characteristics of new capacity installed for each period correspond to the original data. When the setting VINTOPT 2 is used, all vintaged processes are modeled using a different approach, which preserves the average characteristics of new capacity installed for each period, as originally defined by the TIMES attributes. The VINTOPT control variable is currently for experimental use only.
WAVER <YES>	Usually the TIMES model generator interpolates the user-defined time-series data only for the Milestone years, and then uses the value at the Milestone year as a representative value for the whole period. An important exception to this common rule are the cost parameters, which are all interpolated densely, and are thus always fully taken into account. However, in some cases it might be desirable to have some other parameters densely interpolated, such that the calculated weighted average over each projection period would be used as the representative value for the period, instead of the value at the Milestone year. Perhaps the most suitable candidates for applying this kind of an interpolation method are parameters representing projected absolute values, such as demands or remaining residual capacities. There is a switch for activating the Weighted Average Interpolation method described above, to be applied for the demand projections (COM_PROJ) and residual capacities (PRC_RESID), as well as the NCAP_PASTI parameters reflecting the available capacity of the installation period.

Energy Technology Systems Analysis Programme

http://iea-etsap.org/index.php/documentation

Documentation for the TIMES Model

PART IV

October 2016

Authors: M. Gargiulo
K. Vailancourt
R. De Miglio

General Introduction

This documentation is composed of five Parts.

Part I provides a general description of the TIMES paradigm, with emphasis on the model's general structure and its economic significance. Part I also includes a simplified mathematical formulation of TIMES, a chapter comparing it to the MARKAL model, pointing to similarities and differences, and chapters describing new model options.

Part II constitutes a comprehensive reference manual intended for the technically minded modeler or programmer looking for an in-depth understanding of the complete model details, in particular the relationship between the input data and the model mathematics, or contemplating making changes to the model's equations. Part II includes a full description of the sets, attributes, variables, and equations of the TIMES model.

Part III describes the organization of the TIMES modeling environment and the GAMS control statements required to run the TIMES model. GAMS is a modeling language that translates a TIMES database into the Linear Programming matrix, and then submits this LP to an optimizer and generates the result files. Part III describes how the routines comprising the TIMES source code guide the model through compilation, execution, solve, and reporting; the files produced by the run process and their use; and the various switches that control the execution of the TIMES code according to the model instance, formulation options, and run options selected by the user. It also includes a section on identifying and resolving errors that may occur during the run process.

Part IV provides a step-by-step introduction to building a TIMES model in the VEDA-Front End (VEDA-FE) model management software. It first offers an orientation to the basic features of VEDA-FE, including software layout, data files and tables, and model management features. It then describes in detail twelve Demo models (available for download from the ETSAP website) that progressively introduce VEDA-TIMES principles and modeling techniques.

Part V describes the VEDA Back-End (VEDA-BE) software, which is widely used for analyzing results from TIMES models. It provides a complete guide to using VEDA-BE, including how to get started, import model results, create and view tables, and create and modify user sets, and step through results in the model Reference Energy System. It also describes advanced features and provides suggestions for best practices.

PART IV: Getting Started with the VEDA-TIMES Demo Models

TABLE OF CONTENTS FOR PART IV

1 Introduction

This Part of the TIMES documentation provides a step-by-step introduction to building a TIMES model in the VEDA-Front End (VEDA-FE) model management software, using a series of twelve DemoS models (available for download from the ETSAP website) to progressively demonstrate VEDA-TIMES principles and modeling techniques. The remainder of Section 1 describes how to access and set up the TIMES DemoS models. Section 2 provides an orientation to the basic features of VEDA-FE, including software layout, commonly used data files and tables, and model management features. Section 3 then walks through the twelve DemoS models, providing for each a summary of the VEDA-TIMES features and model attributes introduced, a detailed guide to the templates and tables used, and a look at the model results.

1.1 Downloading and setting up the DemoS models

The complete set of VEDA-TIMES DemoS models is available, along with all five Parts of the TIMES documentation, on the ETSAP Documentation web page (http://www.iea-etsap.org/index.php/documentation) under 'VEDA-TIMES Demo Models'. You will also need VEDA-FE and Back End (BE) installed in order to follow along with this manual. VEDA Installation instructions are available at http://support.kanors-emr.org/.

The DemoS model zip folder include two sub-folders:

- ETSAP_DemoS_VFE. In this folder there are 12 subfolders, one for each of the twelve Demos. This folder should be pasted into your VEDA_Models folder (C:\VEDA\VEDA_Models, if you did not change the path during installation).
- DemoS_VBE. This folder contains a database with predefined tables for analyzing model results. This folder should be pasted into your VEDA-BE Databases folder (C:\VEDA\Veda_BE\Databases, if you did not change the path during installation).

To open the first DemoS from VEDA-FE and set up the VEDA-BE database:

- Launch VEDA-FE, and use the VEDA Navigator (described in Section 2.2) to browse to the folder in which the DemoS_001 is stored. (C:\VEDA\VEDA_Models\DemoS_001, if you have followed the default installation).
    - At this point VEDA-FE will load DemoS_001 into the Navigator.
- Launch VEDA-BE, select **Open** from the **File** menu, and browse to the folder where you have stored the DemoS_VBE database.
    - When the DemoS_VBE database is loaded, the list of pre-defined tables can be seen under **Table definition** at the top left of the main window. To view a particular table, scroll down/up the list and select one table, then click the **View Table(s)** button. The table will open with a pre-defined layout than can be modified in a very flexible manner. Note that not all the tables can be used for the first demo steps, in which only a few simple results will be available. If a VEDA-BE table is inconsistent or empty, you will get a pop up message saying that table is empty.

2 Introduction to VEDA Front End

This section provides a brief introduction to using VEDA-FE and VEDA Excel template workbooks for building, browsing, and running a TIMES model. VEDA-FE is used to facilitate TIMES model building based on a modular approach and heavily reliance on flexible Excel workbooks integrated into a core database visible via tabular and diagrammatic browsing tools. It is also used to develop and manage model runs. The main tools available in VEDA-FE are:

- A **Navigator** to oversee the management of the Excel workbooks.
- A **Browser** to view all model data (based on filter and search facilities).
- **Reference energy system** (RES) diagramming and **Commodity/Process Masters** with data views.
- A **Case Manager** window for composing and submitting model runs.

These are described in Section 2.4, following a description of the VEDA-FE template folder structure, file types, and tables used to create model input. VEDA-FE is complimented by VEDA Back End (BE) for analysis of model run results. (See Part V of this documentation for more on VEDA-BE.)

2.1 Folders and subfolders

All VEDA-TIMES model input data is organized in Excel workbooks (or files). VEDA-FE then integrates information from all of these workbooks into a single database to generate a TIMES model. The models managed by VEDA-FE are normally stored in a specific folder (by default \VEDA\VEDA_Models). Within this folder, there is a sub-folder for each individual model a user is working with, including all of the VEDA-TIMES Demo Models ((\VEDA\VEDA_Models\DemoS_001, etc.). The sub-folder structure is identical for each individual model (Figure 1, left side) and includes:

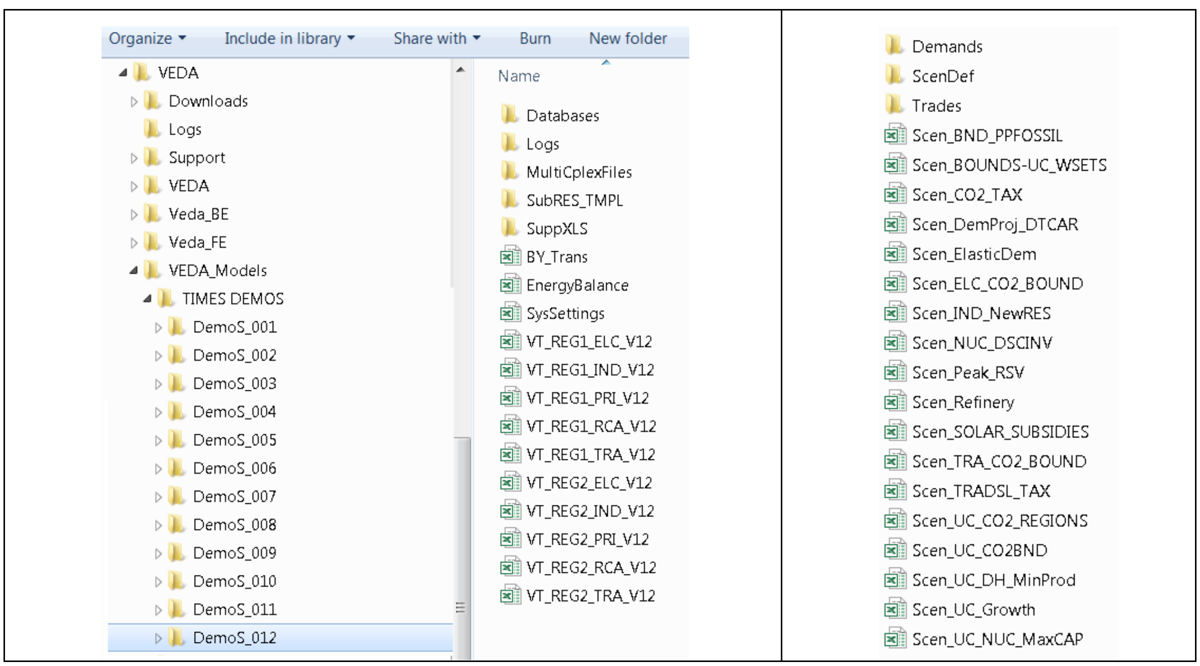


Figure 1. Sub-folders structure for each VEDA-FE model

- The B-Y Templates, the SysSettings files and the BY_Trans file.
- A sub-folder (**SubRES_TMPL**) to store all SubRES files and associated transformation files.
- A sub-folder (**SuppXLS**) to store all scenario files, as well sub-folders for trade files (**Trades**) and demand files (**Demands**) (Figure 1, right side).
- The **Logs** folder, which provides a location for VEDA-FE to write a variety of log files, including QA_Checks, error messages, and run summaries. Its contents are accessible via the **LOGs** button in the Case Manager.
- Users are not concerned with the other sub-folders.

2.2 VEDA-FE Navigator and types of workbook files

VEDA-FE opens displaying the VEDA-Navigator (Figure 2), which provides a comprehensive view of all the files in the various folders managed by VEDA-FE for the current model. The specific folder pathway associated with the active model is displayed on the top bar of the VEDA-Navigator form (in Figure 2, MODEL: C:\VEDA\VEDA_Models\TIMES DEMOS\DemoS_012). Clicking on this bar will open a window where the user can change the active model, by selecting a previously opened model from the list shown, or opening a new one by clicking the **New** button and navigating to the path of the new model folder.

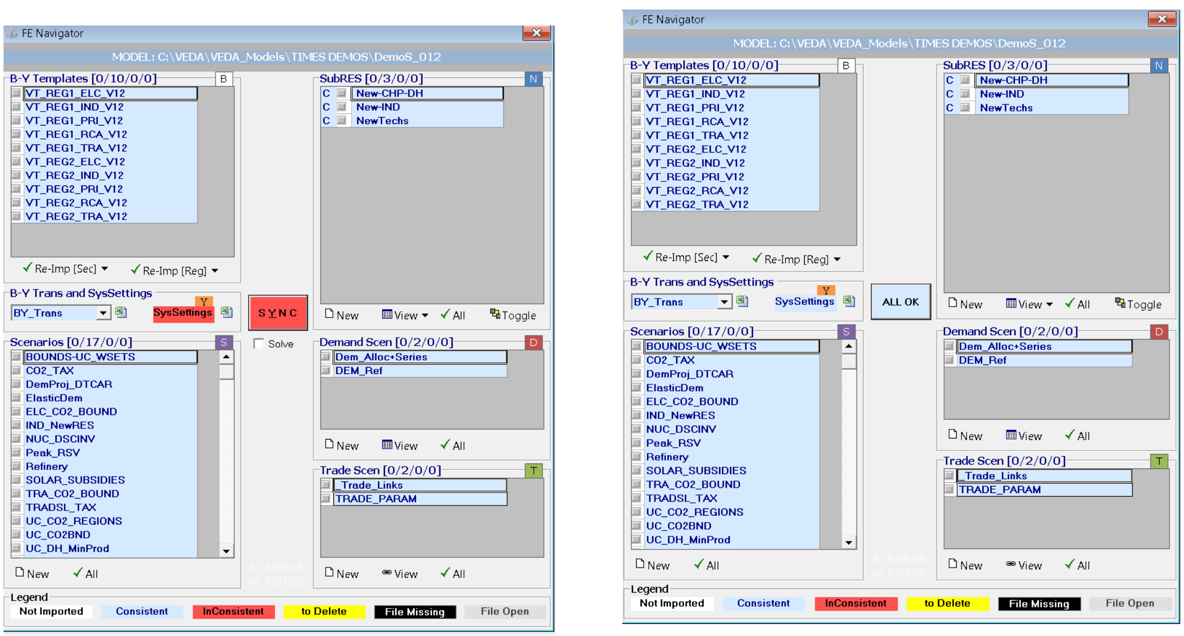


Figure 2. The VEDA-FE Navigator

The VEDA-Navigator is the main vehicle for accessing, importing, and coordinating the various files that make up a model. Its front screen is divided into sub-windows according to the various types of files managed by VEDA-FE:

- **B-Y (Base Year) Templates**: These templates can be used to set up the base-year structure of the model (existing process stock and the base-year end-use demand levels), such that the overall energy flows reflect the energy balance. In other words, the start year of the model can be calibrated to the energy balance in the B-Y Templates. The B-Y templates are named as VT_<workbook name>_<sector>_<Version> (e.g. VT_REG_PRI_V1). The number of B-Y templates and their names depend on both the model structure (e.g., the number of regions and sectors) and the organisation of the input data (e.g., how many regions and sectors in each file). The B-Y templates are introduced in DemoS_001 (Section 3.1) and are modified throughout the evolution of the 12 Demo steps.

- **SysSettings**: This file is used to declare the very basic structure of the model including regions, time slices, start year, etc. It also contains some settings for the synchronization process and can include some additional information. There is only one such file; it has a fixed name that stands for System Settings. The SysSettings file is described in Section 3.1.1.
- **BY_Trans**: These transformation files are used to update the information included in the B-Y templates (update existing values for existing attributes) and/or to insert new information (insert new attributes for existing processes) in the B-Y templates. They work like a scenario file (described below), but the rule-based filters and the update/insert changes apply only to those processes and commodities already existing in the B-Y templates. The BY_Trans file is introduced in DemoS_009 (Section 3.9.1.2).
- **Scenarios**: Scenario files are used to update existing information and/or to insert new information in any part of the RES, including B-Y templates, SubRES files, and Trade files. They are also used to include any additional user constraints in the model. The naming convention is: Scen_<scenario name>. These files can only manipulate (insert or update) information associated with previously declared RES components. New commodities and processes may not be added via scenario files, only new attributes. Scenario files are introduced in DemoS_004 (Section 3.4.3). Several different applications of scenario files are illustrated through the remainder of the Demos.
- **SubRES**: The SubRES files are used to introduce new processes and commodities in the RES that are not part of the B-Y templates. However, while the B-Y templates are region-specific, the SubRES are region independent. For each SubRES file, there is a corresponding transformation (Trans) file allowing the introduction of regional specificity of process attributes, including the availability (or not) of processes in each region. The naming conventions are: SubRES_<name> and SubRES_<name>_Trans. SubRES files are introduced in DemoS_006 (Section 3.6.3).
- **Demand Scen**: The demand files include all the information necessary to project end-use demands for energy services in each region, such as macroeconomic drivers and sensitivity series. Multiple demand files may be used, to model different demand growth scenarios for instance, and the naming convention is: ScenDem_<scenario name>. This section of the Navigator also contains a single file permitting assignment of a demand driver as well as a sensitivity (or elasticity) series each end-use demand to its driver in each region: Dem_Alloc+Series. Demand files and tables are described in DemoS_010 (Section 3.10.1).
- **Trade Scen**. The trade files include all of the attribute specifications for the trade processes. Multiple trade files may be used, to model different trade scenarios or for different commodities. The naming convention is: ScenTrade_<scenario name>. This section of the Navigator contains also a single file in which all uni- or bilateral trade links

between regions are declared: ScenTrade__Trade_Links. Trade files are introduced in DemoS_005 (Section 3.5.3).

The VEDA-Navigator enables easy access to any of the Excel files constituting the currently open model. Double-clicking directly on any file name (or the Excel icon next to it, in the case of the BY_Trans and SysSettings files) will open that file in Excel, while clicking on the bar above each section of the Navigator will open the associated folder in Windows Explorer. (For example clicking on **SubRES** in Figure 2 will open the folder VEDA\VEDA_Models\DemoS_012 \SubRES_TMPL).

The VEDA-Navigator also provides feedback as to the status of the various files and the integrated database managed by VEDA-FE. The consistency of the files and database is immediately evident based upon whether the central button is marked as **SYNC** in red (as shown on the left side of Figure 2) or as **ALL OK** in blue (right side of Figure 2). The status of individual templates is indicated by their colors in the template lists, according to the legend at the bottom of the form. A file is shown as inconsistent (in red) when it has a newer date/time stamp than in the database. <u>Note</u>: you may need to do a **Refresh** (from the **Window** menu, or hit **F5**) to see the current status of the files after a recent change.

Hitting the **SYNC** button will synchronize all files in the application folder marked as inconsistent. You may force synchronization of other files by checking the checkbox next to their names before hitting **SYNC**.

2.3 VEDA-FE workbook tables

The VEDA-FE import program reads each sheet in each file in sequence, looking for VEDA-FE tables to be read, which are identified by table tags including the special character " ~ ". VEDA-FE tables must be separated from the rest of the worksheet, which may contain any other contents, by blank rows and columns. Rows and columns starting with the character " ***** " or with “\I:”, which stands for "ignore", are not read.

The most common types of tables are briefly described in this section. More information on how to use them for specific cases is shown in the sections associated with each step of the demo.

2.3.1 Basic tables needed for any model

The following tables are needed in any VEDA-TIMES model.

- Tables that exist only in the SysSetting file. (Section 3.1.1 describes how to use these tables).
    - **~BookRegions_Map** to declare the workbook name and the list of region names.
    - **~TimeSlices** to declare the time-slice resolution for the model.
    - **~StartYear** to declare the start year of the model.
    - **~ActivePDef** to declare the set of active periods.
    - **~TimePeriods** to declare the time horizon of the model for the ActivePdef.
    - **~ImpSettings** to define some settings for the synchronization process.
    - **~Currencies** to define a default currency for the whole model.
    - **~DefUnits** to define default units by activity, capacity, and commodity for each sector in the model.

- Commodity Definition Tables **(~FI_COMM)** for commodity declaration and definition. These tables can be used in BY, SubRES and SysSetting files. They are described further in Section 2.3.4.
- Process Definition Tables **(~FI_PROCESS)** for process declaration and definition. These tables can be used in BY, SubRES and SysSetting files. They are described further in Section 2.3.5.
- Flexible Import Tables **(~FI_T)** for topology and parameter definition. These tables can be used in BY and SubRES files. They are described further in Section 2.3.6.

2.3.2 Tables need for scenario and transformation files

The following tables can be used to improve and update the model in scenario files and transformation files.

- Transformation Insert Tables **(~TFM_INS)** in scenario and transformation files, used to define absolute values via additional parameters that were not defined in the base year templates.
- Transformation Update Tables **(~TFM_UPD)** in scenario and transformation files operate on existing data defined in previous scenarios. Updates are applied to seed values that are picked up from the closest alphabetically preceding scenario. As shown in Section 2.3.7, Insert and Update tables can use rules to pick out the processes and commodities whose data is to be adjusted.
- Transformation Direct Insert Tables **(~TFM_DINS)** are also used to insert data, but unlike in Insert tables, it is forbidden to define subsets of technologies using text/wildcards, and for each attribute all the required dimensions must be defined (no defaults). These tables can be useful when working with large, detailed source data tables, because VEDA-FE's processing of DINS tables is much faster than that of Insert tables.

2.3.3 Advanced tables

The following tables are special and/or advanced tables that can be used in different types of files to support users in model building.

- Special tags exist for emission commodity tables. With this type of table identifier the data are manipulated during the import process to provide for special calculations on emissions factors.
    - **~COMEMI** to link emissions to commodity consumption. An example on how to use this table is shown in Section 3.7.2.7.
    - **~COMAGG** to define an aggregated commodity TOTCO2.
- Fill tables in scenario files **(~TFM_FILL**) allow extraction of values from the rest of the model database for use in Update or Insert tables. An example is shown in Section 3.7.4.1.
    - The TFM_FILL table is also available in SubRES transformation file. The only difference is that it can only be populated with numbers from the BASE scenario.
    - The fill operation will color the Region cells upon processing to indicate the number of records found, as follow:
        - Blue color represents only one record found, and

- ➢ Purple color represents that more than one record was found for the specified parameter and its dimensions while filling the region value in the relevant row.
  - The user can specify whether multiple values are to be summed, averaged, or counted.
- Different tags exist for transformation tables indicating that the import process is different than from the standard input tables ~FI_T. With this type of table identifier the data are manipulated during the import process and not imported as provided. They are supported in the BY_Trans file, SubRES files, and all scenario files.
  - **~TFM_AVA** to declare the availability of processes in different regions.
- Special tables that exist only in the demand module:
  - **~DRVR_Allocation** to allocate a driver to each end-use demands.
  - **~Series** to define sensitivity and calibration series.
  - **~DRVR_Table** to define demand driver indexes (base-year =1).
- Special tables that exist only in the trade module:
  - **~TradeLinks** to declare uni- or bilateral trade links between regions.
- User constraints are identified with specific identifiers (**~UC_Sets:**)

2.3.4 Commodity definition tables ~FI_COMM

Commodity definition tables (~FI_Comm) are used to declare the non-numerical characteristics of commodities. The columns headers are fixed but their order can be changed. Each commodity needs to be declared (only) once in such a table as shown in Figure 3. They are supported in B-Y Templates, SubRES files, and the SysSettings template. Commodities that are declared in a SubRES can only be used in that SubRES. Care must be taken that commodities are declared only once, as problems can arise if the same commodity is declared twice with conflicting attributes, such as different time slice levels. In large complex models, therefore, a best practice would be to declare them in a single template location only, such as the SysSettings template.

~FI_Comm								
Csets	Region	CommName	CommDesc	Unit	LimType	CTSLvl	PeakTS	Ctype
*Commodity Set Membership	Region Name	Commodity Name	Commodity Description	Unit	Sense of the Balance EQN.	Timeslice Level	Peak Monitoring	Electricity Indicator
NRG		COA	Solid Fuels	PJ				

Figure 3. How to use ~FI_COMM (table from DemoS_001)

The valid column headers for a commodity table ~FI_COMM are described in Table 1.

Table 1. Valid column headers for a commodity table ~FI_COMM

Header	Description
Csets*	The sets to which commodities belong. Valid entries are: NRG (energy), MAT (material), DEM (demand service), ENV (emissions) and FIN (financial). These declarations are inherited until the next one is encountered. In this example, COA (Solid Fuels) is an energy commodity (NRG).
Region*	The region name. By default, it is applied to all regions of the model when not specified. The region designation is used only in the B-Y templates and not allowed in SubRES.

CommName	The commodity name (COA).
CommDesc	The commodity description (Solid Fuels).
Unit	The commodity unit throughout the model (PJ). It is responsibility of the user to be consistent with units.
LimType	The sense of the balance equation for the commodity. Valid entries are LO (Production>=Consumption, FX (Production=Consumption), UP (Production<=Consumption). When not specified, the default is LO.
CTSLvl	The commodity time-slice tracking level. Valid entries are ANNUAL, SEASON, WEEKLY and DAYNITE. When not specified, the default is ANNUAL.
PeakTS*	Peak time slice monitoring. Valid entries are: ANNUAL to generate the peaking equation for all time slices or any specific time slices already defined in the SysSettings file (comma-separated entries allowed). If not specified the default is ANNUAL.
CType	Electricity commodities indicator (ELC).

* Note: Comma separated elements are allowed.

2.3.5 Process definition tables ~FI_PROCESS

Process definition tables (~FI_Process) are used to declare the non-numerical characteristics of processes. The columns headers are fixed but their order can be changed. Each process needs to be declared (only) once in such a table as shown in Figure 4. They are supported in B-Y Templates and SubRES files.

~FI_Process

Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
*								
MIN		MINCOA1	Domestic Supply of Solid Fuels Step 1	PJ				
		MINCOA2	Domestic Supply of Solid Fuels Step 2	PJ				
		MINCOA3	Domestic Supply of Solid Fuels Step 3	PJ				
IMP		IMPCOA1	Import of Solid Fuels Step 1	PJ				
EXP		EXPCOA1	Export of Solid Fuels Step 1	PJ				

Figure 4. How to use ~FI_PROCESS (table from DemoS_001)

The valid column headers for a process table ~FI_PROCESS are described in Table 2.

Table 2. Valid column headers for a process table ~FI_Process

Header	Description
Sets*	The sets to which processes belong. The process set indicates the nature of a process. Valid entries are: ELE (thermal or other power plant), CHP (combined heat and power), PRE (generic process), DMD (demand device), IMP (import process), EXP (export process), MIN (mining process), HPL (heating plant), IPS for inter-period storage, NST for night storage device, STG for general timeslice storage, STS for simultaneous DayNite/Weekly/Seasonal, STK for simultaneous DayNite/Weekly/Seasonal and interperiod storage process. These declarations are inherited until the next one is encountered. In this example, there are three mining processes (MINCOA*), one import process (IMPCOA1) and one export process (EXPCOA1), all related to the supply of solid fuels (COA).
Region	The region name where the process exists (comma-separated entries allowed).

	By default, it is applied to all regions of the model when not specified. The region designation is used only in the B-Y templates and not allowed in SubRES.
TechName	The process name (e.g. MINCOA1), up to 32 characters. (However, it is recommended to limit process names to 27 characters as VEDA-FE may internally add digits for vintaging issues or dummy imports.)
ProcessDesc	The process description (e.g., Domestic supply of Solid Fuels Step 1), up to 255 characters.
Tact	The activity unit of the process (in Figure 4, for example, it is in PJ). It is the user's responsibility to be consistent with units.
Tcap	The capacity unit of the process. It is the user's responsibility to be consistent with units.
Tslvl	The process time-slice operational level. Valid entries are ANNUAL, SEASON, WEEKLY and DAYNITE. When not specified, the default is based on the Sets declaration: DAYNITE (for ELE, STGTSS, and STGIPS), SEASON (for CHP and HPL), ANNUAL (for all others).
PrimaryCG	The Primary Commodity Group (PCG) of the process. Normally none specified as VEDA allocates the PCG by default. A declaration is needed only when the user wants to create a new PCG and/or override the default PCG.
Vintage	Vintage tracking. Valid entries are YES or NO. When not specified, the default is NO.

* Note: Comma separated elements are allowed.

2.3.6 Flexible import tables ~FI_T

The Flexible Import Table **(~FI_T**) is used to create model topology (process inputs and outputs) in B-Y templates and SubRES. It also provides a very flexible structure (hence the name) for specifying numerical parameter values. With this type of identifier, the data is imported as provided and not modified during the import process.

Unlike in most other table types (with the exception of UC tables, described in Section 2.3.8), the ~FI_T table tag is not placed directly above the upper-leftmost table cell. Instead it is placed in the row immediately above the table headers and in the column before the first column containing values. This placement allows any number of columns to be designated for row identifiers, rather than data, as shown in Figure 5.

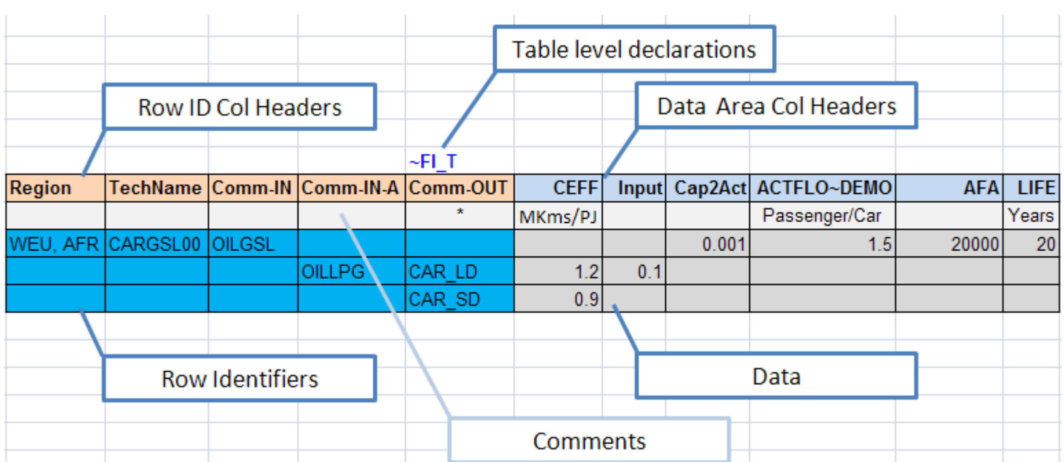


Figure 5. How to use the ~FI_T table

Indexes for the data, including attribute, region, year, and timeslice may be specified as either row identifiers or column headers, so that a table may be laid out to match the configuration of source data with minimal user intervention.

The ~FI_T table has six distinct regions. Valid entries in each of these are:

- Row ID Col Headers. The valid row ID column headers for a ~FI_T flexible import table are described in Table 3.

Table 3. Valid row ID column headers for a flexible import table ~FI_T

Header	Description
Region*	Region declaration
TechName	Technology Name
Comm-IN*	Input Commodity
Comm-IN-A*	Auxiliary Input Commodity
Comm-OUT*	Output Commodity
Comm-OUT-A*	Auxiliary Output Commodity
Attribute	Attribute declaration; single entries permitted
Year	Year declaration; comma-separated entries allowed
TimeSlice*	Time slices declaration; comma-separated entries allowed
LimType	Valid entries are: UP (Upper), LO (Lower), FX (Fixed) and N (Non-binding)
CommGrp	User Defined Commodity Group
Curr	Currency declaration
Stage	For multi-stage stochastic models
SOW	State of the World (Stochastic models)
Other_Indexes	To enter special dimensions that are required in certain attributes

* Note: Comma separated elements are allowed.

- Row Identifiers: elements of the dimension indicated in the row ID column headers.
- Data Area Column Headers: Elements of the following dimensions (elements of multiple dimensions can be separated by ~)
    - Attribute
    - Year
    - TimeSlice
    - LimType
    - Commodity
    - CommGrp (only the internal VEDA commodity groups: DEMO/DEMI/NRGO/NRGI/MATO/MATI/ENVO/ENVI/FINO/FINI can be used as column headers)
    - Region
    - Currency
- Data: numerical entries
- Table level declarations: Declarations like those made in column headers can be included in the table header (following a colon) and will apply to all data that doesn't have a different value for that index specified. For example, ~FI_T: DEMAND would assign

DEMAND as the attribute for all values in the table that don’t have an attribute specification at the column or row level.

- Comments: a comment row is identified by the character " * " or “\I:” as the first character in any of the cells below the Row ID Col Headers or the first character in any of the column headers. (However, caution should be exercised in using " * " to indicate a comment, because it may also be used to indicate a wildcard or an operation in some cells. “\I:” is the safer choice to indicate a comment row/column.)

2.3.7 Transformation Insert and Update tables ~TFM_INS and ~TFM_UPD

~TFM_INS is a transformation table used to insert new attributes and values in a rule-based manner. In this example from DemoS_001, it is used to declare three new attributes (G_DYEAR, Discount, and YRFR) by row as shown in Figure 6.

~TFM_INS					
TimeSlice	LimType	Attribute	AllRegions	REG1	Cset_CN
		G_DYEAR	2005		
		Discount	0.05		
ANNUAL		YRFR	1.00		

Figure 6. How to use ~TFM_INS table

~TFM_UPD is a transformation table used to update pre-existing data in a rule-based manner. For example, in Figure 7 it sets default prices (ACTCOST) for the backstop dummy processes for energy commodities (IMP*Z - dummy IMPort processes ending with “Z”) and demands (IMPDEMZ - a dummy IMPDEMZ process that can feed any demand).

~TFM_UPD																
TimeSlice	LimType	Attribute	Year	Other_Inde	AllRegions	REG1	REG2	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
		ACTCOST			2222			IRE	IMP*Z						
		ACTCOST			8888			IRE	IMPDEMZ						

Figure 7. How to use ~TFM_UPD table

Valid column headers for data entry in transformation insert and update tables (to update existing values or insert new values) are presented in the top portion of Table 4. These tables can identify the items whose data is to updated or inserted using the criteria in the bottom portion of Table 4.

Table 4. Valid column headers for transformation tables

Header	Description
	Insert or update values
Attribute	Name of the attribute; single entries permitted
Year	Year declaration; comma-separated entries allowed; default value = start year
TimeSlice	Time slices declaration; comma-separated entries allowed; default=ANNUAL.
LimType	Valid entries are: UP (Upper), LO (Lower), FX (Fixed) and N (Non-binding)
CommGrp	User Defined Commodity Group
Curr	Currency declaration; default=CUR.

Stage	For multi-stage stochastic models
SOW	State of the World (Stochastic models)
Other_Indexes	To enter special dimensions that are required in certain attributes
AllRegions	Data value that is applicable to all regions
<Regions>	Region-specific data values; these will supersede any declaration in AllRegions column
Commodity and process filtering	
PSet_Set[1]	To identify processes based on TIMES set membership
PSet_PN[2]	To identify processes based on names
PSet_PD[2]	To identify processes based on descriptions
PSet_CI[2]	To identify processes based on commodity inputs
PSet_CO[2]	To identify processes based on commodity outputs
CSet_Set[1]	To identify commodities based on TIMES set membership
CSet_CN[2]	To identify commodities based on names
CSet_CD[2]	To identify commodities based on descriptions
Top_Check	To restrict application of attribute data to those process-commodity combinations where the specified topology already exists in the model, rather than creating new topology. Valid entries: **I/O/A.** **“I”** will retain those combinations where commodities are input to processes. **“O”** => Output; **“A”**=> Input or output. No topology check is performed by default.
Attrib_Cond	To filter based upon whether an attribute is present or missing (precede with “-“) for specified processes.
Val_Cond	Used in conjunction with Attrib_Cond to filter on the value of the specified attribute. Define using '<', '>', ‘<>’, or '='. The condition will be tested across all dimensions (for example, years) for the specified process, region, and attribute.

[1] Comma separated elements are allowed. Each of these fields can have comma-separated entries that are joined by OR.

[2] Comma separated elements and wild cards characters are allowed. The possible wild cards are:
“*” is used as wild card; for example *GAS* would refer to all elements that have GAS in the name with any possible characters before and after GAS.
“-“ before the text used for exclusions; for example, *GAS*,-ELCGAS would refer to all elements that have GAS in the name except for ELCGAS.
“?“ can be used to specify a single character; for example, ???GAS means there are 3 characters before GAS.

2.3.8 User constraints and their tables

User constraints provide the modeller with a flexible framework to add case-study specific constraints to the standard equation set embedded in TIMES. With the help of user constraints, virtually any possible linear relationship between core variables in TIMES can be formulated, and some input attributes can also be brought in as coefficients. User constraints can also be written to link variables across consecutive time slices or periods. Section 6.4 of Part II of the TIMES documentation contains an extensive discussion of the user constraint types available and their mathematics.

Defining user constraints in VEDA-FE templates is a two step process. They are first declared with one or more **~UC_SETS:** tags, which indicate their type and domain of coverage. Then their data is specified using a table with similar structure to that of a **~FI_T** table, as shown in Figure 8.

~UC_Sets: R_E: AllRegions													
~UC_Sets: T_E:													
								~UC_T:UC_RHSRTS					
UC_N	Pset_Set	Pset_PN	Pset_CI	Pset_CO	Cset_CN	Attribute	Year	LimType	UC_COMNET	REG1	REG2	UC_RHSRTS~0	UC_Desc
AU_CO2_BND					TRACO2,ELCCO2		2010	UP	1	1142284	1142284	5	CO2 Bound Constraint
					TRACO2,ELCCO2		2020	UP	1	1227958	1053944		

Figure 8. Defining a user constraint in VEDA-FE

Available UC sets are described in Table 5. Each set definition holds for the entire sheet, unless redefined. All the existing set definitions are applied to all user constraints in a table.

Table 5. UC sets available in VEDA-FE

~UC_SETS:	Signification	Application
R_E	Region_Each	REG1: apply to one particular region
R_S	Region_Sum	REG1,REG2: apply to more than one region (comma separated) AllRegions: will apply to all regions
T_E	Time period_Each	
T_S	Time period_Sum	
TS_E	Time slice_Each	
TS_S	Time slice_Sum	
T_SUC	Time period successive	

A UC table is then structured similarly to a Flexible Import table, with the **~UC_T** tag separating the column headings into row identifiers (UC_INDEXES) and data column headers. Valid row ID (UC_INDEX) column headers are:

Table 6. UC_INDEXES for user constraint tables

UC_INDEXES Column Header	Description
UC_N	Short Name of the UC
Region	Name of the region(s)
PSet_PN*	Comma separated list of process names
PSet_PD*	Comma separated list of process description
Pset_CI*	Comma separated list of (input) commodities to define a set of processes
Pset_CO*	Comma separated list of (output) commodities to define a set of processes
Cset_CN*	Comma separated list of commodity names
Cset_CD*	Comma separated list of commodity description
Side	LHS/RHS; RHS is applicable only in the case of dynamic (across periods) constraints.
Attribute	Any of the UC attributes available in the current TIMES code
UC_ATTR	• Allows modifiers to be applied to the variables used in the UC. These include the GROWTH modifier, to create a constraint that limits the percentage growth in a variable over periods; modifiers to pull input data, such as COST and EFF, into the UC's coefficients;

	and the NEWFLO modifier that applies the UC coefficient to the flows of the new vintage of a process only. More details are found in Section 6.4.6 of Part II. • The contents of this column are comma separated values of UC_Name and UC_GrpType. Several pairs can be separated by ";" • A pair can have UC_Name/GrpType in any order; any element in the list ACT, CAP, NCAP, FLO, IRE, COMCON, COMPRD, COMNET is taken as GrpType and the other one is designated as the UC_Name. Valid UC_Names are provided and described in Section 6.4.6 of Part II. • UC_ATTR can have a ~ appended to it; the default is LHS.
Year	Comma separated list of years is allowed
LimType	UP/LO/FX/N
Top_Check	To control the process-commodity combinations via topology when both indexes exist for the attribute in question. Valid entries: **I/O/A.** **"I"** will retain those combinations where commodities are input to processes. **"O"** => Output; **"A"**=> Input or output. Default = **A.**

* Wild cards allowed

Valid data column headers are:

- Any of the UC attributes available in the current TIMES code
- Years (including 0 for interpolation setting)
- Region
- UP/LO
- LHS/RHS

Multiple values can be separated by "~". Any specification without a region identifier in the column is applied to the region in the row identifier area. If there is no region, it applies to all regions in the active R_E/R_S specification.

A user constraint definition can span multiple rows of the table (to attach numbers/attributes and other indexes to different sets of processes/commodities).

2.4 VEDA-FE database tools

Once the templates have been imported and assembled as a model database within VEDA-FE, it is possible to review the resulting data by means of powerful filtering tools and dynamic data cubes (pivot tables), and it is also possible to view the RES by requesting that the network diagram be displayed.

2.4.1 Browser

The database browser can be accessed from the main menu (**Basic Functions, Browse/Edit, TIMES View** or **VEDA View)** or by pressing **F7**. The **TIMES View** shows the data with TIMES attribute names, while the **VEDA View** shows the same information, but with the attribute names as used in the files[1]. The browser allows the user to view subsets of the assembled data in a cube by selecting the scenario(s), region(s), process(es), commodity(ies),

[1] A variety of alternate attribute names, or *aliases*, are available for many TIMES parameters, as shown in the Attribute Master (see Section 2.4.4).

and/or the attribute(s) of interest: the new nuclear power plants in REG1 in the example shown in Figure 9.

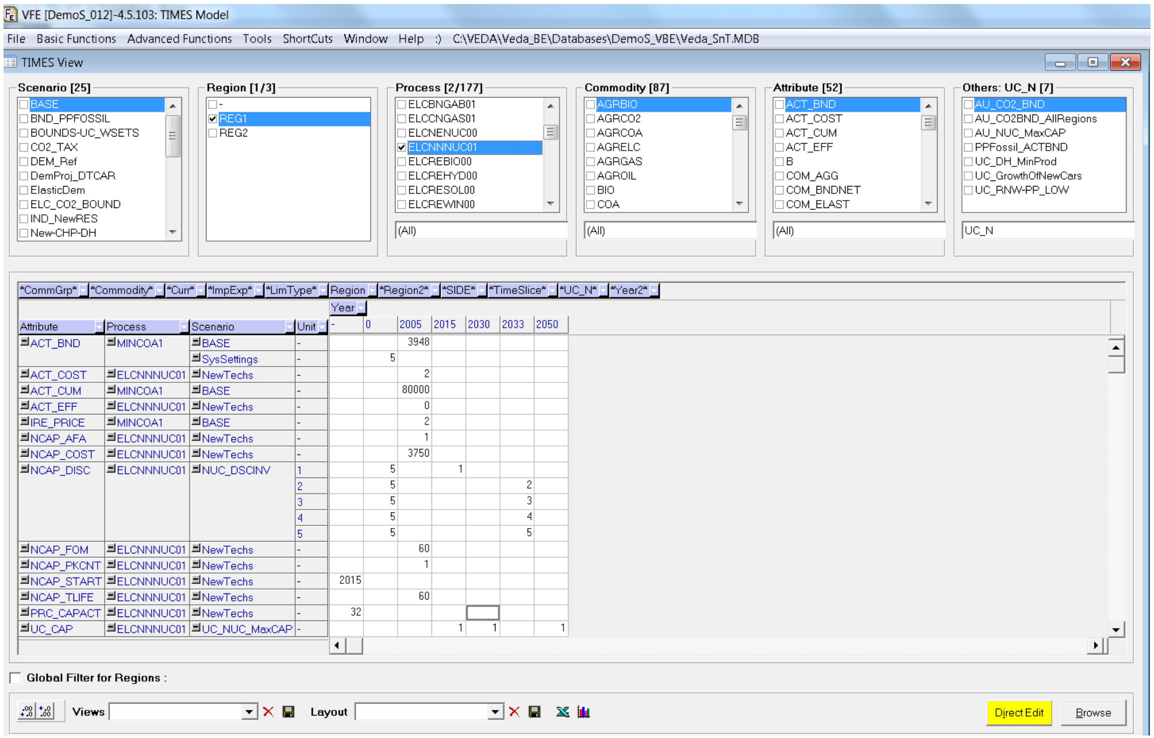


Figure 9. Browser for the model database

First, elements are selected manually or via search tools (right-click) in the different dimensions, and then the information is displayed in a default layout by clicking the **Browse** button. It is possible to rearrange the layout of the cube by adding/removing dimensions (columns and rows) to/from the table by dragging/dropping components from/to the area above the current row designator columns.

When the cursor is placed over a dimension a crosshair appears. Then, holding the left mouse button down and sliding to a new position, a green line will appear indicating that the dimension may now be dropped there. Any dimension not positioned as part of the row/column table layout definition appears at the top of the page. These dimensions have their values summed in the cube. For each dimension on top of the page, if more than one value exists for that dimension, its name will be displayed between two asterisks reinforcing that some values in the cube may be aggregates. Note that for any dimension where only a single value exists, said dimension is automatically moved up top. Using the pull-down arrow associated with each header, individual entries may temporarily be removed by unselecting them from the list of elements.

2.4.2 RES viewer

The RES viewer can be accessed from the main menu (Basic Functions, RES). It is possible to navigate around the model by clicking on the name of a commodity or process, allowing the user to see: 1) in the case of a commodity, all processes producing and consuming that commodity; and/or 2) in the case of a process, all input and output flows. The right panel of Figure 10 shows the RES viewer as zoomed in on commodity ELCNUC. We see the single process that produces it and the two that consume it. The legend in the left panel defines the color coding of the commodities and processes shown. By clicking on any item, the user can cascade through the RES to better visualize the interrelationships and competing processes throughout the network.

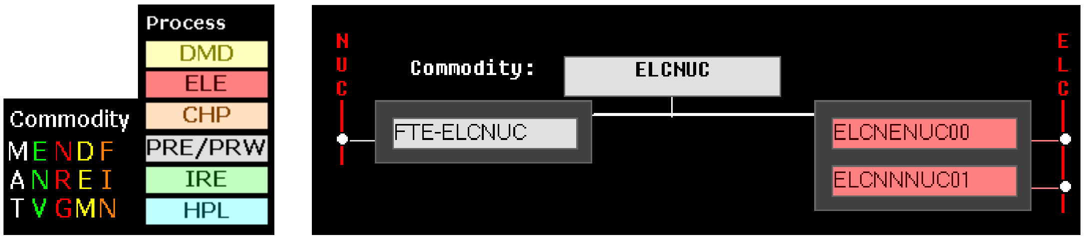


Figure 10. Schematic representation of the RES

2.4.3 Commodity/Process Master

The **Commodity/Process Master** viewer can be accessed from the main menu (**Advanced Functions**). This feature can be used to examine a process or commodity's declaration, connectivity, and data details. As in the RES viewer, it is possible to navigate around the model viewing all the commodities/processes immediately before/after the focus item in the RES. In addition, the **Information** and **Data** tabs provide all of the declaration and data information about the focus commodity/process (see Figure 11 and Figure 12). It is also possible to get a full list of commodities/processes in the model by clicking the label **All Commodities** (**Processes**).

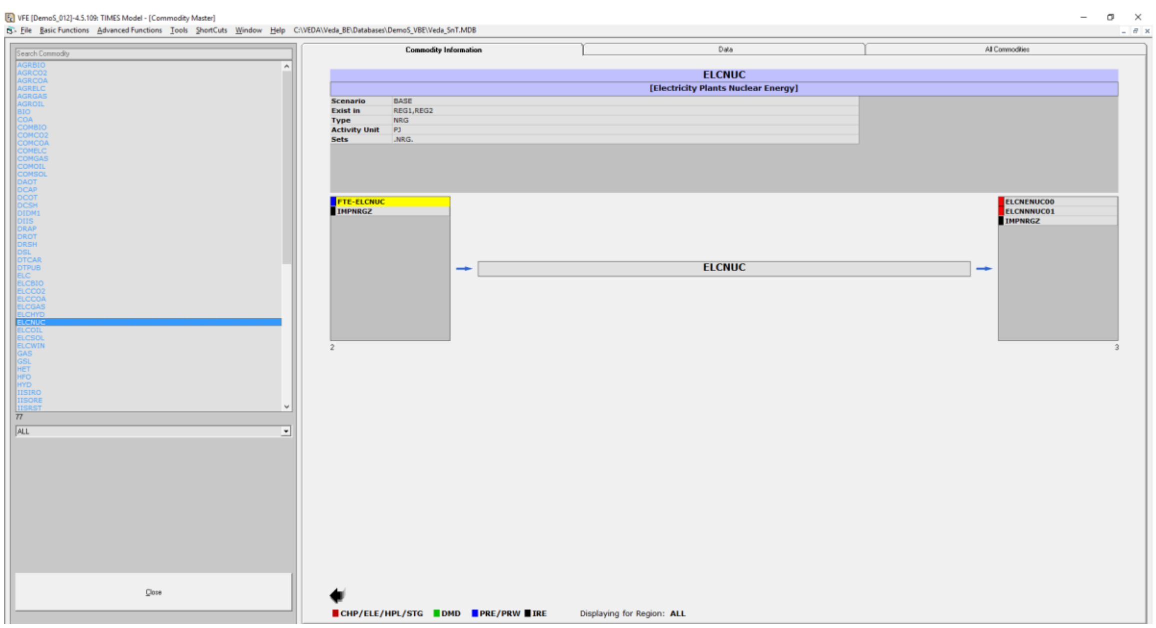


Figure 11. Commodity Master view

2.4.4 Attribute Master Table

The Attribute Master table is available under **Advanced Functions/Attribute Master** (as shown in Figure 13).

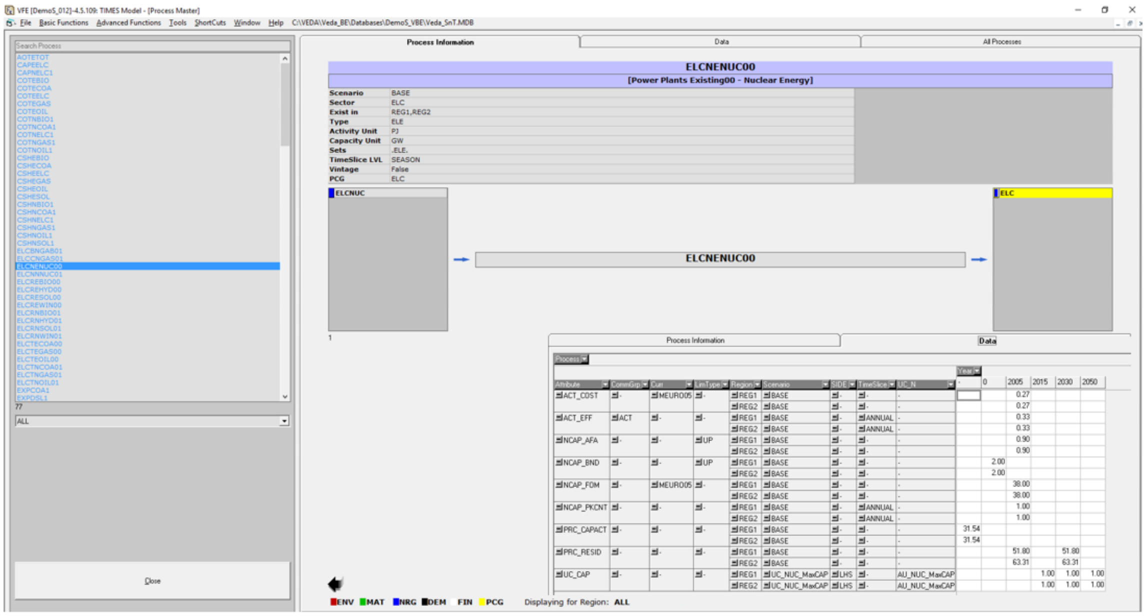


Figure 12. Process Master view and data view

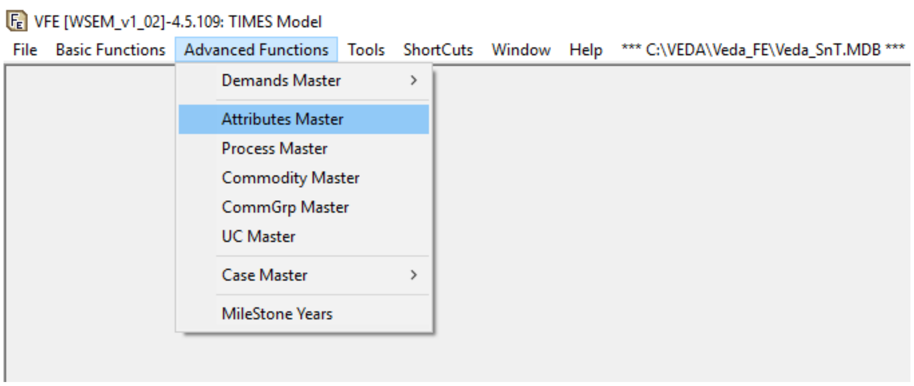


Figure 13. How to find the Attribute Master Table

The Attribute Master table (Figure 14) shows all the TIMES attributes/parameters supported by VEDA-FE and can assist in creating the FI_T, TFM_INS and UPD tables. The table uses the following color code:

- Grey cells indicate a VEDA-FE default applied to an attribute.
- Green or red cells are used to specify whether an attribute is interpolated/extrapolated by default or not (a user rule can always be defined).
- Light blue cells indicate that an attribute index is required.

The table consists of the following columns.

- Attribute: lists the name of each supported attribute that can be used in VEDA-FE tables.

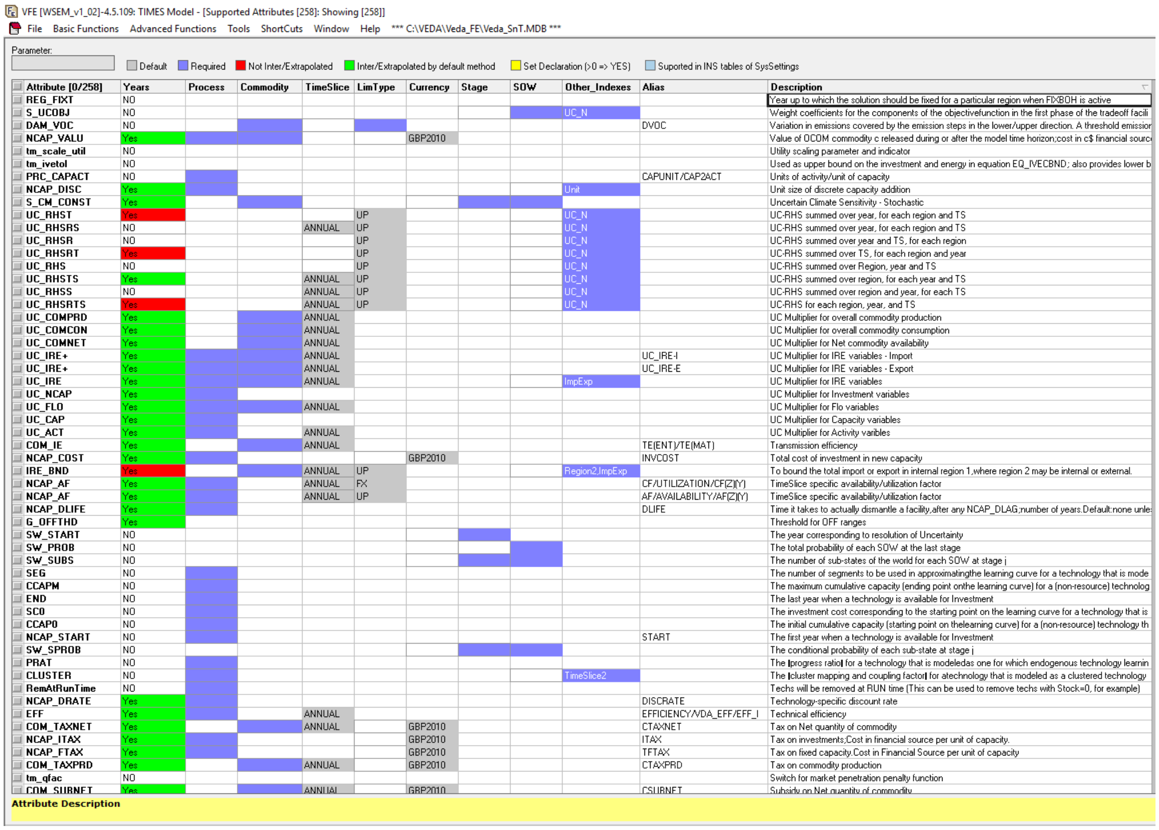


Figure 14. Attribute Master table

- Years: specifies whether the attribute can be defined by year (YES) or not (NO). For attributes that can be, the color of the cell indicates the default interpolation/extrapolation (I/E) behavior. If green, there is a default I/E applied. If red, there is not.
- Commodity/Process: specifies whether an attribute may be defined by commodity, process, or both.
- Timeslice column: specifies whether attribute can be defined by timeslice and lists any default applied by VEDA-FE.
- Limtype: specifies whether a limit type is required (light blue) and lists any default applied by VEDA-FE.
- Currency: shows the default currency that will be inserted if none is specified.
- Stage and SOW: these columns indicate where this information is required for attributes used in the stochastic version of TIMES.
- Other_Indexes: list additional information required by an attribute.
- Alias: lists alternative names for the TIMES attribute that can be used in templates.
- Description: provides a description of the attribute.

2.4.5 Case Manager

The Case Manager, available from the **Basic Functions** menu, or by pressing **F9**, is used to compose and submit a model run. Figure 15 illustrates the main features of the Case Manager, as follows:

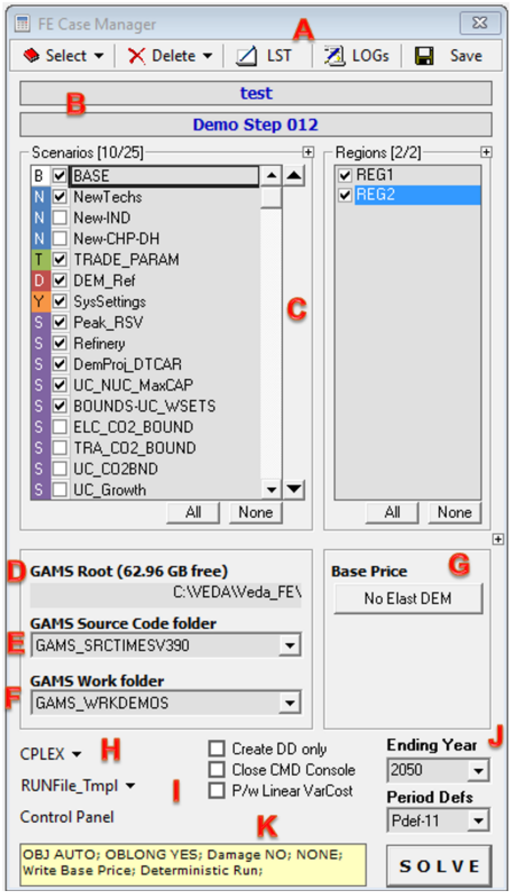


Figure 15. Case Manager

A. **Select**, **Delete**, **LST**, **LOGs** and **Save** buttons. **Select** can be used to choose a previously saved run configuration (run name, run description, scenarios, regions, ….), while **Delete** is used to delete an existing configuration.
LST opens the list file associated with a completed run, with information about the run and equations if enabled.
LOGs opens a QA check log file with information and any warnings about a completed run. (See Part III of the TIMES documentation for more on the LST and QA check files.)
Save is used to save a new configuration or update an existing one.

B. The first row is used to specify a scenario name (spaces in the name are not allowed), while the second row can be used for scenario description.

C. The **Scenarios** and **Regions** lists are used to check and uncheck which scenarios and regions are included in a run configuration. In Figure 15, for example, the scenario New-IND is unchecked, so its data will not be included in the model run. The colored letters in the scenario panel indicate the scenario type. (E.g., white is the BASE scenario from the VT files, light blue are the SubRES, and purple are the Scenario files).

D. Indicates the VEDA installation folder. Clicking on the path here will open a browse to select the installation folder.

E. Indicates the TIMES source code (model generator) installation folder. A left click on **GAMS Source** will open the folder.

F. Indicates the folder in which the results of this run will be saved. It is possible to create a new save folder by selecting **New Folder** from the dropdown menu here. A left click on **GAMS Work folder** will open the folder.

G. This button is used to choose the reference prices in the TIMES elastic demand variant.

H. The **Cplex** button opens a window used to set up options for the CPLEX solver. The dropdown menu allows selection of an alternative solver on your computer.
The RUNfile_Tmpl button opens the run file used by TIMES when the model is submitted to be solved. The dropdown allows selection from alternative versions of the run file.
Control Panel provides access to a selection of control panel switches (described in Part III of the TIMES documentation) and TIMES variants (described in Parts I and II).

I. **Create DD** is used to create a data dictionary (DD) file without solving the model.
Close DMD Console if checked will automatically close the "DOS" window at the end of the run.
P/w Linear VarCost used to reduce run size by projecting linear variable costs.

J. **Ending Year** allows selection of the ending year of a run.
Period Defs allows selection among period definitions (saved in the SysSetting file).

K. Summary of run option selections.

More information on VEDA-FE and description of additional features are available at http://support.kanors-emr.org/. Each of the remaining sections of this document describes one incremental step of the VEDA-TIMES Demo Models.

3 TIMES demo models

This section explains how to progress in the use of TIMES features and variants using the set of VEDA-TIMES Demo Models. This is a set of VEDA-TIMES models that start from an energy balance and focus on building a model incrementally employing a standard approach to describe the underlying Reference Energy System (RES) as well as specific naming conventions.

The first step model starts with a simple supply curve feeding a single demand. The Demos then grow step by step to build out the RES, adding new commodities, processes (or technologies) and regions, while introducing new attributes (or parameters) and more advanced TIMES modelling features, and explaining the *why* of the different choices made in VEDA-FE for building these models.

The VEDA-TIMES Demo Models consist of several incremental steps. Steps 1 to 12 are considered the Basic Demo models (Table 7), and are described in this section.). For each step, it provides:

- A brief description of the step model and the objectives in terms of VEDA-TIMES features demonstrated;
- A summary of attributes introduced and files created, modified, and/or replaced;
- A step-by-step description of the template tables created and/or modified in each file; and
- A brief look at the results.

Table 7. The Basic Demo models

Demo	Folder name	Short description
001	DemoS_001	Resource supply
002	DemoS_002	More demand options and multiple supply curves
003	DemoS_003	Power sector: basics
004	DemoS_004	Power sector: sophistication
005	DemoS_005	2-region model with endogenous trade: compact approach
006	DemoS_006	Multi-region with separate regional templates
007	DemoS_007	Adding complexity
008	DemoS_008	Split Base-Year (B-Y) templates by sector: demands by sector
009	DemoS_009	SubRES sophistication (CHP, district heating) and Trans files
010	DemoS_010	Demand projections and elastic demand
011	DemoS_011	Linking input templates and VEDA-BE sets
012	DemoS_012	More modelling techniques

3.1 DemoS_001 - Resource supply

Description. This is the first step and therefore represents a very simple model that serves as the starting point for the development of a more complex model: it includes a single supply curve and a single demand for one commodity in a single region over two time periods.

Objective. The objective is to introduce examples of how to implement in VEDA-FE templates the most basic types of energy commodities and processes that are normally part of a typical TIMES model, along with their respective attributes: a three-step supply curve, an import and an export option, one generic demand and one demand process for one energy commodity (i.e. coal).

This first demo is used also to introduce the SysSettings workbook, the base year template (or VT template), and how to use the most common VEDA-FE tables.

Attributes introduced[2]:		Files created
G_DYEAR	EFF	SysSettings
Discount	AFA	VT_REG_PRI_v01
YRFR	INVCOST	
CUM	FIXOM	
COST	LIFE	
ACT_BND	DEMAND	

The first step model is built using only two files: the default SysSettings file and one B-Y Template (VT_REG_PRI_V01). The base year transformation file (BY_Trans) is created by default; it is empty at this stage. Figure 16 shows the VEDA-FE Navigator (see Section 2.2) for the DemoS_001. This is the first window you will see when you switch from another model to the DemoS_001.

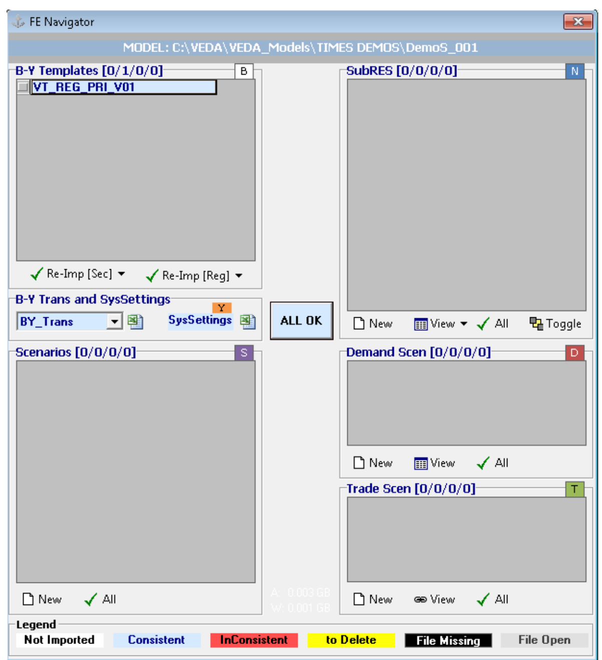


Figure 16. The files included in DemoS_001

The reference energy system (RES) of this first demo can be viewed in VEDA-FE (see Section 2.4.2 for more information), and it is shown in Figure 17. The RES shows an energy service demand called TPSCOA satisfied by an end-use demand device called DTPSCOA, which uses as its input the commodity called COA. The COA commodity can be also exogenously exported outside the model boundary with the export technology called EXPCOA1.

[2] The meaning of all the attributes, along with their qualifier indexes, as said above can be found in VEDA-FE, Advanced Functions menu, Attributes Master (Figure 14).

The production of the COA commodity is based on one import technology (IMPCOA1) and on a three step local supply curve with the technologies MINCOA1, MINCOA2 and MINCOA3.

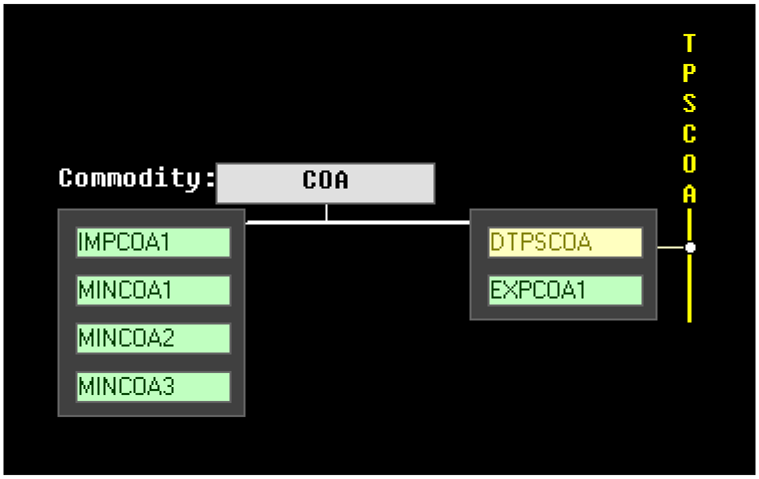


Figure 17. Reference energy system DemoS_001

The next two sections explain how this TIMES model for delivering the commodity TPSCOA at the minimum cost is built in VEDA-FE sheet by sheet for the two templates of this first demo.

3.1.1 SysSetting template

This file is used to declare the very basic structure of any VEDA-TIMES model, including its regions, time slices, start year, etc. It also contains some settings for the synchronization process and can include some additional information. In this example, this file contains the following sheets:

- Region-Time slices
- TimePeriods
- Import Settings
- Interpol_Extrapol_Defaults
- Constants
- Defaults
- Commodity Group. (This sheet is not used in the basic Demos. In general it can be used to build user commodity groups.)

3.1.1.1 Region-Time slices

This sheet contains two tables (Figure 18):

- ~BookRegions_Map is used to define
  1. The workbook name (here, REG), which needs to be the same for each B-Y Template of a region.
  2. The list of model region names (REG1). In this first step, there is only one region and one file.

- ~TimeSlices is used to define the time-slice resolution for the model at different hierarchical levels: SEASON, WEEKLY and DAYNITE. In this first step, there is only one time slice defined by the user for the seasonal level and called ANNUAL.

~BookRegions_Map	
BookName	Region
REG	REG1

~TimeSlices		
Season	Weekly	DayNite
ANNUAL		

Figure 18. Regions and time slices definition in the SysSettings file for DemoS_001

3.1.1.2 <u>TimePeriods sheet</u>

This sheet contains three tables (Figure 19):

- ~StartYear is used to define the start year of the model (2005 for this example and all the other steps).
- ~ActivePDef is used to select the set of active periods (Pdef-1, by default) from all those defined in the following table.
- ~TimePeriods is used to specify period definitions by specifying the number of years for each period. In this step, only a single period definition has been created (Pdef-1), which contains 1 year for the first period (start year) and 2 years for the second period.

~StartYear
2005
~ActivePDef
Pdef-1
~TimePeriods
Pdef-1
1
2

Figure 19. Start year and time period definition in the SysSettings file

3.1.1.3 <u>Import Settings sheet</u>

This sheet contains one table (Figure 20):

- ~ImpSettings is used to declare some settings for the synchronization process (1= active; 0=non active).
    - Check #DIV/0 and #REF errors in Templates – VEDA will produce a log after the import process identifying errors made by the user.
    - Dummy Imports – These backstop dummy processes can be introduced in the model automatically by VEDA in order to avoid infeasibilities that may arise

when there is not enough energy carrier produced or when a demand cannot be supplied. These aid in the model debugging process.

- o Vintage Bounds - VEDA can automatically generate bounds to prohibit investments in processes when a newer vintage becomes available, assuming that the last two characters of the process name are used for the first year of availability.
- o DumVarforUC – This dummy variable option can be introduced in the model automatically by VEDA for each User Constraint in order to avoid infeasibilities due to a user constraint. These aid in the model debugging process.

~ImpSettings

Option	Value
Check #DIV/0 and #REF errors in Templates	1
Create Dummy Imports for Energy and Material Commodities	1
Create Dummy Imports for Demands	1
Generate Vintage Bounds	0
DumVarforUC	0

Figure 20. Import settings in the SysSettings file

3.1.1.4 <u>Interpol_Extrapol_Defaults sheet</u>

This sheet normally contains two tables, one for setting user interpolation rules applied to all the other files unless the user specifies new rules in other templates to overwrite this information, and one for setting the default prices of dummy import processes. There is only the second table in the current version (Figure 21).

- ~TFM_UPD (transformation update table) is a transformation table used to update pre-existing data in a rule-based manner. In this example, it sets default prices (ACTCOST) for the backstop dummy processes for energy commodities (processes with names matching IMP*Z – dummy IMPort processes ending with "Z") and demands (IMPDEMZ - a dummy IMPDEMZ process that can feed any demand). These costs should be a few orders of magnitude higher than real import costs in your model in order to ensure that these processes only become active when real fuel supplies are insufficient or unavailable.
- More information about this type of table is available in Section 2.3.7.

~TFM_UPD

TimeSlice	LimType	Attribute	Year	Other_Inde	AllRegions	REG1	REG2	Pset_Set	Pset_PN	Pset_PD
		ACTCOST			2222			IRE	IMP*Z	
		ACTCOST			8888			IRE	IMPDEMZ	

Figure 21. Dummy import prices in the SysSettings file

3.1.1.5 <u>Constants sheet</u>

This sheet contains one table (Figure 22):

- ~TFM_INS is a transformation table used to insert new attributes and values in a rule-based manner. In this first step, it is used to declare three new TIMES attributes.

- G_DYEAR - discounting year; this is a user input and in this example is 2005.
- DISCOUNT - overall discount rate for the energy system; this is a user input and in this example is 5% and is constant for the entire modelling horizon. The same rate is used for depreciation of investments.
- YRFR - fraction of year for each time slice; this is a user input and in this example is 100% for the single ANNUAL time slice.

- More information about this type of table is available in Section 2.3.7.

~TFM_INS					
TimeSlice	LimType	Attribute	AllRegions	REG1	Cset_CN
		G_DYEAR	2005		
		Discount	0.05		
ANNUAL		YRFR	1.00		

Figure 22. Constant declarations in the SysSettings file

3.1.1.6 Defaults sheet

This sheet contains two tables (Figure 23):

- ~Currencies is used to define a default currency for the whole model; this is a user input. In this example the default unit is million 2005 euros (MEuro05). It is important to note that for TIMES this is just a label called MEuro05. It is the user's responsibility to be consistent with costs and units in the model.
- ~DefUnits is used to define units for activity, capacity and commodity for each sector in the model: petajoules (PJ) and petajoules per year (Pja) in this case. Again, it is the user's responsibility to ensure consistency in the units used in any TIMES model. It is possible to use any units, but it is important to be coherent across the model.

~Currencies
Currency
MEuro05

~DefUnits	
Option	PRI
Process_ActUnit	PJ
Process_CapUnit	Pja
Commodity_Unit	PJ

Figure 23. Default declarations in the SysSettings file

3.1.2 B-Y Template

The B-Y templates are used to set up the base structure of the model, and in principle it is possible to build a full model using just B-Y templates. This is the approach used for this first example. Later when the model grows to include more commodities, technologies, sectors, regions, and additional information to run different scenarios, we will demonstrate the flexibility and modularity of VEDA-FE using different types of workbooks to input information.

In this first example the B-Y Template is used to set up the base-year process stock and the base-year end-use demand levels, such that the overall energy flows reflect the energy balance.

3.1.2.1 RES&OBJ sheet

This sheet contains some pictures showing the normal completion of a run in VEDA-FE with the value of the objective function and the same objective function in the VEDA-BE table. It is also showing the reference energy system of this model step using the VEDA-FE feature Go-To RES described in Section 2.4.2.

3.1.2.2 EnergyBalance sheet

This sheet contains the energy balance at the model start year (2005) for REG1 (Figure 24). The energy balance in itself is not imported into the model; the table is not identified with any VEDA table header (cell starting with the character "~"). However, it allows the user to calibrate the model start year with appropriate historical energy flows. A typical energy balance comprises two dimensions:

- Different types of energy commodities in columns. In this simple example, the different types of energies are partially aggregated in categories (e.g. solid fuels, renewable energies, etc.). The first row of the table includes codes defined by the modeller that are used to name the energy commodities in the model.
- Steps of the whole supply-demand chain in rows. This simple example shows three main sections: primary energy supply, energy conversion and final energy consumption. For each energy commodity, the primary energy supply minus the energy used for conversion yield the remainder for final energy consumption. The first column of the table includes codes defined by the modeller that are used to name energy processes in a uniform manner in the model.

		COA	GAS	OIL	NUC	RNW	SLU	HET	ELC	
		Solid Fuels	Natural Gas	Crude Oil	Nuclear Energy	Renewable Energies	Industrial Wastes	Derived Heat	Electricity	Total
	PRIMARY									
MIN	Domestic Supply	8098	7899	5379	10775	5027	0	0	0	**37178**
IMP	Imports	6463	13292	39960	0	113	0	0	1168	**60995**
EXP	Exports	-1147	-2516	-14831	0	-72	0	0	-1127	**-19693**
TPS	**Total Primary Supply**	13414	**18675**	**30508**	**10775**	**5067**	**0**	**0**	**41**	**78480**
	CONVERSION									
ESC	Energy Sector Consumption	-58	-793	-1849	0	-4	-2	0	0	**-2705**
ELC	Electricity Plants	-9598	-5636	-1225	-10775	-1256	-33	1738	11581	**-15204**
HPL	Heat Plants	-161	-301	-50	0	-140	-2	659	0	**5**
REF	Petroleum Refineries			-31736						**-31736**
	Total Conversion	**-9817**	**-6730**	**-34859**	**-10775**	**-1400**	**-36**	**2396**	**11581**	**-49640**
	FINAL									
RSD	Residential	357	5160	2289	0	1294	0	865	2872	**12837**
COM	Commercial	57	1752	855	0	67	1	255	2527	**5514**
IND	Industry	1897	4437	2016	0	722	117	634	4088	**13911**
AGR	Agriculture	44	201	797	0	63	0	16	19	**1141**
TRA	Transport	1	21	14851	0	131	0	0	266	**15270**
OTH	Other	1189	0	393	0	1390	0	627	650	**4249**
NEN	Non Energy	52	634	4073		0	0	0	0	**4759**
BNK	Bunkers	0	0	2111		0	0	0	0	**2111**
TFC	**Total Final Consumption**	**3597**	**12205**	**27385**	**0**	**3667**	**118**	**2396**	**10423**	**59791**

Data used in the template to buld the model

	COA
Domestic Supply Curve Share - Step 1	75%
Domestic Supply Curve Share - Step 2	25%

Figure 24. Initial energy balance at start year 2005 for REG1 – Covered in DemoS_001

The portion of the energy balance that is developed in this first step model is identified using a different color (orange): primary supply of solid fuels (COA).

A greater level of disaggregation can be added along both commodity and sector dimensions using additional user assumptions as data sources. In this example, shares are provided below the energy balance table to split the total domestic production of solid fuels (COA) into more than one step. This way, it is possible to set up in the model a supply curve defined by the maximum production and cost of each step.

3.1.2.3 Pri_COA

This sheet shows how to declare commodities and processes (in their respective declaration tables) and to describe specific supply processes (in a flexible import table): primary supply of solid fuels (COA) in this example.

In any TIMES model, all commodities and processes in the model need to be declared once in commodity tables (identified with ~FI_Comm) and process tables (identified with ~FI_Process) with a structure as explained in Sections 2.3.4 and 2.3.5 and shown in Figure 25 and Figure 26.

~FI_Comm								
Csets	**Region**	**CommName**	**CommDesc**	**Unit**	**LimType**	**CTSLvl**	**PeakTS**	**Ctype**
*Commodity Set Membership	Region Name	Commodity Name	Commodity Description	Unit	Sense of the Balance EQN.	Timeslice Level	Peak Monitoring	Electricity Indicator
NRG		COA	Solid Fuels	PJ				

Figure 25. A typical commodity declaration table

~FI_Process								
Sets	**Region**	**TechName**	**TechDesc**	**Tact**	**Tcap**	**Tslvl**	**PrimaryCG**	**Vintage**
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
*								
MIN		MINCOA1	Domestic Supply of Solid Fuels Step 1	PJ				
		MINCOA2	Domestic Supply of Solid Fuels Step 2	PJ				
		MINCOA3	Domestic Supply of Solid Fuels Step 3	PJ				
IMP		IMPCOA1	Import of Solid Fuels Step 1	PJ				
EXP		EXPCOA1	Export of Solid Fuels Step 1	PJ				

Figure 26. A typical process declaration table

Unlike the tables used to declare commodities and processes, the tables used to describe specific processes are very flexible (~FI_T). They are built using first **Row ID column headers** before and below the ~FI_T tag to identify the process names (TechName), descriptions (TechDesc), commodity inputs (Comm-IN), and commodity outputs (Comm-OUT), as well as the years of data (Year) when relevant. Then **Data column headers** after the ~FI_T are used to describe these processes. The number and arrangement of rows and columns is totally flexible in these tables. More information about the ~FI_T tables is available in Section2.3.6.

In the first model step, a flexible import table is used to describe the primary supply options for COA (Figure 27):

- A 3-step domestic coal supply curve through three mining processes (MINCOA*), each characterized with the cumulative amount of resources available over the modelling horizon (CUM), the annual cost per unit of energy (COST) and a bound on the annual production (ACT_BND) for the start year 2005 and the following period 2006. Bounds need to be combined with the LimType (UP), which is indicated in a specific column in this example. When not specified, it is UP by default (see Attribute Master Table, Section 2.4.4).

- Import and export options are characterized with the COST and ACT_BND attributes.

				~FI_T			
TechName	Comm-IN	Comm-OUT	Year	LimType	CUM	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity			Reserves Cumulative Value	Cost	Annual Production Bound
*Units					PJ	M€2005/PJ	PJ
MINCOA1		COA	2005	UP	80000	2.00	6074
			2006	UP			6074
MINCOA2		COA	2005	UP	160000	2.50	2025
			2006	UP			2025
MINCOA3		COA			320000	3.00	
IMPCOA1		COA				2.75	
EXPCOA1	COA		2005	UP		2.75	1147
			2006	UP			1147

*Blue cells are linked to the energy balance.

Figure 27. Description of supply options in a flexible table

3.1.2.4 <u>DemTechs_TPS</u>

This sheet shows how to declare commodities and processes (in their respective tables) and to describe specific demand processes (in a flexible import table): a demand process to deliver the total primary supply coal demand, in this example.

A new DEM commodity (TPSCOA -Demand Total Primary Supply – COA) and a new DMD process (DTPSCOA – Demand technology Total Primary Supply – COA) are declared in the commodity and process tables (Figure 28), as described in the previous section.

~FI_Comm								
Csets	Region	CommName	CommDesc	Unit	LimType	CTSLvl	PeakTS	Ctype
*Commodity Set Membership	Region Name	Commodity Name	Commodity Description	Unit	Sense of the Balance EQN.	Timeslice Level	Peak Monitoring	Electricity Indicator
DEM		TPSCOA	Demand Total Primary Supply - COA	PJ				

~FI_Process								
Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
*								
DMD		DTPSCOA	Demand Technology Total Primary Supply - COA	PJ	PJa			

Figure 28. Declaration of demand commodities and processes

A flexible import table is used to describe the demand option for total solid fuels (Figure 29):

- A demand process for the total primary supply of COA (DTPSCOA) is characterized with an efficiency (EFF), an annual availability factor (AFA), an investment cost (INVCOST), a fixed operation and maintenance cost (FIXOM), and a technical lifetime (LIFE). By default this technical lifetime is also used as the economic lifetime, unless a specific economic lifetime (ELIFE) is defined.

		~FI_T						
TechName	Comm-IN	Comm-OUT	STOCK	EFF	AFA	INVCOST	FIXOM	LIFE
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Lifetime
*Units			PJa			M€2005/PJ	M€2005/PJa	Years
DTPSCOA	COA	TPSCOA		1.00	0.95	10	0.20	20

Figure 29. Description of a simple demand processes

3.1.2.5 Demands

This sheet is used to specify the demand (DEMAND) value for the TPSCOA at the base year 2005 (Figure 30). This value comes from the energy balance and represents the total final COA consumption and the total consumed for energy conversion. This demand is constant over the time horizon of the analysis due to the default interpolation/extrapolation applied to the attribute Demand. The future values can be changed by specifying new inputs for the future years/periods.

	~FI_T		
Attribute	CommName	*Unit	2005
*	Demand Commodity Name	Demand Unit	Demand Value
*Units			PJ
Demand	TPSCOA	PJ	13414

*Blue cells are linked to the energy balance. Here, the demand value is equivalent to the sum of Total Conversion plus Total Final consumption.

Figure 30. Definition of base year demand values

3.1.3 Solving the model

The model is solved via the FE Case manager (**Basic Functions, Case Manager**), explained more in detail in Section 2.4.5.

For all step models, all cases (runs) are pre-defined by default (Figure 31) with a name and a description (here, DemoS_001; Demo Step 001), the components to be included in the run (BASE, SysSettings), the Regions (REG1), the Ending Year (2006), and the Period Defs (Pdef-1). It is important to note that the BASE component represents all the base year information included in all B-Y Templates together (only VT_REG_PRI_V01 in this example).

The optimizer options (**CPLEX button**) and the model variants (**Control Panel**) are also set by default. The model can be launched by clicking the **SOLVE** button. The model will be solved using the TIMES source code indicated under **GAMS Source Code folder** and the results files stored in the folder indicated below **GAMS Work folder**.

3.1.4 Results analysis in VEDA-BE

The results of a model run in VEDA-FE can be imported into VEDA-BE through the VEDA-BE menu command **Results, Import/Archive**.

A results database distributed with the Demo models called **DemoS_VBE** already contains pre-defined tables as well as commodity/process sets and all step runs described in this manual. This database should be pasted into the VEDA-BE/Databases folder. Then it can be opened in VEDA-BE by selecting **Open Database** from the **File** menu and browsing to the VEDA-BE/Databases/ DemoS_VBE folder.

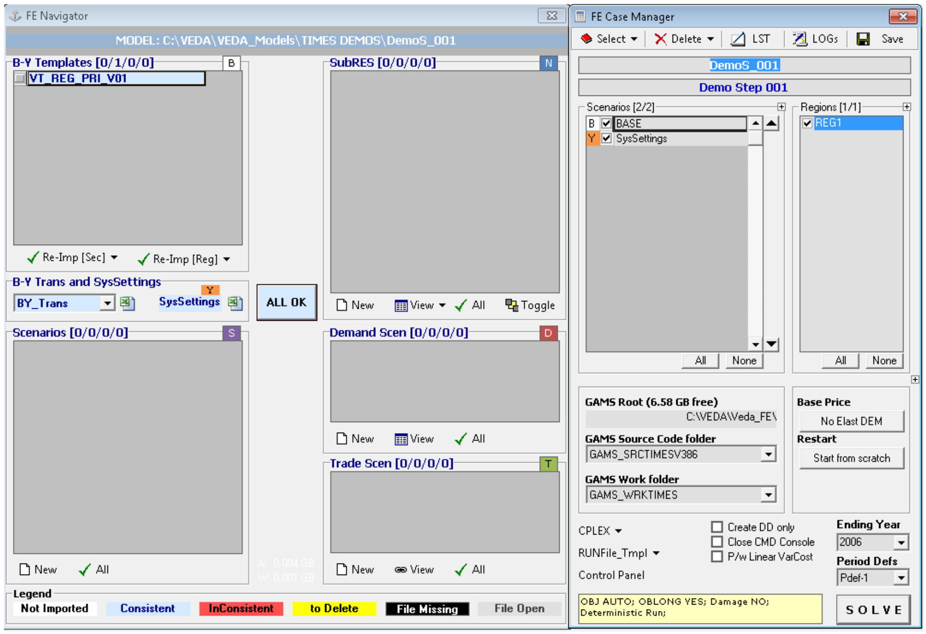


Figure 31. VEDA-FE Case Manager for submitting model runs

The list of pre-defined tables can be seen under **Table definition** at the top left of the main window. To view a particular table, scroll down/up the list and select it, then click the **View Table(s)** button. The table will open with a pre-defined layout than can be modified in a very flexible manner. Not all of the tables can be used for the first demo steps, in which only few results and information will be available. If a VEDA-BE table is inconsistent or empty you will get a pop up message saying that table is empty.

For more information on the capabilities and use of VEDA-BE, see Part V of the TIMES documentation.

The main VEDA-BE tables that can be checked for the first DemoS are:

- __Check Dummy Imports (Figure 32)

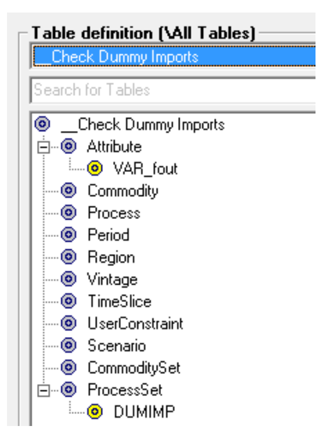


Figure 32. __Check Dummy Imports table in VEDA-BE

- o In a healthy model this table should be empty. If not, it means the model has some infeasibilities and is using some dummy technologies (built by default in VEDA-FE) to satisfy the commodity/demand production.
- o This table is built by selecting the attribute VAR_FOUT and the ProcessSet DUMIMP (this is a user-defined process set).

- _SysCost
  - o This table (Figure 33), built selecting the attribute Reg_Obj, shows the total system cost discounted to the G_DYEAR defined in the SysSettings file (in this example 2005). Figure 34 shows the total system cost in million euros for the model run to 2006, based on two periods (2005 and 2006) for a total of three years.

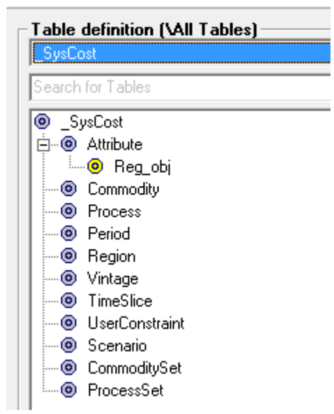


Figure 33. _SysCost table in VEDA-BE

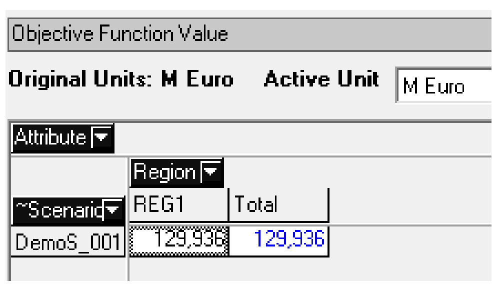


Figure 34. Total system cost in DemoS_001

- o The Scenario label shows the scenario name (DemoS_001) for the run we are viewing, while under the column Region we see the region name (REG1) and the value of the objective function. The column Total is shows the total by row (over regions). In this case, we only have the single region REG1, so the value is the same.
- o The _SysCost table provides a key model run indicator. In TIMES models, the Objective-Function is to minimize the total discounted cost of the system, properly augmented by the ‘cost’ of lost demand (when using the elastic demand features). See Parts I and II of the TIMES documentation for more on the model objective function.

- All costs
    - This table can be used to show the undiscounted cost elements of the model solution (Figure 35).

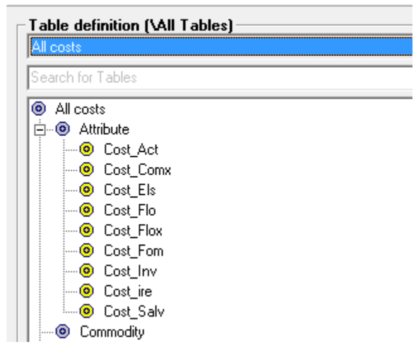
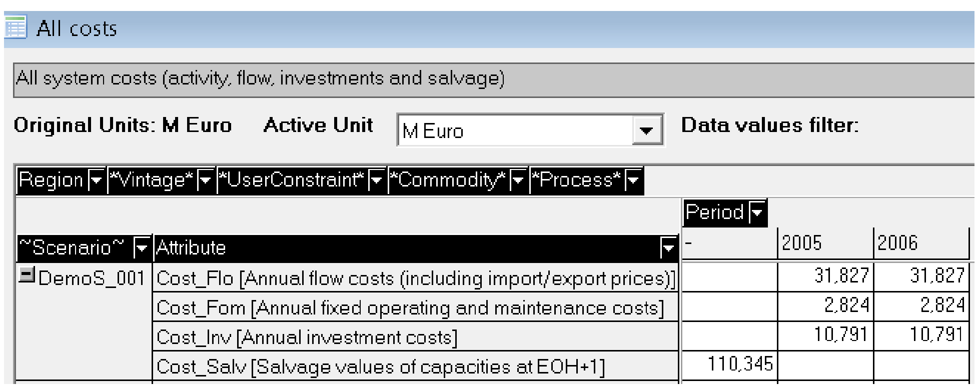


Figure 35. All costs table in VEDA-BE

    - The cost elements, each an individual attribute selected in the table definition, comprise capital costs for investing in and/or dismantling processes (Cost_Inv), fixed O&M costs (Cost_Fom), activity costs (Cost_Act), flow costs including import and export prices (Cost_Flo), implied costs of endogenous trade (Cost_ire), taxes and subsidies (Cost_Flox, Cost_Comx), salvage value of processes and commodities at the end of the planning horizon (Cost_Salv), and welfare loss resulting from reduced end-use demands (Cost_Els).
    - The undiscounted cost elements (in million euros) that are part of the solution for this first step for REG1 are shown below (Figure 36).


All costs

All system costs (activity, flow, investments and salvage)

Original Units: M Euro   Active Unit: M Euro   Data values filter:

Region | *Vintage* | *UserConstraint* | *Commodity* | *Process*

~Scenario~	Attribute	Period: -	2005	2006
DemoS_001	Cost_Flo [Annual flow costs (including import/export prices)]		31,827	31,827
	Cost_Fom [Annual fixed operating and maintenance costs]		2,824	2,824
	Cost_Inv [Annual investment costs]		10,791	10,791
	Cost_Salv [Salvage values of capacities at EOH+1]	110,345		

Figure 36. All system costs results by element

    - The attribute column in this case shows both the attribute name and description, while the Period columns show the value of each attribute in each model period, except the salvage value (Cost_Salv), which does not take a period index.
- Demands (Figure 37)
    - This table is used to show the energy service demand(s). In this case there is only the single demand called TPSCOA, which is in PJ (Figure 38).

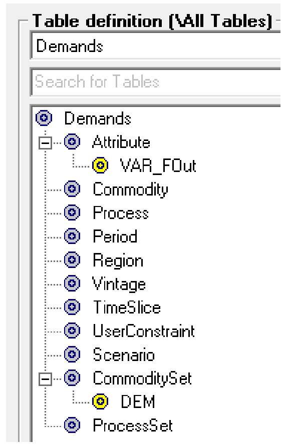


Figure 37. Demands table in VEDA-BE

- The Demands table shows, from left to right, for the scenario DemoS_001, region REG1, process (or technology) DTPSCOA, a flow out (Var_FOut – production or output from the process) for the commodity Demand Total Primary Supply – COA (TPSCOA), the values for the periods 2005 and 2006.

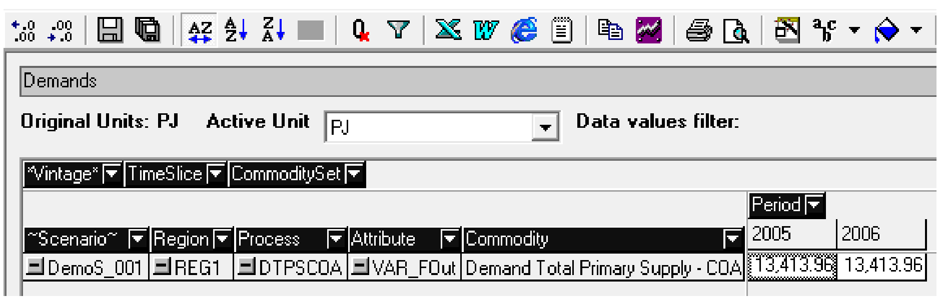


Figure 38. TPSCOA demand

- Fuel Supply (Figure 39)
  - This table is built selecting the attribute VAR_FOut (flow out) and the process set IRE (that includes all the process defined in ~FI_PROCESS tables as MIN, IMP and EXP). In other words, this table can be used to check the output from all the processes that belong to import and mining sets. The export process is characterised with an input and not an output, so it not possible to check the behavior of the export process by selecting only VAR_FOut.
  - The COA demand is met in a significant proportion with imports (6,462.67 PJ) and the rest with domestic resources through the first two steps of the supply curve. (The third step is not used, because it has higher COST than the imports, see Figure 40.) The demand and supply balance of COA is constant between 2005 and 2006, as described above in Section 3.1.2.5.

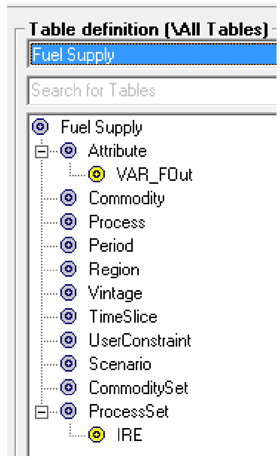
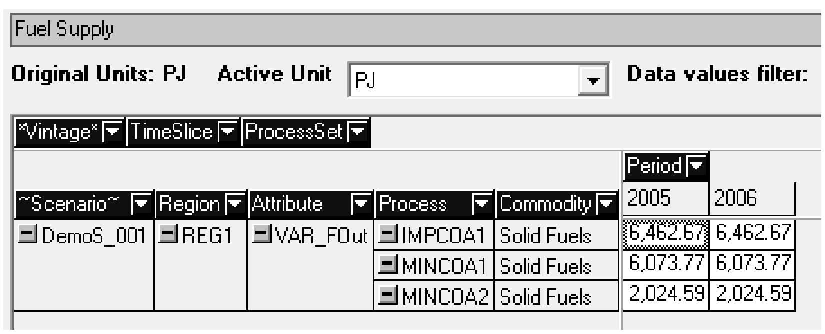


Figure 39. Fuel Supply table in VEDA-BE

- In this example the marginal technology, that is, the technology that would produce the next additional unit of the COA commodity, is the import technology. This information will be reflected in the commodity marginal price for COA, which will be equal to the production cost of the COA commodity from the marginal technology.


Fuel Supply

Original Units: PJ   Active Unit PJ   Data values filter:

Vintage TimeSlice ProcessSet

~Scenario~	Region	Attribute	Process	Commodity	2005	2006
DemoS_001	REG1	VAR_FOut	IMPCOA1	Solid Fuels	6,462.67	6,462.67
			MINCOA1	Solid Fuels	6,073.77	6,073.77
			MINCOA2	Solid Fuels	2,024.59	2,024.59

Figure 40. Fuel Supply results by process and period

- Prices_All
  - This table (Figure 41), built selecting the attribute EQ_CombalM, can be used for showing commodities' marginal prices in the run.

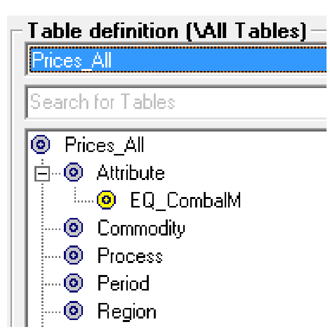


Figure 41. Marginal prices table in VEDA-BE

- As noted above, the marginal price of COA (solid fuels) is the same as the production cost from the marginal technology (import of solid fuels). In this example, it is 2.75 MEuro/PJ in both periods (Figure 42). The marginal price of TPSCOA (Demand Total Primary Supply – COA) in 2005 depends on the new capacity investment that must happen in that year to serve the demand. The marginal price for 2005 can be calculated by taking in account the marginal prices of the solid fuels commodity, the investment cost of the demand technology, the operating cost for the demand technology, and finally the salvage cost. In 2006 there isn't any new investment, so the marginal price will be only a function of the fuel cost.

Attribute				Period	
~Scenario~	Region	Commodity	TimeSlice	2005	2006
DemoS_001	REG1	Demand Total Primary Supply - COA	ANNUAL	5.65	2.75
		Solid Fuels	ANNUAL	2.75	2.75

Figure 42. Marginal prices results for DemoS_001

3.2 DemoS_002 - More demand options and multiple supply curves

Description. The second step model includes a greater number of supply, demand, import and export options for additional commodities in a single region over two time periods.

Objective. The objective is to show how to expand the model with more examples of commodities (energy and emissions) and of typical processes along with their respective attributes, including emission coefficients. On the supply side, it includes more three-step supply curves (e.g., for oil & gas in addition to coal), extraction processes, and import and export options, as well as the introduction of new sector fuel processes (processes used to change fuel names into sectoral commodity names). The demand side is also expanded with the presentation of two demands for energy services (residential and transportation) and corresponding end-use devices in each sector. Emission commodities (e.g. CO_2) and emission tracking are also introduced at the end-use device level in both the residential and transport sectors.

Attributes introduced:	Files updated
STOCK ENV_ACT START	VT_REG_PRI_v02

Files. The second step model is built by modifying the B-Y Template (VT_REG_PRI_V02) to add processes as well as energy and emission commodities. The SysSettings file is the same as in the DemoS_001.

3.2.1 B-Y Templates

3.2.1.1 EnergyBalance sheet

The energy balance is the same as in the first step although a larger portion is covered in this second step model (Figure 43). In addition to the primary supply of solid fuels (COA), the model covers the primary supply of natural gas (GAS) and crude oil (OIL) as well as the demand for GAS and OIL in the residential and transportation sectors (rather than for the aggregated primary supply as for COA).

A higher degree of disaggregation is also provided. On the supply side, the same level of disaggregation as for COA is provided for GAS and OIL, with shares to split the total domestic production in more than one step. On the demand side, fuel consumption is split by sector and by end use in the residential sector (space heating, appliances, and other). GAS is allocated at 100% to the Other end use in the residential sector and OIL at 100% to the single end use D1 in the transportation sector.

		COA	GAS	OIL	NUC	RNW	SLU	HET	ELC	
		Solid Fuels	Natural Gas	Crude Oil	Nuclear Energy	Renewable Energies	Industrial Wastes	Derived Heat	Electricity	Total
	PRIMARY									
MIN	Domestic Supply	8098	7899	5379	10775	5027	0	0	0	37178
IMP	Imports	6463	13292	39960	0	113	0	0	1168	60995
EXP	Exports	-1147	-2516	-14831	0	-72	0	0	-1127	-19693
TPS	Total Primary Supply	13414	18675	30508	10775	5067	0	0	41	78480
	CONVERSION									
ESC	Energy Sector Consumption	-58	-793	-1849	0	-4	-2	0	0	-2705
ELC	Electricity Plants	-9598	-5636	-1225	-10775	-1256	-33	1738	11581	-15203
HPL	Heat Plants	-161	-301	-50	0	-140	-2	659	0	5
REF	Petroleum Refineries			-31736						-31736
	Total Conversion	-9817	-6730	-34859	-10775	-1400	-36	2396	11581	-49640
	FINAL									
RSD	Residential	357	5160	2289	0	1294	0	865	2872	12837
COM	Commercial	57	1752	855	0	67	1	255	2527	5514
IND	Industry	1897	4437	2016	0	722	117	634	4088	13911
AGR	Agriculture	44	201	797	0	63	0	16	19	1141
TRA	Transport	1	21	14851	0	131	0	0	266	15270
OTH	Other	1189	0	393	0	1390	0	627	650	4249
NEN	Non Energy	52	634	4073		0	0	0	0	4759
BNK	Bunkers	0	0	2111		0	0	0	0	2111
TFC	Total Final Consumption	3597	12205	27385	0	3667	118	2396	10423	59791

Data used in the template to buld the model

	COA	GAS	OIL
Domestic Supply Curve Share - Step 1	75%	50%	80%
Domestic Supply Curve Share - Step 2	25%	50%	20%

Sector	Break-out by end-use	Solid Fuels	Natural Gas	Crude oil	Nuclear Energy	Renewable Energies	Industrial Wastes	Derived Heat	Electricity	
RSD	SH									Space Heating
RSD	AP									Appliancens
RSD	OT		1							Other
COM	D1									Demand 1
COM	D6									Demand 6
TRA	D1			1						Demand 1

Figure 43. Energy balance at start year 2005 for REG1 – Covered in DemoS_002

3.2.1.2 Pri_COA/GAS/OIL sheets

These new Pri_GAS and the Pri_OIL sheets have exactly the same structure as the Pri_COA sheet (which has not been modified from the first step) including:

- A commodity table to declare additional energy commodities (NRG): GAS - Natural gas (PJ) and OIL - Crude oil (PJ).
- A process table to declare additional supply options for GAS and OIL: mining processes (MINGAS* and MINOIL*), import processes (IMPGAS1, IMPOIL1), and export processes (EXPGAS1, EXPOIL1).
- A flexible import table to describe the primary supply options for GAS and OIL: 3-step domestic supply curves through three mining processes, as well as import and export options. All are characterized with the same attributes.

3.2.1.3 Sector_Fuels sheet

This is a new sheet that is used to construct sector fuel processes (FTE-*), which produce sector fuels from primary fuels, e.g.: GAS becomes RSDGAS and OIL becomes TRAOIL in this example (Figure 44). This is done to make it easy to track fuel consumption at the sectoral level as well as to add sectoral emissions (which could be constrained separately). These technologies can be also used to add additional information on the sectoral commodities, for example additional costs to simulate a sectoral tariff for GAS or an investment cost to simulate new investments in infrastructure and so on. The same approach is used to declare the new commodities and processes in their respective tables.

		~FI_T			
TechName	Comm-IN	Comm-OUT	STOCK	EFF	LIFE
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Lifetime
*Units			PJa		Years
FTE-RSDGAS	GAS	RSDGAS		1.00	30
FTE-TRAOIL	OIL	TRAOIL		1.00	30

Figure 44. Introduction of sector fuel processes

3.2.1.4 DemTechs_RSD and DemTechs_TRA sheets

Demand processes (DMD) are introduced in these sheets (Figure 45). They consume an energy commodity (RSDGAS, TRAOIL) to produce directly the energy service commodity: residential–other (DROT) and transport (DTD1) in this example. In both sectors, there are existing (ROTEGAS and TOTEOIL) and new processes (ROTNGAS and TOTNOIL).

- The existing processes are characterized with their existing installed capacity (STOCK), corresponding in this case to the energy consumption required to produce these energy services in the base year as given by the energy balance and the additional fuel split assumptions. They also have an efficiency (EFF), an annual availability factor (AFA) and a life time (LIFE).
- Existing processes characterised in VEDA B-Y Templates with a base year STOCK cannot increase their capacity endogenously through new investment because when synchronizing the templates, by default VEDA-FE inserts the attribute NCAP_BND with interpolation/extrapolation rule number 2, setting an upper bound of EPS (epsilon, or effectively zero) for all years. (For more information on interpolation/extrapolation see

Table 8 in Section 3.3.2.2) New technologies thus are needed to replace the existing capacity as it retires or increase the amount of capacity available after the base year.

- The new processes do not have an existing installed capacity, but they are available in the database to be invested in to replace the existing ones and meet the demand for energy services. They are characterized with an investment cost (INVCOST), a fixed operation and maintenance cost (FIXOM), and the year in which they become available (START). The model can invest in these new technologies only beginning in that START year.
- Finally, emission commodities (ENV) are also introduced along with these processes: CO2 emissions in the residential (RSDCO2) and the transport (TRACO2) sectors in this example (in kt). An emission coefficient (ENV_ACT in kt/PJ_{output}) is provided for each process based on the technology output. It is also possible to define emissions coefficients based on fuel input (see Section 3.7.2.7).

		~FI_T								
TechName	Comm-IN	Comm-OUT	STOCK	EFF	AFA	INVCOST	FIXOM	LIFE	START	ENV_ACT
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Lifetime	Start Year	Activity Emission Coefficient
*Units			PJa			M€2005/PJ	M€2005/PJa	Years	Year	kt/PJ
ROTEGAS	RSDGAS	DROT	5486	1.00	0.95		0.24	10		
		RSDCO2								56.1
ROTNGAS	RSDGAS	DROT		1.20	0.95	12	0.24	20	2006	
		RSDCO2								46.8

		~FI_T								
TechName	Comm-IN	Comm-OUT	STOCK	EFF	AFA	INVCOST	FIXOM	LIFE	START	ENV_ACT
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Lifetime	Start Year	Activity Emission Coefficient
*Units			PJa			M€2005/PJ	M€2005/PJa	Years	Year	kt/PJ
TOTEOIL	TRAOIL	DTD1	16666	1.00	0.90		0.20	10		
		TRACO2								65
TOTNOIL	TRAOIL	DTD1		1.10	0.90	10	0.20	15	2006	
		TRACO2								59

Figure 45. End-use demand processes

3.2.1.5 Demands sheet

The demand table is expanded to include the demand for the new energy services created at this step: residential–other (DROT) and transport (DTD1). The 2005 values come from the energy balance sheet and then will be constant, as explained in Section 3.1.2.5, until new data is input for future years.

3.2.2 Results

There are more demands for energy services (Figure 46) and fuel supply options (Figure 47) in this second step model compared with the first step. Also, a new piece of information available at this second step is CO2 emissions by sector (Figure 48), which are computed from the input coefficients provided for each process and the activity of each process. These three tables can be viewed in the same way as explained for DemoS_001, and if results for both DemoS_001 and DemoS_002 have been imported, then it will be possible to see and compare results for the two scenarios.The main findings from the results analysis are:

- The domestic demand for transportation (DTC1) represents the major proportion (44%) of total domestic demand for energy. This sector relies on oil and also accounts for the largest part of the CO2 emissions (TRACO2), although no coefficient was provided for solid fuels combustion emissions.

Demands

Original Units: PJ **Active Unit** PJ **Data values filter:**

Attribute | *Vintage* | TimeSlice | CommoditySet

~Scenario~	Region	Commodity	Process	2005	2006
DemoS_001	REG1	TPSCOA [Demand Total Primary Supply - COA]	DTPSCOA	13,414	13,414
			Total	13,414	13,414
DemoS_002	REG1	DROT [Demand Residential Sector - Other]	ROTEGAS	5,160	4,690
			ROTNGAS		470
			Total	5,160	5,160
		DTD1 [Demand Transport Sector - Demand 1]	TOTEOIL	14,851	13,500
			TOTNOIL		1,351
			Total	14,851	14,851
		TPSCOA [Demand Total Primary Supply - COA]	DTPSCOA	13,414	13,414
			Total	13,414	13,414

Figure 46. Results - Demands table for DemoS_002

Fuel Supply

Original Units: PJ **Active Unit** PJ **Data values filter:**

Attribute | *Vintage* | TimeSlice | ProcessSet

~Scenario~	Region	Commodity	Process	2005	2006
DemoS_001	REG1	COA [Solid Fuels]	IMPCOA1 [Import of Solid Fuels Step 1]	6,463	6,463
			MINCOA1 [Domestic Supply of Solid Fuels Step 1]	6,074	6,074
			MINCOA2 [Domestic Supply of Solid Fuels Step 2]	2,025	2,025
			Total	14,561	14,561
DemoS_002	REG1	COA [Solid Fuels]	IMPCOA1 [Import of Solid Fuels Step 1]	6,463	6,463
			MINCOA1 [Domestic Supply of Solid Fuels Step 1]	6,074	6,074
			MINCOA2 [Domestic Supply of Solid Fuels Step 2]	2,025	2,025
			Total	14,561	14,561
		GAS [Natural Gas]	MINGAS1 [Domestic Supply of Natural Gas Step 1]	3,950	3,950
			MINGAS2 [Domestic Supply of Natural Gas Step 2]	3,726	3,648
			Total	7,676	7,598
		OIL [Crude Oil]	IMPOIL1 [Import of Crude Oil Step 1]	24,303	2,418
			MINOIL1 [Domestic Supply of Crude Oil Step 1]	4,303	9,849
			MINOIL2 [Domestic Supply of Crude Oil Step 2]	1,076	2,462
			Total	29,682	14,728

Figure 47. Results – Fuel Supply table for DemoS_002

- The demand for residential–other (DROT) and transportation (DTC1) is first fully satisfied with the existing demand processes (ROTEGAS and TOTEOIL) in the base year 2005, but the new demand processes (ROTNGAS and TOTNOIL) start penetrating in 2006. The new processes are more efficient and require less energy to satisfy the demand. The existing processes satisfy less demand in 2006 because their STOCK in 2006 is lower than in 2005. The STOCK decreases between the base year value and zero linearly over the technical LIFE. For example, for ROTEGAS the base (2005) stock is 5486 PJ and will be zero in 2015 (because the residual technical life is 10 years). The stock value between 2005 and 2015 is linearly interpolated between 5486 PJ and 0 PJ.
- A large proportion of the oil imported in 2005 is destined to export markets (exports reach their upper limit because the export price is no higher than that of the marginal oil supply, the import price), while in 2006 the demand from export markets decreases to

zero and more oil is produced domestically to meet the domestic demand for transportation oil.

Emissions by Sector

Original Units: Kt **Active Unit** Kt **Data values filter:**

Attribute | *Process* | *Vintage* | TimeSlice | CommoditySet

~Scenario~	Region	Commodity	Period 2005	2006
DemoS_002	REG1	RSDCO2 [Residential Carbon dioxide]	289,464	285,074
		TRACO2 [Transport Carbon dioxide]	965,331	957,345
		Total	1,254,796	1,242,419

Figure 48. Results – Emissions by Sector table for DemoS_002

Objective-Function = 496 637 M euros (see the _SysCost table in VEDA-BE).

All the system cost components can be seen from the VEDA-BE table **All costs**. As the model includes different types of energy commodities, it is relevant to have a look at their respective marginal prices (Figure 49). Marginal prices of oil are the highest due to higher production costs and import prices. Marginal (shadow) prices for process activity (Figure 50) allow us to understand why the third step of the supply curve for fossil fuels (MINCOA3, MINGAS3, MINOIL3) are not part of the optimal solution, as they are more expensive. For example the VAR_ActM for MINCOA1 is -0.75. This means that if we relax the upper activity bound of this technology of by GJ than the objective function will decrease by 0.75 euros, while forcing the production of 1 GJ from MINCOA3 will increase the objective function by 0.25 euros.

In TIMES, the shadow prices of commodities play a very important diagnostic role. If some shadow price is clearly out of line (i.e., if it seems much too small or too large compared to anticipated market prices), this indicates that the database may contain some errors. For instance, if the shadow price of a commodity is zero and the quantity supplied is non zero, as pointed out by the second theorem of Linear Programming, it means that there is more supply than demand for that commodity. The examination of shadow prices is just as important as the analysis of the quantities produced and consumed of each commodity and of the technological investments.

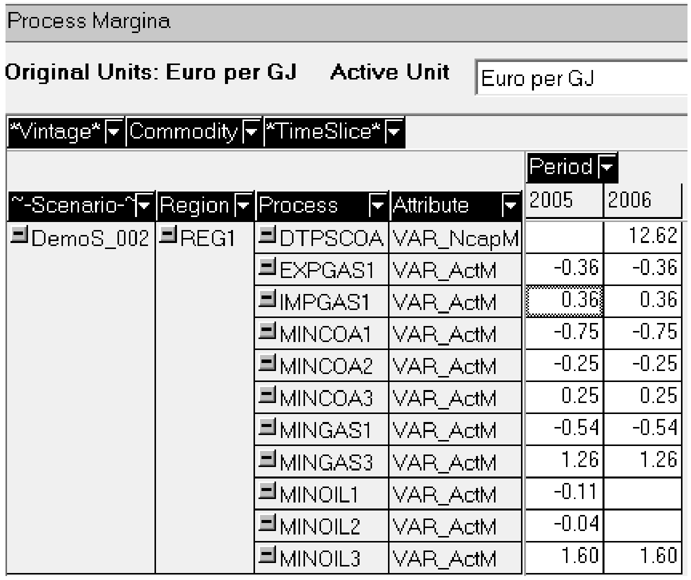
Energy Prices

Original Units: Euro per GJ **Active Unit** Euro per GJ **Data values**

Attribute | CommoditySet

~Scenario~	Region	Commodity	TimeSlice	Period 2005	2006
DemoS_001	REG1	COA [Solid Fuels]	ANNUAL	2.75	2.75
DemoS_002	REG1	COA [Solid Fuels]	ANNUAL	2.75	2.75
		GAS [Natural Gas]	ANNUAL	4.14	4.14
		OIL [Crude Oil]	ANNUAL	8.00	8.00
		RSDGAS [Residential Natural Gas]	ANNUAL	4.14	4.14
		TRAOIL [Transport Crude Oil]	ANNUAL	8.00	8.00

Figure 49. Results – Prices_Energy table for DemoS_002



Process Margina

Original Units: Euro per GJ Active Unit Euro per GJ

Vintage | Commodity | *TimeSlice*

~-Scenario-~	Region	Process	Attribute	Period 2005	2006
DemoS_002	REG1	DTPSCOA	VAR_NcapM		12.62
		EXPGAS1	VAR_ActM	-0.36	-0.36
		IMPGAS1	VAR_ActM	0.36	0.36
		MINCOA1	VAR_ActM	-0.75	-0.75
		MINCOA2	VAR_ActM	-0.25	-0.25
		MINCOA3	VAR_ActM	0.25	0.25
		MINGAS1	VAR_ActM	-0.54	-0.54
		MINGAS3	VAR_ActM	1.26	1.26
		MINOIL1	VAR_ActM	-0.11	
		MINOIL2	VAR_ActM	-0.04	
		MINOIL3	VAR_ActM	1.60	1.60

Figure 50. Results – marginal process activity prices in DemoS_002

3.3 DemoS_003 - Power sector: basics

Description. The third step model demonstrates the modelling of a simple power sector in a single region over more than two time periods. From the base year of 2005, the time horizon is expanded from 2006 to 2020.

Objective. The objective is to show how to model a typical power sector with different types of power plants (e.g., thermal, nuclear and renewable) along with their respective attributes and the transmission efficiency of the network. Other objectives are to add more time periods, to show how to project future demands (e.g. constant or growing), and to explain the powerful interpolation/extrapolation rules existing in VEDA-TIMES, as well as the difference between model years and data years.

Attributes introduced:	Files updated
COM_IE	SysSettings
CAP2ACT	VT_REG_PRI_v03

Files. The third step model is built by modifying:

1. the SysSettings file to add more time periods and declare the transmission efficiency of the electricity network.
2. the B-Y Template (VT_REG_PRI_V03) to model the power sector and insert interpolation/extrapolation rules.

3.3.1 SysSettings file

3.3.1.1 TimePeriods sheet

The ~TimePeriods table is used to extend the time horizon of the model by adding three active periods of five years each (Figure 51). These specifications are saved as a new time period definition (Pdef-5). The time horizon is extended to 2020, with the milestones years being 2005, 2006, 2010, 2015 and 2020.

~TimePeriods	
Pdef-1	Pdef-5
1	1
2	2
	5
	5
	5

Figure 51. New time periods definition in the SysSettings file

The changes to the period definitions can also be seen within VEDA-FE, by selecting **MileStone Years** from the **Advanced Functions** menu, as shown in Figure 52. In addition to showing the currently selected definitions, the MileStone years tool can be used to modify the definition or add a new set of definitions in an easy way. Input desired period lengths in the yellow area, adding new periods below the currently defined ones as desired. These changes will be reflected in the period specifications shown on the right. You may then choose to **Save** and overwrite the previous definition, or enter a new name in the dropdown menu in the lower left hand corner of the form to create a new set. These changes will then be reflected in the SysSettings file.

BaseYear							
2005			1	2	3	4	5
PeriodLength	Start		2005	2006	2008	2013	2018
1	Mid		2005	2006	2010	2015	2020
2	End		2005	2007	2012	2017	2022
5	Lnth		1	2	5	5	5
5							
5							

Figure 52. Milestones years from the new time period definition

With the introduction of the interpolation/extrapolation rules, it is possible to run the model on a longer time horizon without having to declare data values for all periods up to 2020.

3.3.1.2 Constants sheet

The transformation table is also used to insert a new constant in the model: the transmission efficiency (COM_IE) for the electricity (ELC) commodity in REG1 (Figure 53).

~TFM_INS					
TimeSlice	LimType	Attribute	AllRegions	REG1	Cset_CN
		G_DYEAR	2005		
		Discount	0.05		
ANNUAL		YRFR	1.00		
		COM_IE		0.90	ELC

Figure 53. New constant declarations in the SysSettings file

3.3.2 B-Y Templates

3.3.2.1 EnergyBalance sheet

The energy balance is the same as in the second step although a larger portion of it is covered in this third step model (Figure 54). The energy used for conversion into electricity and the total electricity generation are now included.

		COA	GAS	OIL	NUC	RNW	SLU	HET	ELC	
		Solid Fuels	Natural Gas	Crude Oil	Nuclear Energy	Renewable Energies	Industrial Wastes	Derived Heat	Electricity	Total
	PRIMARY									
MIN	Domestic Supply	8098	7899	5379	10775	5027	0	0	0	37178
IMP	Imports	6463	13292	39960	0	113	0	0	1168	60995
EXP	Exports	-1147	-2516	-14831	0	-72	0	0	-1127	-19693
TPS	Total Primary Supply	13414	18675	30508	10775	5067	0	0	41	78480
	CONVERSION									
ESC	Energy Sector Consumption	-58	-793	-1849	0	-4	-2	0	0	-2705
ELC	Electricity Plants	-9598	-5636	-1225	-10775	-1256	-33	1738	11581	-15203
HPL	Heat Plants	-161	-301	-50	0	-140	-2	659	0	5
REF	Petroleum Refineries			-31736						-31736
	Total Conversion	-9817	-6730	-34859	-10775	-1400	-36	2396	11581	-49640
	FINAL									
RSD	Residential	357	5160	2289	0	1294	0	865	2872	12837
COM	Commercial	57	1752	855	0	67	1	255	2527	5514
IND	Industry	1897	4437	2016	0	722	117	634	4088	13911
AGR	Agriculture	44	201	797	0	63	0	16	19	1141
TRA	Transport	1	21	14851	0	131	0	0	266	15270
OTH	Other	1189	0	393	0	1390	0	627	650	4249
NEN	Non Energy	52	634	4073		0	0	0	0	4759
BNK	Bunkers	0	0	2111		0	0	0	0	2111
TFC	Total Final Consumption	3597	12205	27385	0	3667	118	2396	10423	59791

Data used in the template to buld the model

Figure 54. Energy balance at start year 2005 for REG1 – Covered in DemoS_003

3.3.2.2 Pri_COA/GAS/OIL sheets

These sheets were all modified in a similar way to show the use of interpolation/extrapolation rules in VEDA-TIMES (Figure 55). With the introduction of the interpolation/extrapolation rules, it is possible to run the model for a longer time horizon without having to declare data values for all periods up to 2020.

To activate an interpolation/extrapolation (I/E) rule for a specific process, insert a data row and write a "0" as the Year. In this example, an interpolation/extrapolation rule will be enabled for the processes MINCOA1, MONCOA2 and EXPCOA1. Then, an interpolation/extrapolation code is indicated under the attribute. In this example, option 5 will be applied to the activity bound (ACT_BND) of these processes. The option codes for the interpolation/extrapolation rules are presented in Table 8. The code 5 means full interpolation and forward extrapolation of the attribute.

In this example, MINCOA1 has an activity bound of 6074 PJ in the year 2005, and due to the I/E rule, the 2005 value is kept constant over the time horizon. Just remember that the ACT_BND is not I/E by default, so when no I/E rule is explicitly specified in the template, the bound will be applied only to the periods defined in the year column.

Default interpolation/extrapolation mechanisms are embedded in the TIMES code itself (for more information see Section 3.1.1 of Part II of the TIMES documentation). It is also useful to check the Attribute Master table in VEDA-FE (see Section 2.4.4) for more information about which attributes are interpolated/extrapolated by default and which are not.

				~FI_T			
TechName	Comm-IN	Comm-OUT	Year	LimType	CUM	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity			Reserves Cumulative Value	Cost	Annual Production Bound
*Units					PJ	M€2005/PJ	PJ
MINCOA1		COA	2005		80000	2.00	6074
			0				5
MINCOA2		COA	2005		160000	2.50	2025
			0				5
MINCOA3		COA			320000	3.00	
IMPCOA1		COA				2.75	
EXPCOA1	COA		2005	UP		2.75	1147
			0	UP			5

Figure 55. PRI_COA sheet with interpolation/extrapolation rules

Table 8. Interpolation/extrapolation codes in TIMES

Option code	Action	Applies to
0 (or none)	Interpolation and extrapolation of data in the default way as predefined in TIMES (see below)	All
< 0	No interpolation or extrapolation of data (only valid for non-cost parameters).	All
1	Interpolation between data points but no extrapolation.	All
2	Interpolation between data points entered, and filling-in all points outside the interpolation window with the EPS value.	All
3	Forced interpolation and both forward and backward extrapolation throughout the time horizon.	All
4	Interpolation and backward extrapolation	All
5	Interpolation and forward extrapolation	All
10	Migrated interpolation/extrapolation within periods	Bounds, RHS
11	Interpolation migrated at end-points, no extrapolation	Bounds, RHS
12	Interpolation migrated at ends, extrapolation with EPS	Bounds, RHS
14	Interpolation migrated at end, backward extrapolation	Bounds, RHS
15	Interpolation migrated at start, forward extrapolation	Bounds, RHS
YEAR (≥ 1000)	Log-linear interpolation beyond the specified YEAR, and both forward and backward extrapolation outside the interpolation window.	All

3.3.2.3 <u>Pri_RNW and Pri_NUC sheets</u>

As with supply curves for fossil fuels, mining processes are created for the uranium resources and the renewable potential (Figure 56). They are considered unlimited and at no cost in this simple example.

				~FI_T			
TechName	Comm-IN	Comm-OUT	Year	LimType	CUM	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity			Reserves Cumulative Value	Cost	Annual Production Bound
*Units					PJ	M€2005/PJ	PJ
MINRNW1		RNW					

				~FI_T			
TechName	Comm-IN	Comm-OUT	Year	LimType	CUM	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity			Reserves Cumulative Value	Cost	Annual Production Bound
*Units					PJ	M€2005/PJ	PJ
MINNUC1		NUC					

Figure 56. Description of new supply options

3.3.2.4 Sector_Fuels sheet

Additional sector fuel processes (FTE-*) are defined and characterized in this sheet, namely to produce the electricity sector fuels from primary fuels, including fossil fuels (e.g. COA to ELCCOA) and other sources (e.g. NUC to ELCNUC). The same approach is used to declare the new commodities and processes in their respective tables.

3.3.2.5 Con_ELC sheet

A series of processes are created to represent different types of power plants (Figure 57). These are conversion processes that consume electricity sector fuels (ELCGAS, ELCNUC, etc.) to produce electricity (ELC).

- The existing processes are characterized with their existing installed capacity (STOCK) in GW (calculated from the information given in the energy balance in terms of energy consumption for electricity production and technical attribute values). They also have an efficiency (EFF), an annual availability factor (AFA), fixed and variable O&M costs (FIXOM, VAROM), a life time (LIFE), and a CO2 emission coefficient (ENV_ACT).
- By default, all attribute values apply to the base year 2005 when not specified. It is possible to declare any attribute values for future years using the command "~" followed by the year, as for the installed capacity attribute in this case (STOCK~2030). By default, an existing installed capacity (STOCK) decreases to zero at the end of its lifetime (e.g., after 30 years for ELCTECOA00). By specifying an installed capacity value for 2030, as for ELCTENUC00, a new retirement profile is defined (constant in this example), and it is not necessary to specify a life duration.
- The new processes do not have an existing installed capacity, but they are available in the database to be invested in to replace the existing ones and meet the demand for electricity. They are characterized in addition with an investment cost (INVCOST) as well as the year where they become available (START).
- A new attribute is introduced (CAP2ACT) allowing the conversion between the process capacity and activity units. In this example a coefficient of 31.536 PJ/GW is needed (1GW * 365 days * 24 hours = 8760 GWh = 31.536 PJ). When not specified and when both capacity and activity are tracked in the same unit, the CAP2ACT is equal to 1.

The same approach is used to declare the new commodities and processes in their respective tables (Figure 58) including the declaration of existing and new power plants as ELE processes. The process names follow a convention where T=thermal, C=CHP, R=Renewable, N=Nuclear.

		~FI_T											
TechName	Comm-IN	Comm-OUT	STOCK	STOCK~2030	EFF	AFA	INVCOST	FIXOM	VAROM	LIFE	START	ENV_ACT	CAP2ACT
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Retirement Capacity	Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Variable O&M Cost	Lifetime		Activity Emission Coefficient	Capacity to Activity Factor
*Units			GW	GW			M€2005/GW	M€2005/GW	M€2005/PJ	Years		kt	PJ/GW
ELCTECOA00	ELCCOA	ELC	137		0.38	0.85		40.00	0.50	30			31.536
		ELCCO2										260	
ELCTEGAS00	ELCGAS	ELC	104		0.49	0.85		35.00	0.40	20			31.536
		ELCCO2										114	
ELCTEOIL00	ELCOIL	ELC	11		0.25	0.85		20.00	0.20	30			31.536
		ELCCO2										306	
ELCRERNW00	ELCRNW	ELC	88	88	1.00	0.45		70.00					31.536
ELCTENUC00	ELCNUC	ELC	125	125	0.33	0.90		38.00	0.27				31.536
ELCTNCOA00	ELCCOA	ELC			0.42	0.85	1650	35.00	0.40	40	2006		31.536
		ELCCO2										238	
ELCTNGAS00	ELCGAS	ELC			0.52	0.85	750	30.00	0.35	30	2006		31.536
		ELCCO2										108	
ELCTNOIL00	ELCOIL	ELC			0.30	0.85	250	15.00	0.20	40	2006		31.536
		ELCCO2										255	

Figure 57. Existing and new power plants

~FI_Comm								
Csets	Region	CommName	CommDesc	Unit	LimType	CTSLvl	PeakTS	Ctype
*Commodity Set Membership	Region Name	Commodity Name	Commodity Description	Unit	Sense of the Balance EQN.	Timeslice Level	Peak Monitoring	Electricity Indicator
NRG		ELC	Electricity	PJ				ELC
ENV		ELCCO2	Electricity Plants Carbon dioxide	kt				

~FI_Process								
Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
*								
ELE		ELCTECOA00	Power Plants Existing00 - Solid Fuels	PJ	GW			
		ELCTEGAS00	Power Plants Existing00 - Natural Gas	PJ	GW			
		ELCTEOIL00	Power Plants Existing00 - Crude Oil	PJ	GW			
		ELCRERNW00	Power Plants Existing00 - Renewable Energies	PJ	GW			
		ELCTENUC00	Power Plants Existing00 - Nuclear Energy	PJ	GW			
		ELCTNCOA00	Power Plants New00 - Solid Fuels	PJ	GW			
		ELCTNGAS00	Power Plants New00 - Natural Gas	PJ	GW			
		ELCTNOIL00	Power Plants New00 - Crude Oil	PJ	GW			

Figure 58. Declaration of electricity commodities and processes

3.3.2.6 DemTechs_ELC sheet

The total demand for electricity (ELC) is modelled in a simplistic manner as for solids fuels (COA). A flexible table is used to describe the demand device for electricity (Figure 59):

- A process for the total demand of ELC (DTPSELC) is characterized with an efficiency (EFF), an annual availability factor (AFA), an investment cost (INVCOST) a fixed operation and maintenance cost (FIXOM), and a life time (LIFE).

		~FI_T						
TechName	Comm-IN	Comm-OUT	STOCK	EFF	AFA	INVCOST	FIXOM	LIFE
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Lifetime
*Units			PJa			M€2005/PJ	M€2005/PJa	Years
DTPSELC	ELC	TPSELC		1.00	0.95	10	0.20	20

Figure 59. Description of a simple electricity demand processes

3.3.2.7 Demands

The end-use demand table is expanded to include the demand for electricity (TPSELC) in the base year as well as for future years (Figure 60). While the demand for other fuels or for energy services will be kept constant over time (extrapolated at a constant level by default), the demand for electricity is set up to increase by an annual growth rate of 1% through 2020.

	~FI_T							
Attribute	CommName	*Unit	2005	2006	2010	2015	2020	
*	Demand Commodity Name	Demand Unit	Demand Value					Demand Driver (annual growth)
*Units			PJ					
Demand	TPSCOA	PJ	3597					
Demand	DROT	PJ	5160					
Demand	DTD1	PJ	14851					
Demand	TPSELC	PJ	10423	10527	10955	11513	12101	1%

Figure 60. Definition of base year and future years demand values

3.3.3 Results

The demands for energy and energy services are extended to the 2020 horizon (Figure 61), increasing by 1% per year (TPSELC) or remaining constant (all others). The effects of the interpolation/extrapolation rules applied on the activity bound of certain supply processes can be seen below (

Figure 62). The activity of the first two mining processes (first two steps of the domestic supply curves) for fossil fuels (COA, GAS, OIL) is controlled by the annual activity bound (set constant for each period by the interpolation rule) and the cumulative bound (CUM). The combination of these two conditions leads to a significant increase in imports to meet the growing demand for energy. Exports are also kept constant using the same interpolation/extrapolation rules. More primary supply options exist now with the addition of the electric fuels such as nuclear and renewables.

Results from the new electricity sector are introduced (Figure 63 and Figure 64). The total generating installed capacity increases from 466.3 GW in 2005 to 541.6 GW in 2020. Most of this increase is coming from new coal-fired power plants (ELCTNCOA00), the most expensive process but the least expensive fuel. The installed capacity of nuclear and renewable power plants remain constant as specified in the B-Y Template. Electricity production is coming mainly from fossil fuels (64%), with a smaller contribution from nuclear (26%) and renewables (9%). The oil plants are working only in the base year, as calibrated to the energy balance, because the fuel is too expensive compared to the other available options.

Demands

Original Units: PJ **Active Unit** PJ **Data values filter:**

Attribute | *Process* | *Vintage* | TimeSlice | CommoditySet

Region	~-Scenario-~	Commodity	Period 2005	2006	2010	2015	2020
REG1	DemoS_002	DROT [Demand Residential Sector - Other]	5,160	5,160			
		DTD1 [Demand Transport Sector - Demand 1]	14,851	14,851			
		TPSCOA [Demand Total Primary Supply - COA]	13,414	13,414			
	DemoS_003	DROT [Demand Residential Sector - Other]	5,160	5,160	5,160	5,160	5,160
		DTD1 [Demand Transport Sector - Demand 1]	14,851	14,851	14,851	14,851	14,851
		TPSCOA [Demand Total Primary Supply - COA]	3,597	3,597	3,597	3,597	3,597
		TPSELC [Demand Total Primary Supply - ELC]	10,423	10,527	10,955	11,513	12,101

Figure 61. Results – demand for energy services in DemoS_003

Fuel Supply

Original Units: PJ **Active Unit** PJ **Data values filter:**

Attribute | *Vintage* | TimeSlice | ProcessSet

~-Scenario-~	Region	Commodity	Process	Period 2005	2006	2010	2015	2020
DemoS_003	REG1	COA [Solid Fuels]	IMPCOA1	6,244	7,551	9,895	12,890	21,826
			MINCOA1	6,074	6,074	6,074	6,074	208
			MINCOA2	2,025	2,025	2,025	2,025	2,025
		GAS [Natural Gas]	IMPGAS1	5,412	5,052	9,217	9,634	8,225
			MINGAS1	3,950	3,950	630		
			MINGAS2	3,950	3,950	1,630		
		NUC [Nuclear Energy]	MINNUC1	10,775	10,775	10,775	10,775	10,775
		OIL [Crude Oil]	IMPOIL1	25,528	24,181	26,241	28,332	28,332
			MINOIL1	4,303	4,303	2,218		
			MINOIL2	1,076	1,076	555		
		RNW [Renewable Energies]	MINRNW1	1,256	1,256	1,256	1,256	1,256

Figure 62. Results – fuel supply options in DemoS_003

ELC plants capacity and new capacity

Original Units: GW **Active Unit** GW **Data values filter:**

Attribute | *Vintage*

~Scenario~	Region	ProcessSet	Process	Period 2005	2006	2010	2015	2020
DemoS_003	REG1	ELECOA [Coal Power Plants]	ELCTECOA00	137	133	115	92	69
			ELCTNCOA00		26	82	154	227
		ELEGAS [Gas Power Plants]	ELCTEGAS00	104	98	78	52	26
		ELENUC [Nuclear Power Plants]	ELCTENUC00	125	125	125	125	125
		ELEOIL [Oil Power Plants]	ELCTEOIL00	11	11	10	8	6
		ELERNW [Renewable Power Plants]	ELCRERNW00	88	88	88	88	88
	Total			466	482	498	519	542

Figure 63. Results – electricity capacity in DemoS_003

ELC Plants Production

Original Units: PJ Active Unit Billion Kwh Data values filter:

Attribute | *Process* | *Vintage* | TimeSlice

~Scenario~	Region	ProcessSet	2005	2006	2010	2015	2020
DemoS_003	REG1	ELECOA [Coal Power Plants]	1,024	1,180	1,466	1,831	2,205
		ELEGAS [Gas Power Plants]	772	733	579	386	193
		ELENUC [Nuclear Power Plants]	988	988	988	988	988
		ELEOIL [Oil Power Plants]	85				
		ELERNW [Renewable Power Plants]	349	349	349	349	349
		Total	3,217	3,249	3,381	3,554	3,735

Figure 64. Results – electricity activity in DemoS_003

Objective-Function = 3,185,019 M euros (see the _SysCost table in VEDA-BE). This cost is significantly higher compared to the optimal cost obtained with DemoS_002 because of the addition of the electricity sector. All the system cost components can be seen in the VEDA-BE table **All costs**, as well as the marginal fuel prices in **Price_Energy** and the process activity in **Process Marginals**.

3.4 DemoS_004 - Power sector: sophistication

Description. The fourth step model expands the modelling to a more sophisticated power sector in the same single region over the 2020 horizon.

Objective. The objective is to introduce the concepts of time slices, peak, and peak reserve capacity. Time slices are added to the model to adequately capture the timing of the electricity demand, and the peak reserve capacity requirement is illustrated through scenario variants, with and without peak reserve capacity factor. This step model is also used to show how interpolation/extrapolation specifications can be moved to the SysSettings file and applied to all instances of an attribute in the model using a single declaration.

Attributes introduced:	Files updated
PEAK COM_FR COM_PEAK COM_PKRSV COM_PKFLX	SysSettings VT_REG_PRI_v04 Files created Scen_Peak_RSV Scen_Peak_RSV-FLX

Files. The forth step model is built:

1. by modifying the SysSettings file to add new time slices and to insert default interpolation/extrapolation options;
2. by modifying the B-Y Template (VT_REG_PRI_V04) to declare the contribution of power plants to the peak and add the load curve of electricity demand;
3. by creating scenario files to illustrate the peak reserve capacity requirement (Figure 65).

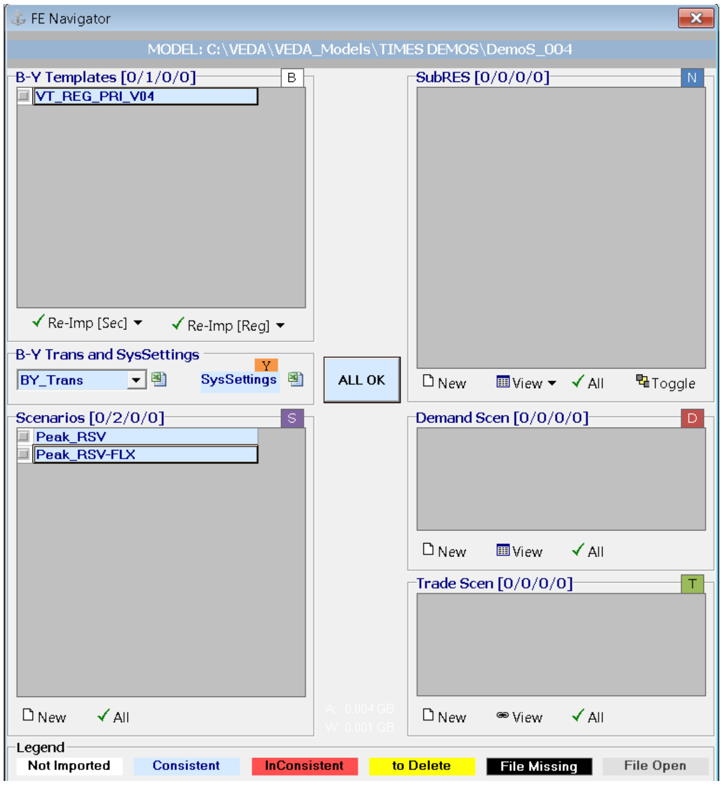


Figure 65. The files included in DemoS_004

3.4.1 SysSettings file

3.4.1.1 Region-Time Slices

The ~TimeSlices table is used to create four time slices (Figure 66) and replace the previous single ANNUAL time slice. There are four time slices combining two seasons (W- Winter and S-Summer) and two intraday periods or day-night periods (D- Day and N- Night).

~TimeSlices		
Season	Weekly	DayNite
S		D
W		N

Figure 66. New time slices definition in the SysSettings file

3.4.1.2 Interpol_Extrapol_Defaults

A table is added for setting the default interpolation/extrapolation rules (Figure 67). A transformation table used to update pre-existing data (~TFM_UPD) in a rule-based manner, it sets the default interpolation/extrapolation rule, indicated by the 0 in the Year column, for the attribute defined in the Attribute column and all the processes defined in the model. In this case, this is the same interpolation/extrapolation rule used for each of the supply processes (see Figure 30) in the B-Y Template. It is now moved into the SysSettings file and applied to the activity bound (ACT_BND) of all processes at once.

~TFM_UPD					
TimeSlice	LimType	Attribute	Year	AllRegions	Pset_PN
	UP	ACT_BND	0	5	

Figure 67. Default table for interpolation/extrapolation rules in the SysSettings file

3.4.1.3 Constants

The existing transformation table is also used to insert new constants in the model: fractions of year for the new time slices (YRFR) replace the single ANNUAL time slice (100%) as declared in the previous steps (Figure 68). The timeslice name is identified in the first column (TimeSlice), while their fractions (for the attribute called YRFR) over one year are declared for AllRegions as for the other constants of the model. The fraction values, as with any other input in the model, are the user's responsibility. In this case, it is important that they sum to 100%.

~TFM_INS					
TimeSlice	LimType	Attribute	AllRegions	REG1	Cset_CN
		G_DYEAR	2005		
		Discount	0.05		
SD		YRFR	0.25		
SN		YRFR	0.23		
WD		YRFR	0.25		
WN		YRFR	0.27		
		COM_IE		0.90	ELC

Figure 68. New time slice declarations in the SysSettings file

3.4.2 B-Y Templates

3.4.2.1 Con_ELC

A new attribute is declared for all existing and new processes representing power plants (Figure 69):

- Their contribution to peak (Peak), i.e., the fraction of a process's capacity that is considered to be secure and thus will most likely be available to contribute to the peak (and reserve capacity) load in the highest demand time-slice of a year for a commodity (electricity or heat only). In this case, the capacity contribution of all thermal and nuclear power plants is 100%, while the capacity contribution of the renewable power plant is 50%. Indeed, many types of supply processes can be regarded as predictably available with their entire capacity contributing during the peak and thus have a peak coefficient equal to 1 (100%), whereas others (such as wind turbines or solar plants) are attributed a peak coefficient less than 1 (100%), since they are on average only fractionally available at peak. (E.g., a wind turbine typically has a peak coefficient of 0.25 or 0.3 maximum).

Another important change to mention is the start year of one new process (ELCTNOIL00) that can be installed from the 2005 base year to cover the additional capacity needed for the reserve equation (5%), as defined in the scenario files.

		~FI_T							
TechName	**Comm-IN**	**Comm-OUT**	**STOCK**	**STOCK~2030**	**(...)**	**START**	**ENV_ACT**	**CAP2ACT**	**Peak**
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Retirement Capacity			Emission Coefficient	Capacity to Activity Factor	% contribution to PEAK
*Units			GW	GW			kt	PJ/GW	
ELCTECOA00	ELCCOA	ELC	137					31.536	1.00
		ELCCO2					260		
ELCTEGAS00	ELCGAS	ELC	104					31.536	1.00
		ELCCO2					114		
ELCTEOIL00	ELCOIL	ELC	11					31.536	1.00
		ELCCO2					306		
ELCRERNW00	ELCRNW	ELC	88	88				31.536	0.50
ELCTENUC00	ELCNUC	ELC	125	125				31.536	1.00
ELCTNCOA00	ELCCOA	ELC				2006		31.536	1.00
		ELCCO2					238		
ELCTNGAS00	ELCGAS	ELC				2006		31.536	1.00
		ELCCO2					108		
ELCTNOIL00	ELCOIL	ELC				2005		31.536	1.00
		ELCCO2					255		

Figure 69. Peak contribution for different types of power plants

Additional information is required to complete the declaration of the electricity commodity and processes in their respective tables (Figure 70 and Figure 71). Along with the new time slices, it is possible to specify the tracking level of the electricity commodity (ELC) in the **CTSLvl** column: DAYNITE. (When not specified, as in the previous step, the default is ANNUAL.) **PeakTS (**peak time slice monitoring**)** directs TIMES to generate the peak equation for the specified time slices. It is possible to declare any of the time slices defined in the SysSettings file, or ANNUAL (the default) to generate the peaking equation for all time slices. Since it is left blank here, the peak equation will be generated in all time slices once it has been requested using COM_Peak (see Section 3.4.3.1). Finally, it is important that the user enter ELC in the **Ctype** column when declaring an electricity commodity that may be produced by combined heat and power (CHP) plants, as this commodity will be in DemoS_009.

For the electricity processes, the process table is used to define the time slice level of operation in the **Tslvl** column (Figure 71). For example, the coal-fired and the nuclear power plants are defined at the SEASON time slice level, meaning that their operational level does not vary across DAYNITE time slices. (When not specified, the default is based on the Sets declaration: DAYNITE (for ELE), SEASON (for CHP and HPL) ANNUAL (for all others).)

~FI_Comm								
Csets	**Region**	**CommName**	**CommDesc**	**Unit**	**LimType**	**CTSLvl**	**PeakTS**	**Ctype**
*Commodity Set Membership	Region Name	Commodity Name	Commodity Description	Unit	Sense of the Balance EQN.	Timeslice Level	Peak Monitoring	Electricity Indicator
NRG		ELC	Electricity	PJ		DAYNITE		ELC
ENV		ELCCO2	Electricity Plants Carbon dioxide	kt				

Figure 70. Declaration of time slice level for electricity commodity

~FI_Process								
Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Commodity Group	Vintage Tracking
*								
ELE		ELCTECOA00	Power Plants Existing00 - Solid Fuels	PJ	GW	SEASON		
		ELCTEGAS00	Power Plants Existing00 - Natural Gas	PJ	GW			
		ELCTEOIL00	Power Plants Existing00 - Crude Oil	PJ	GW			
		ELCRERNW00	Power Plants Existing00 - Renewable Energies	PJ	GW			
		ELCTENUC00	Power Plants Existing00 - Nuclear Energy	PJ	GW	SEASON		
		ELCTNCOA00	Power Plants New00 - Solid Fuels	PJ	GW	SEASON		
		ELCTNGAS00	Power Plants New00 - Natural Gas	PJ	GW			
		ELCTNOIL00	Power Plants New00 - Crude Oil	PJ	GW			

Figure 71. Declaration of time slice operational level for processes

3.4.2.2 Pri_COA/GAS/OIL

These sheets were all modified back to remove the interpolation/extrapolation rules: the flag to activate an interpolation/extrapolation rule (additional rows with a "0" as the Year) and the rule code in the attribute column.

3.4.2.3 Demands

A table is added to define the load curve of the demand for electricity (TPSELC) in the base year, which will also apply for future years (Figure 72). The attribute (COM_FR) is introduced to declare the fraction of the electricity demand occurring in each time slice.

		~FI_T	
Attribute	CommName	Timeslices	2005
*	Demand Commodity Name		
*Units			
COM_FR	TPSELC	SD	0.30
COM_FR	TPSELC	SN	0.20
COM_FR	TPSELC	WD	0.27
COM_FR	TPSELC	WN	0.23

Figure 72. Definition of load curve for the electricity demand

The TPSELC commodity is the demand commodity produced by a demand technology (end-use technology) called DTPSELC (Figure 73) and defined in the sheet DemTechs_ELC. This technology takes as input the ELC commodity that will be consumed by timeslice as defined by the COM_FR attribute for TPSELC.

		~FI_T						
TechName	Comm-IN	Comm-OUT	STOCK	EFF	AFA	INVCOST	FIXOM	LIFE
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Lifetime
*Units			PJa			M€2005/PJ	M€2005/PJa	Years
DTPSELC	ELC	TPSELC		1.00	0.95	10	0.20	20

Figure 73. Demand technology producing DTPSELC

3.4.3 Scenario files

3.4.3.1 Scen_Peak_RSV and Scen_Peak_RSV-FLX

Two scenario files are created to insert new information in the RES that can be retained or not in the configuration of the model at the time of solving the model (see Section 2.4.5). A transformation table **~TFM_INS** is used to declare new attributes (Figure 74):

- COM_Peak - Specify that the peaking equation will be generated for the ELC commodity.
- COM_PKRSV - Declare the capacity fraction (%) that is required for the peak reserve. This is the option used in the first scenario file (Peak_RSV).
- COM_PKFLX - Declare the fraction (%) by which the actual peak demand exceeds the average calculated demand, by time slice. This is the option used in the second scenario file (Peak_RSV- FLX) for the Summer-Day time slice (SD), although in practice COM_PKFLX is typically used alongside COM_PKRSV.

The TIMES peak equation allows the user to require that the total capacity of all processes producing a commodity at each time period and in each region exceed, by a certain percentage, the average demand in the time-slice when the highest demand occurs. This peak reserve factor (COM_PKRSV) insures against several contingencies, such as possible commodity shortfall due to uncertainty regarding its supply (e.g. water availability in a reservoir), unplanned equipment down time, and random peak demand that exceeds the average demand during the time-slice when the peak occurs. This constraint is therefore akin to a safety margin to protect against random events not explicitly represented in the model. Optionally, COM_PKFLX can be used to reflect the fact that the actual system peak demand is greater than the average demand in the model's peak slice, allowing COM_PKRSV to represent a more typical utility reserve margin.

~TFM_INS

TimeSlice	LimType	Attribute	Year	Attrib_Cond	Val_Cond	AllRegions	REG1	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
		COM_PEAK					1.00							ELC	
		COM_PKRSV	2005				5%							ELC	
		COM_PKRSV	2020				20%							ELC	

~TFM_INS

TimeSlice	LimType	Attribute	Year	Attrib_Cond	Val_Cond	AllRegions	REG1	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
		COM_PEAK					1.00							ELC	
		COM_PKRSV												ELC	
SD		COM_PKFLX	2005				5%							ELC	
SD		COM_PKFLX	2020				20%							ELC	

Figure 74. Declaration of the peak reserve in a scenario file

3.4.4 Results

Three cases are solved with this step model, with a different selection of scenario files (Figure 75): the DemoS_004 case is solved using only the two components (BASE, SysSettings), while the DemoS_004a case is solved adding one scenario file (Peak_RSV), and the DemoS_004b case is solved adding the other scenario file (Peak_RSV-FLX). The different cases can be loaded in the FE Case Manager using the **Select** menu. Choosing the **Single** option will solve each case individually, while choosing the **Batch** option will launch multiple cases simultaneously (i.e., the cases will be launched automatically by VEDA-FE one after the other).

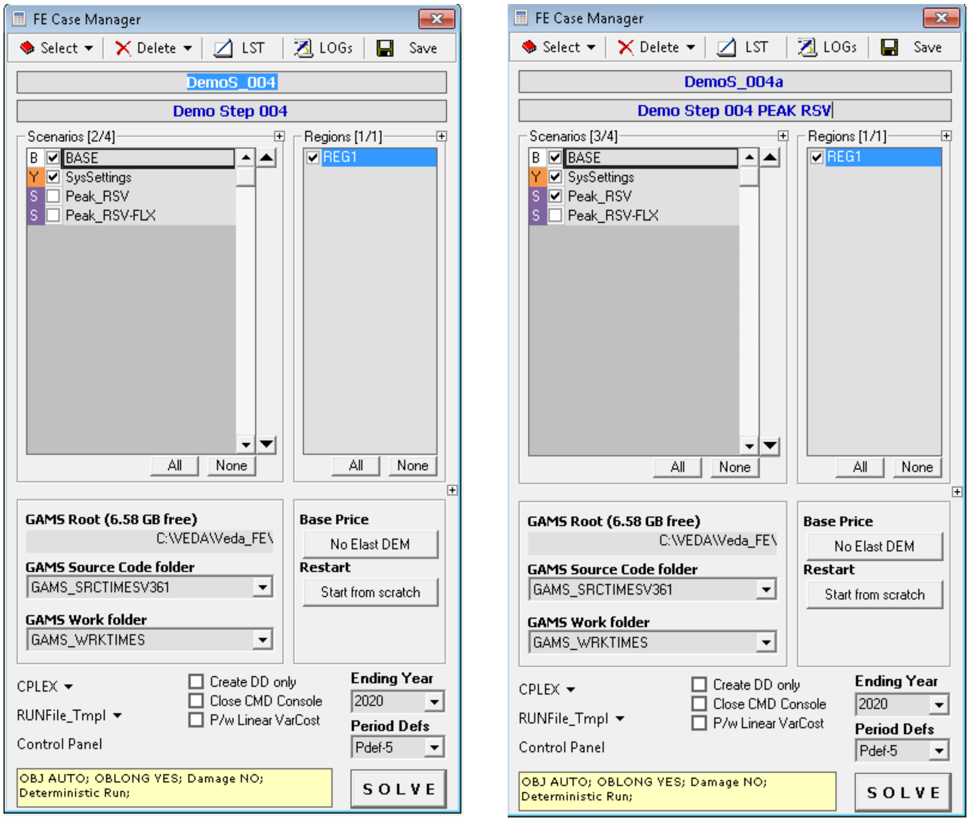


Figure 75. Solving different cases using different scenario files

The impacts of the improvements made in the electricity sector on the electricity generating capacity are shown in Figure 76, namely.

- The effect of adding new time slices and of specifying the seasonal operational level for the coal-fired power plant in DemoS_004, compared with DemoS_003: there is a switch from coal-fired generation to natural gas-fired generation due to its greater flexibility (time slice level DAYNITE for gas, as opposed to SEASON for coal) to satisfy the electricity demand. The additional natural gas supply is coming from import sources.
- The effect of declaring a peak reserve factor on the total capacity in DemoS_004a, compared with DemoS_004: there is additional capacity required that is coming from oil-fired power plants as new power plants are available from 2005. The total capacity in DemoS_004a is increasing from 507 GW in 2005 to 659 GW in 2020 (compared with 466 GW to 542 GW without the peak reserve requirement).
- There is no effect on the generating capacity in DemoS_004b, compared with DemoS_004a.

The electricity price varies across years and time slices (Figure 77).

ELC plants capacity and new capacity

Original Units: GW Active Unit GW Data values filter:

Attribute | *Process* | *Vintage*

Region	~-Scenario-~	ProcessSet	2005	2006	2010	2015	2020
REG1	DemoS_003	Coal Power Plants	137	158	197	246	296
		Gas Power Plants	104	98	78	52	26
		Nuclear Power Plants	125	125	125	125	125
		Oil Power Plants	11	11	10	8	6
		Renewable Power Plants	88	88	88	88	88
		Total	466	482	498	519	542
	DemoS_004	Coal Power Plants	137	158	184	189	194
		Gas Power Plants	104	98	90	108	128
		Nuclear Power Plants	125	125	125	125	125
		Oil Power Plants	11	11	10	8	6
		Renewable Power Plants	88	88	88	88	88
		Total	466	482	498	519	542
	DemoS_004a	Coal Power Plants	137	158	184	189	194
		Gas Power Plants	104	98	90	108	128
		Nuclear Power Plants	125	125	125	125	125
		Oil Power Plants	53	52	66	93	123
		Renewable Power Plants	88	88	88	88	88
		Total	507	523	554	605	659

Figure 76. Results – electricity generation capacity in DemoS_004

Energy Prices

Original Units: Euro per GJ Active Unit Euro per GJ Data values filter:

Attribute | CommoditySet

~-Scenario-~	Region	-Commodity-	TimeSlice	2005	2006	2010	2015	2020
DemoS_004	REG1	Electricity	SD	35.78	12.97	14.05	14.05	14.05
			SN	35.78	12.97	11.79	11.79	11.79
			WD	35.78	12.97	14.24	14.24	14.24
			WN	35.78	12.97	11.79	11.79	11.79
DemoS_004a	REG1	Electricity	SD	29.85	12.97	12.85	12.85	12.85
			SN	29.85	12.97	10.59	10.59	10.59
			WD	29.85	12.97	13.05	13.05	13.05
			WN	29.85	12.97	10.59	10.59	10.59

Figure 77. Results – electricity price by time slice in DemoS_004

Other interesting results to show are related to the peak contribution specifically (Figure 78). The peak equation expresses that the available capacity must exceed demand for the electricity (ELC) commodity in any time slice by a certain margin, so the dual value of the peak equation describes the premium consumers have to pay in addition to the commodity price (dual value of EQ_COMBAL) during the peak time slice (SD in this case) to ensure adequate system capacity. The peak marginal is similar, though not identical, when using COM_PKRSV and COM_PKFLX, owing to the differences in how they are applied in the TIMES equations.

`Peak Equation

Original Units: **Active Unit** **Data values filter:**

~Scenario~	Region	Attribute	Commodity	TimeSlice	2005	2006	2010	2015	2020
DemoS_004a	REG1	EQ_Peak	Electricity	SD	-2,848.96	-2,712.32	-2,392.62	-1,975.97	-1,594.03
				SN	-1,828.90	-1,695.27	-1,368.66	-942.87	-546.86
				WD	-2,536.59	-2,397.01	-2,065.00	-1,632.25	-1,233.34
				WN	-2,095.53	-1,937.20	-1,549.19	-1,043.34	-572.45
		EQ_PeakM	Electricity	SD	12.23		4.48	4.69	4.89
DemoS_004b	REG1	EQ_Peak	Electricity	SD	-2,991.40	-2,875.05	-2,631.88	-2,272.36	-1,912.84
				SN	-1,816.11	-1,670.66	-1,286.43	-738.90	-172.20
				WD	-2,522.71	-2,370.29	-1,975.73	-1,410.80	-826.57
				WN	-2,080.44	-1,908.16	-1,452.15	-802.63	-130.31
		EQ_PeakM	Electricity	SD	11.65		4.07	4.07	4.07

Figure 78. Results – Dual values of the peak equations in DemoS_004

Objective-Function = 3,187,361 M euros (see the _SysCost table in VEDA-BE). This cost is only slightly higher with the peak reserve requirement and the additional investments in generating capacity: 3,211,296 M euros.

3.5 DemoS_005 - 2-region model with endogenous trade: compact approach

Description. At the fifth step, the model evolves from being a single region model to become a compact multi-regional model (2 or more regions in the same set of B-Y Templates). This approach is relevant when all the model regions are under the control of a single individual.

Objective. The objective is to create the multi-regional model framework typical to larger or more complex models, namely the trade matrix that allows the modelling of energy trade movements (uni-directional or bi-directional trade between two regions). Another objective is to demonstrate how to limit emissions from a sector in a particular region or from the entire energy system of all regions through emission bounds or user constraints. Scenario variants illustrate the impact of a cap on CO_2 emissions from the electricity sector only and of a cross-region user constraint on the total CO_2 emissions from the transport and electricity sectors.

Attributes introduced: COM_BNDNET UC_RHSRTS UC_COMNET	Files updated SysSettings VT_REG_PRI_v05 Files created Scen_TRADE_PARAM Scen_ELC_CO2_BOUND Scen_UC_CO2BND Files removed Scen_Peak_RSV-FLX

Files. The fifth step model is built:

1. by modifying the SysSettings file to add one region;
2. by modifying the B-Y Template (VT_REG_PRI_V05) to disaggregate the energy balance between two regions and to regionalize some process attributes;

3. by creating trade files to capture the trade movements between the two regions;
4. by creating more scenario files to limit GHG emissions (Figure 79).

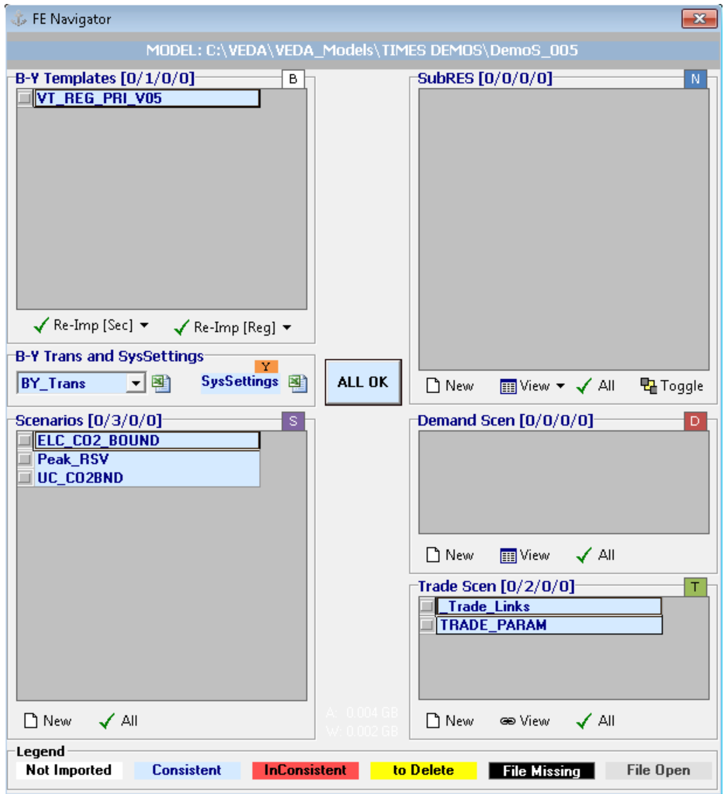


Figure 79. The files included in DemoS_005

3.5.1 SysSettings file

3.5.1.1 Region-Time Slices

The ~BookRegions_Map table is used to create one additional region: REG2 (Figure 80) in the same workbook (REG).

~BookRegions_Map	
BookName	Region
REG	REG1
	REG2

Figure 80. New region definition in the SysSettings file

3.5.2 B-Y Templates

3.5.2.1 EnergyBalance, EB1, EB2

The energy balance is disaggregated between two regions (Figure 81) using shares on production, conversion, and final consumption of various energy commodities: REG1 becomes producer and consumer of solid fuels (100%), crude oil (30%) and renewable energies (100%), while REG2 becomes producer and consumer of natural gas (100%), crude oil (70%), and nuclear energy (100%). The same portion of the energy balance as in the fourth step is used in this fifth step model.

		COA	GAS	OIL	NUC	RNW	SLU	HET	ELC	
	REG1	Solid Fuels	Natural Gas	Crude Oil	Nuclear Energy	Renewable Energies	Industrial Wastes	Derived Heat	Electricity	Total
	PRIMARY									
MIN	Domestic Supply	8098	0	1614	0	5027	0	0	0	14739
IMP	Imports	6463	0	11988	0	113	0	0	584	19148
EXP	Exports	-1147	0	-4449	0	-72	0	0	-563	-6232
TPS	Total Primary Supply	13414	0	9152	0	5067	0	0	20	27654
	CONVERSION									
ESC	Energy Sector Consumption	-58	0	-555	0	-4	-1	0	0	-617
ELC	Electricity Plants	-9598	0	-367	0	-1256	-16	869	5791	-4578
HPL	Heat Plants	-161	0	-15	0	-140	-1	329	0	12
REF	Petroleum Refineries	0	0	-9521	0	0	0	0	0	-9521
	Total Conversion	-9817	0	-10458	0	-1400	-18	1198	5791	-14704
	FINAL									
RSD	Residential	357	0	687	0	1294	0	433	1436	4206
COM	Commercial	57	0	256	0	67	1	127	1264	1772
IND	Industry	1897	0	605	0	722	59	317	2044	5643
AGR	Agriculture	44	0	239	0	63	0	8	10	364
TRA	Transport	1	0	4455	0	131	0	0	133	4720
OTH	Other	1189	0	118	0	1390	0	314	325	3336
NEN	Non Energy	52	0	1222	0	0	0	0	0	1274
BNK	Bunkers	0	0	633	0	0	0	0	0	633
TFC	Total Final Consumption	3597	0	8215	0	3667	59	1198	5211	21948

		COA	GAS	OIL	NUC	RNW	SLU	HET	ELC	
	REG2	Solid Fuels	Natural Gas	Crude Oil	Nuclear Energy	Renewable Energies	Industrial Wastes	Derived Heat	Electricity	Total
	PRIMARY									
MIN	Domestic Supply	0	7899	3765	10775	0	0	0	0	22440
IMP	Imports	0	13292	27972	0	0	0	0	584	41848
EXP	Exports	0	-2516	-10381	0	0	0	0	-563	-13461
TPS	Total Primary Supply	0	18675	21355	10775	0	0	0	20	50826
	CONVERSION									
ESC	Energy Sector Consumption	0	-793	-1294	0	0	-1	0	0	-2088
ELC	Electricity Plants	0	-5636	-857	-10775	0	-16	869	5791	-10625
HPL	Heat Plants	0	-301	-35	0	0	-1	329	0	-7
REF	Petroleum Refineries	0	0	-22216	0	0	0	0	0	-22216
	Total Conversion	0	-6730	-24402	-10775	0	-18	1198	5791	-34936
	FINAL									
RSD	Residential	0	5160	1603	0	0	0	433	1436	8631
COM	Commercial	0	1752	598	0	0	1	127	1264	3742
IND	Industry	0	4437	1411	0	0	59	317	2044	8268
AGR	Agriculture	0	201	558	0	0	0	8	10	777
TRA	Transport	0	21	10396	0	0	0	0	133	10550
OTH	Other	0	0	275	0	0	0	314	325	913
NEN	Non Energy	0	634	2851	0	0	0	0	0	3485
BNK	Bunkers	0	0	1478	0	0	0	0	0	1478
TFC	Total Final Consumption	0	12205	19169	0	0	59	1198	5211	37843

Figure 81. Energy balance at start year 2005 for REG1 & REG 2–Covered in DemoS_005

3.5.2.2 Pri_COA/GAS/OIL

These sheets are updated to include two regions and to regionalize some process attributes. There are several ways of accounting for the regionalization of some attributes. For instance, it is possible to insert a **Region** column on the left side of any **~FI_T** table and to indicate in which region(s) the process is available (Figure 82). A process can be available in only one region (e.g. MINGAS* and IMPGAS1) or in several regions (EXPGAS1). In this later case, different rows can be inserted to declare different values for some of the attributes (ACT_BND of EXPGAS1);

the values that remain on the initial row will apply to all regions (COST of EXPGAS1). The additional rows approach is mainly used when all attributes of a process vary across regions.

In the process table (~FI_ Process), the region where each process is available can be specified (Figure 83): MINGAS* and IMPGAS1 processes exist only in REG2, while the EXPGAS1 process exists in both regions (by default, when the **Region** column is empty, it applies to all regions). Comma-separated entries are also allowed, for instance, when a process exists in more than one region but not in all regions.

			~FI_T			
Region	**TechName**	**Comm-IN**	**Comm-OUT**	**CUM**	**COST**	**ACT_BND**
Region Name	*Technology Name	Input Commodity	Output Commodity	Reserves Cumulative Value	Cost	Annual Production Bound
	*Units			PJ	M€2005/PJ	PJ
REG2	MINGAS1		GAS	15000	3.60	3950
REG2	MINGAS2		GAS	20000	4.14	3950
REG2	MINGAS3		GAS	30000	5.40	
REG2	IMPGAS1		GAS		4.50	
	EXPGAS1	GAS			4.50	
REG1						2516
REG2						2516

Figure 82. Regionalization of process attributes using additional rows

~FI_Process								
Sets	**Region**	**TechName**	**TechDesc**	**Tact**	**Tcap**	**Tslvl**	**PrimaryCG**	**Vintage**
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
*								
MIN	REG2	MINGAS1	Domestic Supply of Natural Gas Step 1	PJ				
	REG2	MINGAS2	Domestic Supply of Natural Gas Step 2	PJ				
	REG2	MINGAS3	Domestic Supply of Natural Gas Step 3	PJ				
IMP	REG2	IMPGAS1	Import of Natural Gas Step 1	PJ				
EXP		EXPGAS1	Export of Natural Gas Step 1	PJ				

Figure 83. Region specification in the default process table

3.5.2.3 Con_ELC

This sheet is also updated to include two regions and to regionalize some process attributes. However, a different approach is used (Figure 84): columns are inserted (duplicated) only for those attributes that vary across regions: the STOCK attribute in this example. As for the year, the regions are identified using the " ~ " command after the attribute. The additional columns approach is mainly used when only few attributes of a process vary across regions.

The column approach is also used in the following sheets, namely for the STOCK attribute: Sector_Fuels, DemTechs_TPS, DemTechs_ELC, DemTechs_RSD and DemTechs_TRA. The row approach is used in the Demand sheet.

3.5.3 Trade files

Two trade files are created to model the energy trade movements between the two regions.

		~FI_T						
TechName	Comm-IN	Comm-OUT	STOCK~REG1	STOCK~REG2	STOCK~REG1~2030	STOCK~REG2~2030	EFF	(...)
*Technology Name	Input Commodity	Output Commodity	Existing Installed	Existing Installed	Retirement Capacity	Retirement Capacity	Efficiency	
*Units			GW	GW	GW	GW		
ELCTECOA00	ELCCOA	ELC	137	0			0.38	
		ELCCO2						
ELCTEGAS00	ELCGAS	ELC	0	104			0.49	
		ELCCO2						
ELCTEOIL00	ELCOIL	ELC	3	8			0.25	
		ELCCO2						
ELCRERNW00	ELCRNW	ELC	88	0	88	0	1.00	
ELCTENUC00	ELCNUC	ELC	0	125	0	125	0.33	
ELCTNCOA00	ELCCOA	ELC					0.42	
		ELCCO2						
ELCTNGAS00	ELCGAS	ELC					0.52	
		ELCCO2						
ELCTNOIL00	ELCOIL	ELC					0.30	
		ELCCO2						

Figure 84. Regionalization of process attributes using additional columns

3.5.3.1 Scen_Trade_Links

The ~ TradeLinks tables are used to declare the traded commodities and their links between regions (Figure 85): either bilateral links between regions (e.g. ELC trade between REG 1 (importer/exporter) and REG2 (importer/exporter) or unilateral links between regions (e.g. GAS trade between REG 1 (importer) and REG2 (exporter). For each link declared (1=active links), VEDA-FE will automatically create an IRE (inter-regional trade) process to which attributes may then be associated (e.g., bounds, investment costs, etc.). The naming convention for IRE processes is:

- Bilateral trade: TB_<fuel name>_<exporter region>_<importer region>_<01> (e.g. TB_ELC_REG1_REG2_01)
- Unilateral trade: TU_<fuel name>_<exporter region>_<importer region>_<01> (e.g. TU_GAS_REG2_REG1_01)

~TradeLinks		
ELC	REG1	REG2
REG1		1
REG2	1	

~TradeLinks		
GAS	REG1	REG2
REG1		
REG2	1	

Figure 85. Examples of trade matrix for bilateral and unilateral links

3.5.3.2 Scen_Trade_Param

In this file, a transformation table **~TFM_INS** is used to insert new attributes for trade processes (Figure 86), for example: an investment cost (INVCOST) for all unilateral trade processes (TU_*). Trade processes are created automatically after the user declares unilateral or bilateral links between regions in the **_Trade_Links** file.

~TFM_INS																
TimeSlice	LimType	Attribute	Year	Attrib_Cond	Val_Cond	AllRegions	REG1	REG2	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
		INVCOST				10				TU_*						

Figure 86. Declaration of attributes for IRE processes

3.5.4 Scenario files

Two more scenario files are created to insert new information in the RES that can be retained or not in the configuration of the model at the time of solving the model. Of the previous scenario files, only the Scen_Peak_RSV file is retained for further analysis.

3.5.4.1 Scen_ELC_CO2_Bound

This file is used to introduce a bound (limit) on the CO2 emissions from the power sector in REG1. A transformation table ~TFM_INS is used (Figure 87) to declare an upper bound on annual emissions (Attribute = COM_BNDNET; LimType = UP), on the CO2 emissions from the electricity sector only (ELCCO2) in REG1. In this example the upper bound is calculated as a percentage reduction target from the power sector CO2 emissions in a reference scenario for 2010 (10% = 993,548 kt) and 2020 (20% = 1,017,340 kt). It is necessary to run the step model without any limit on emissions first to get the reference emission trajectory (run DemoS_005) and to calculate the bounds as a reduction target from the reference emissions. An interpolation rule is used with the "0" flag in the Year column and the interpolation/extrapolation option in the region column where the bounds are declared. The code 5 means full interpolation and forward extrapolation.

~TFM_INS

TimeSlice	LimType	Attribute	Year	AllRegions	REG1	REG2	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
	UP	COM_BNDNET	2010		993548								ELCCO2	
	UP	COM_BNDNET	2020		1017340								ELCCO2	
	UP	COM_BNDNET	0		5								ELCCO2	

Figure 87. Declaration of emission bounds for the power sector

3.5.4.2 Scen_UCCO2_BND – user constraint

This file shows another way used to introduce bounds (limits) on the CO2 emissions from both the power and the transportation sectors in each region (REG1 and REG2). The idea is to build a user constraint (Figure 88) that specifies the maximum amount of emissions in a specific year for the sum of TRACO2 and ELCCO2 emission commodities.

These upper bounds (or limits) are again calculated as a percentage reduction target from the CO2 emissions (sum in kt) of the power and the transportation sector in a reference scenario for 2010 (10%) and 2020 (20%). It is necessary to run the step model without any limit on emissions first to get the reference emission trajectory (run DemoS_005) and to calculate the bounds as a reduction target from the reference emissions.

~UC_Sets: R_E: AllRegions
~UC_Sets: T_E:

								~UC_T:UC_RHSRTS					
UC_N	Pset_Set	Pset_PN	Pset_CI	Pset_CO	Cset_CN	Attribute	Year	LimType	UC_COMNET	REG1	REG2	UC_RHSRTS~0	UC_Desc
AU_CO2_BND					TRACO2,ELCCO2		2010	UP	1	1142284	1142284	5	CO2 Bound Constraint
					TRACO2,ELCCO2		2020	UP	1	1227958	1053944		

Figure 88. Declaration of emission bounds using a user constraint

The UC scenario template is set up as described in Section 2.3.8. The sets declarations above the table indicate:

- ~UC_Sets: R_E: AllRegions: The constraints are to be applied to all regions in the model, individually (E=each). That is, the bounds imposed for REG1 and REG2 are separate, and there is no emissions trading between regions.
- ~UC_Sets: T_E: The constraints are imposed to each time period individually. There is no banking or borrowing between periods.

The table level declaration following the table tag (~UC_T:**UC_RHSRTS**) indicates that any column without an index will be interpreted as the right hand side of the constraint, in this case, the indicated bounds in REG1 and REG2 in the given years. This right hand side bounds 1 times the net production (UC_COMNET) of the sum of TRACO2 and ELCCO2. The interpolation/extrapolation option 5 indicates full interpolation and forward extrapolation.

3.5.5 Results

Three cases are solved with this step model, with a different selection of scenario files: the DemoS_005 case is solved without any limit on CO2 emissions and using only the three main components (BASE, TRADE_PARAM, SysSettings), while the DemoS_005a case is solved adding one scenario file (ELC_CO2_BOUND) to put a limit on CO2 emissions from the REG1 power sector, and the DemoS_005b case is solved adding the other scenario file (UC_CO2_BND) to put a limit on both the power and the transportation sectors in both regions.

A first sample of results shows the different configuration of the energy supply systems in the two regions (Figure 89). As mentioned earlier, the REG1 becomes the main provider of solid fuels, renewable energies and some crude oil (from both domestic production and imports). REG1 is also getting electricity from REG2. REG2 becomes the main provider of natural gas, nuclear energy and some crude oil (from both domestic production and imports).

Fuel Supply

Original Units: PJ **Active Unit** PJ **Data values filter:**

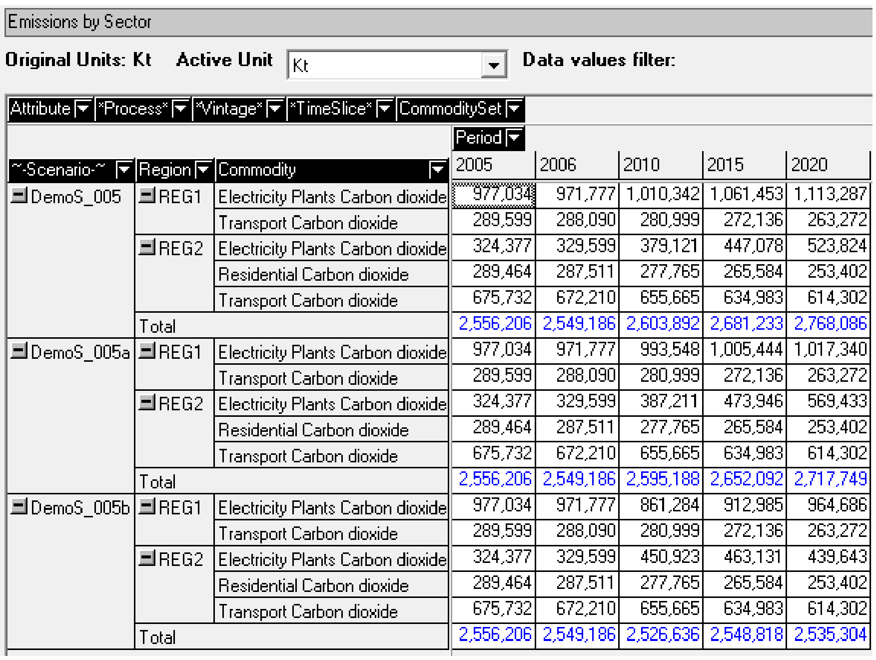
Attribute | *Process* | *Vintage* | *TimeSlice* | ProcessSet

~Scenario~	Region	Commodity	Period: 2005	2006	2010	2015	2020
DemoS_004a	REG1	Crude Oil	30,702	29,559	29,014	28,332	28,332
		Natural Gas	13,312	12,952	12,116	12,552	13,470
		Nuclear Energy	10,775	10,775	10,775	10,775	10,775
		Renewable Energies	1,256	1,256	1,256	1,256	1,256
		Solid Fuels	14,342	15,650	17,203	17,376	17,564
		Total	70,387	70,191	70,363	70,290	71,397
DemoS_005	REG1	Crude Oil	9,155	8,881	8,772	8,636	8,500
		Electricity	774	837	866	904	955
		Renewable Energies	1,256	1,256	1,256	1,256	1,256
		Solid Fuels	14,342	14,481	14,868	15,380	15,899
		Total	25,527	25,455	25,762	26,175	26,609
	REG2	Crude Oil	21,027	20,723	20,469	20,150	19,832
		Natural Gas	13,118	13,517	14,226	15,220	16,371
		Nuclear Energy	10,775	10,775	10,775	10,775	10,775
		Total	44,920	45,015	45,469	46,145	46,978

Figure 89. Results – fuel supply options for both regions in DemoS_005

A second sample of results shows the evolution of the emissions in the different sectors of the two regions (Figure 90):

- Emissions from the power and the transportation sectors as projected in the DemoS_005 case were used to compute the emissions limits in the other two cases.
- A limit on the CO2 from the power sector in REG1 (DemoS_005a) leads to a lower electricity production from solid fuels, and an emission increase in REG2, which produces more electricity from natural gas to supply REG1 (Figure 91).
- With a limit on the CO2 from both the power and the transportation sector in REG1 and in REG2 (DemoS_005b), all the emission reductions are coming from the power sector in both regions. Emissions from the transportation sector are not affected compared with the reference case (DemoS_005) meaning that the power sector of both regions could provide enough reduction options at a lower cost to meet the target. Because there is no trading in emissions between regions, REG2 must cut back on its electricity generation from natural gas, and it begins importing natural gas-fired electricity from REG1, which in turn imports natural gas from REG2 (Figure 91).


Emissions by Sector

Original Units: Kt   Active Unit Kt   Data values filter:

Attribute | *Process* | *Vintage* | *TimeSlice* | CommoditySet

~-Scenario-~	Region	Commodity	2005	2006	2010	2015	2020
DemoS_005	REG1	Electricity Plants Carbon dioxide	977,034	971,777	1,010,342	1,061,453	1,113,287
		Transport Carbon dioxide	289,599	288,090	280,999	272,136	263,272
	REG2	Electricity Plants Carbon dioxide	324,377	329,599	379,121	447,078	523,824
		Residential Carbon dioxide	289,464	287,511	277,765	265,584	253,402
		Transport Carbon dioxide	675,732	672,210	655,665	634,983	614,302
	Total		2,556,206	2,549,186	2,603,892	2,681,233	2,768,086
DemoS_005a	REG1	Electricity Plants Carbon dioxide	977,034	971,777	993,548	1,005,444	1,017,340
		Transport Carbon dioxide	289,599	288,090	280,999	272,136	263,272
	REG2	Electricity Plants Carbon dioxide	324,377	329,599	387,211	473,946	569,433
		Residential Carbon dioxide	289,464	287,511	277,765	265,584	253,402
		Transport Carbon dioxide	675,732	672,210	655,665	634,983	614,302
	Total		2,556,206	2,549,186	2,595,188	2,652,092	2,717,749
DemoS_005b	REG1	Electricity Plants Carbon dioxide	977,034	971,777	861,284	912,985	964,686
		Transport Carbon dioxide	289,599	288,090	280,999	272,136	263,272
	REG2	Electricity Plants Carbon dioxide	324,377	329,599	450,923	463,131	439,643
		Residential Carbon dioxide	289,464	287,511	277,765	265,584	253,402
		Transport Carbon dioxide	675,732	672,210	655,665	634,983	614,302
	Total		2,556,206	2,549,186	2,526,636	2,548,818	2,535,304

Figure 90. Results – emissions by sector and by region in DemoS_005

Finally, the marginal price of CO2 (i.e. the price to pay in euros to reduce the last ton of CO2 to meet the reduction targets) in both scenarios with limits on emissions is particularly relevant and represents the level of tax that would be necessary to achieve the reduction targets that are prescribed in the scenario files (Figure 92).

Electricity endogenous trade

Original Units: PJ **Active Unit** PJ **Data values filter:**

Vintage | *TimeSlice*

~Scenario~	Commodity	Region	Attribute	Process	2005	2006	2010	2015	2020
DemoS_005	ELC	REG1	VAR_FOut	TB_ELC_REG1_REG2_01	774	837	866	904	955
		REG2	VAR_FIn	TB_ELC_REG1_REG2_01	774	837	866	904	955
DemoS_005a	ELC	REG1	VAR_FOut	TB_ELC_REG1_REG2_01	774	837	937	1,140	1,358
		REG2	VAR_FIn	TB_ELC_REG1_REG2_01	774	837	937	1,140	1,358
DemoS_005b	ELC	REG1	VAR_FIn	TB_ELC_REG1_REG2_01					208
			VAR_FOut	TB_ELC_REG1_REG2_01	774	837	1,494	1,042	418
		REG2	VAR_FIn	TB_ELC_REG1_REG2_01	774	837	1,494	1,042	418
			VAR_FOut	TB_ELC_REG1_REG2_01					208

Endogenous gas trade

Original Units: PJ **Active Unit** PJ **Data values filter:**

Vintage | TimeSlice

~Scenario~	Commodity	Region	Attribute	Process	2015	2020
DemoS_005b	GAS	REG1	VAR_FOut	TU_GAS_REG2_REG1_01	1,714	4,907
		REG2	VAR_FIn	TU_GAS_REG2_REG1_01	1,714	4,907

Figure 91. Results – endogenous trades in DemoS_005

CO2 Prices

Original Units: MEuro per kton **Active Unit** MEuro per kton **Data values filter:**

Attribute | TimeSlice | CommoditySet

~Scenario~	Region	Commodity	2010	2015	2020
DemoS_005a	REG1	Electricity Plants Carbon dioxide	-0.0026	-0.0026	-0.0032
DemoS_005b	REG1	Electricity Plants Carbon dioxide	-0.0026	-0.0380	-0.0380
		Transport Carbon dioxide	-0.0026	-0.0380	-0.0380
	REG2	Electricity Plants Carbon dioxide		-0.0724	-0.0878
		Transport Carbon dioxide		-0.0724	-0.0878

Figure 92. Results – emissions by sector and by region in DemoS_005

Objective-Function = 3,204,949 M euros (see the _SysCost table in VEDA-BE) with 1,225,688 M euros for REG1 and 1,979,261 M euros for REG2. This cost is less than 0.1% higher with the emission limits for the power sector (3,206,161 M euros) and 1.4% higher with the emission limits for the power and the transportation sectors (3,250,281 M euros). More details about the impacts of the emission limits on the different cost components of the system in each region are shown below (Figure 93).

All system costs (activity, flow, investments and salvage)

Original Units: M Euro **Active Unit** M Euro **Data values filter:**

Process | *Vintage* | *UserConstraint* | *Commodity*

~-Scenario-~	Region	Attribute	-	2005	2006	2010	2015	2020
DemoS_005	REG1	Cost_Act		1,858	1,859	1,893	1,940	1,989
		Cost_Flo		66,600	64,792	66,083	67,572	72,308
		Cost_Fom		14,313	14,478	15,134	15,990	16,873
		Cost_Inv		7,730	8,720	12,666	17,667	22,811
		Cost_Salv	84,969					
		Cost_ire		10,401	8,644	8,936	9,289	9,864
	REG2	Cost_Act		2,045	2,104	2,247	2,445	2,673
		Cost_Flo		122,596	121,957	129,502	135,317	137,951
		Cost_Fom		9,645	10,201	11,651	13,544	15,519
		Cost_Inv		4,192	5,915	12,017	19,793	27,976
		Cost_Salv	81,152					
		Cost_ire		-10,401	-8,644	-8,936	-9,289	-9,864
DemoS_005b	REG1	Cost_Act		1,858	1,859	1,642	1,841	2,169
		Cost_Flo		66,600	64,792	61,976	60,832	60,628
		Cost_Fom		14,313	14,478	14,366	15,686	17,498
		Cost_Inv		7,730	8,720	10,570	17,848	27,283
		Cost_Salv	78,311					
		Cost_ire		10,401	8,644	16,812	29,441	30,616
	REG2	Cost_Act		2,045	2,104	2,483	2,499	2,395
		Cost_Flo		122,596	121,957	135,261	144,316	153,281
		Cost_Fom		9,645	10,201	12,284	13,617	14,726
		Cost_Inv		4,192	5,915	12,996	22,018	32,800
		Cost_Salv	80,071					
		Cost_ire		-10,401	-8,644	-16,812	-29,441	-30,616

Figure 93. Results – emissions by sector and by region in DemoS_005

3.6 DemoS_006 - Multi-region with separate regional templates

Description. At the sixth step, the configuration of the multi-regional model developed previously shifts from a single set of B-Y Templates for all regions to a separate sets of B-Y Templates for each region. This approach is relevant when the model regions are under the control of more than one individual.

Objective. The objective is again to create the multi-regional model framework typical to larger or more complex models, with the trade matrix and limits on emissions of all regions, but additionally to introduce the concept of technology repositories (i.e., SubRES) that include a number of new processes (in competition) that are available in the database to replace the existing ones at the end of their lifetime or to meet an increasing demand.

The motivation behind these repositories is mainly to avoid repeating the new process specifications for each region; all attributes specifications apply to all regions unless a transformation file is used to regionalize some values when necessary.

Simultaneously, the role of the vintage feature is illustrated to handle processes for which characteristics change over time (other than investment cost) when new capacity is built. As in step 5, the scenario variants illustrate the impact of a cap on CO_2 emissions from the electricity sector only and of a cross-region user constraint on the total CO_2 emissions from the transport and electricity sectors.

Attributes introduced:	Files updated
N.A.	SysSettings

	Files created SubRES_NewTechs VT_REG1_PRI_v06 VT_REG2_PRI_v06 Files replaced VT_REG1_PRI_v05

Files. The sixth step model is built 1) by modifying the SysSettings file to add one B-Y Template, 2) by replacing the B-Y Template (VT_REG_PRI_V05) by two B-Y Template (VT_REG1_PRI_v06, VT_REG2_PRI_v06) to disaggregate the energy balance between two regions in two separate files, and 3) by creating a SubRES file to add new processes to the model (Figure 94).

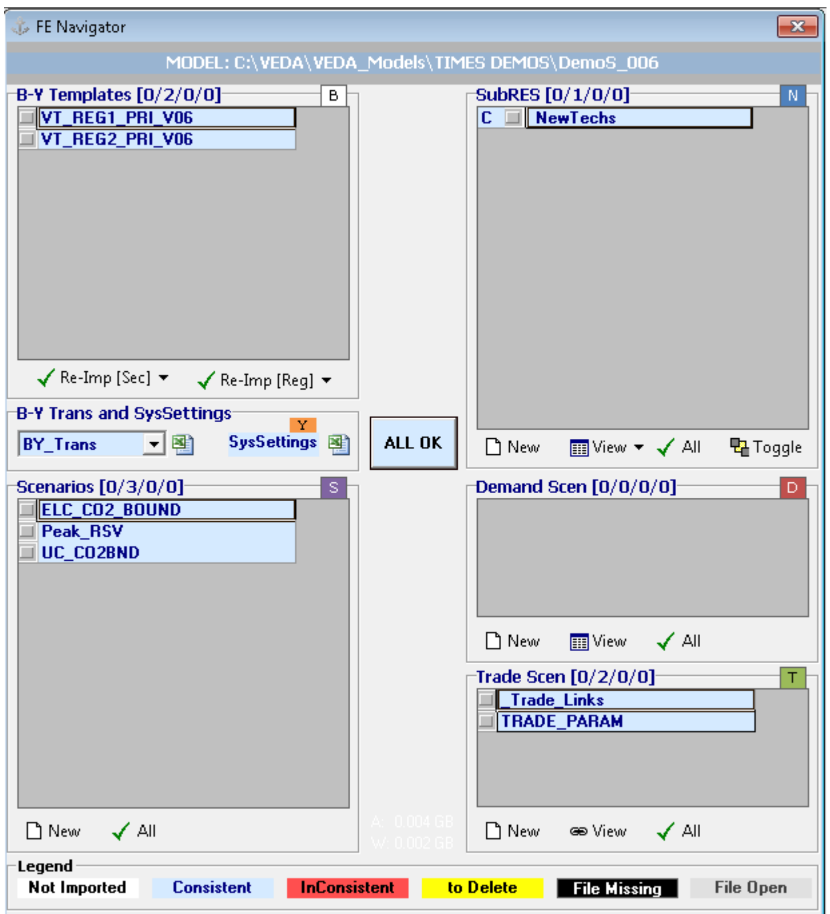


Figure 94. The files included in DemoS_006

3.6.1 SysSettings file

3.6.1.1 <u>Region-Time Slices</u>

The ~BookRegions_Map table is used to create one additional workbook: one for each region REG1 and REG2 (Figure 95).

~BookRegions_Map	
BookName	Region
REG1	REG1
REG2	REG2

Figure 95. New workbook name definitions in the SysSettings file

3.6.2 B-Y Templates

The structure of the two B-Y Templates (VT_REG1_PRI_v06 and VT_REG2_PRI_v06) is identical to the structure of the B-Y Template of the fourth step model and uses the same energy balances defined in the fifth step model for REG1 and REG2 respectively. There is no change to report, except that new power plants are moved from the B-Y Template to the new process repository.

3.6.3 SubRES_NewTechs

Two files are created to add new processes in the model, the SubRES and SubRES_Trans files. The SubRES file is a repository of new processes available for all regions. In the SubRES, by default, all attribute specifications apply to all regions. This approach is convenient for models with multiple regions because a single set of declarations can be made for all regions. The SubRES file includes one sheet for each sector: PRI_ELC, PRI_RSD, PRI_TRA, PRI_FuelSec. (Due to the way SubRES are processed in VEDA-FE, it is required that the name of each sheet start with a valid name of one of the model sectors, as defined in the names of the B-Y templates. In this case, PRI is the only such model sector, and so all sheets in the SubRES template begin with PRI_.)

With this approach, the B-Y Templates now include only processes with existing capacity in the base year 2005, and all new processes are defined in the SubRES. Duplicate definition should be avoided. The new power plants are now declared in this file without any regional specification (Figure 96). Other new processes are created in the other sheets following the same rules: new processes do not have an existing installed capacity, but they are characterized with an investment cost (INVCOST) as well as the year where they become available (START).

The role of the vintage feature is illustrated to handle processes for which characteristics other than investment cost change over time when new capacity is built. In this example, the new gas-fired power plant (ELCTNGAS00) has its efficiency and emission coefficient evolving between 2006 and 2020. The process ELCTNGAS00 is vintaged (Vintage=Yes) in the ~FI_Process table (Figure 97).

			~FI_T										
TechName	Comm-IN	Comm-OUT	Year	START	EFF	AFA	INVCOST	FIXOM	VAROM	LIFE	ENV_ACT	CAP2ACT	Peak
*Technology Name	Input Commodity	Output Commodity			Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Variable O&M Cost	Lifetime	Activity Emission Coefficient	Capacity to Activity Factor	% contribution to PEAK
*Units							M€/GW	M€/PJa	M€/PJ	Years	kt	(Act Unit/Cap Unit)	
ELCTNCOA00	ELCCOA	ELC		2006	0.42	0.85	1650	35.00	0.40	40		31.536	1.00
		ELCCO2									238		
ELCTNOIL00	ELCOIL	ELC		2005	0.30	0.85	250	15.00	0.20	40		31.536	1.00
		ELCCO2									187		
ELCTNGAS00	ELCGAS	ELC		2006		0.85	750	30.00	0.35	30		31.536	1.00
		ELCCO2	2006		0.50						153		
		ELCCO2	2010		0.51						150		
		ELCCO2	2015		0.52						147		
		ELCCO2	2020		0.55						139		

Figure 96. Example of new processes in the SubRES file

~FI_Process

Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
ELE		ELCTNCOA00	Power Plants Existing00 - Solid Fuels	PJ	GW	SEASON		
		ELCTNGAS00	Power Plants Existing00 - Natural Gas	PJ	GW			Yes
		ELCTNOIL00	Power Plants Existing00 - Crude oil and Petroleum Products	PJ	GW			

Figure 97. Example of a new process with vintage tracking in the SubRES file

3.6.3.1 <u>SubRES_NewTechs_Trans</u>

For each SubRES_<user-name> file, there is an associated SubRES_<user-name>_Trans file. The transformation files contain the mapping and transformation operations that control the inheritance (or not) of new processes into the various regions of the model, as well as to change any process characteristics, such as investment costs, by region. In this example, the file is empty, so all new processes in the SubRES are available in both regions with identical characteristics.

3.6.4 Results

The results are very similar to those obtained with the previous step model since most of the changes occurred in the way the information is structured in different files rather than in the energy system itself. However, the impact of the vintage feature for the new gas-fired power plants is illustrated (Figure 98).

ELC Plants Production

Original Units: PJ   Active Unit: Billion Kwh   Data values filter:

Attribute | *TimeSlice*

Region	ProcessSet	-Process-	~-Scenario-~	Vintage	Period 2006	2010	2015	2020
REG2	Gas Power Plants	ELCTNGAS00	DemoS_005	2006	133			
				2010		436		
				2015			825	
				2020				1,224
				Total	133	436	825	1,224
			DemoS_006	2006	10	10	10	10
				2010		235	235	237
				2015			405	417
				2020				560
				Total	10	245	650	1,224

Figure 98. Results – fuel supply options for both regions in DemoS_005

Objective-Function = 3,205,281 M euros (see the _SysCost table in VEDA-BE) with 1,293,017 M euros for REG1 and 1,912,264 M euros for REG2. These costs are similar to those computed with the previous step model DemoS_005.

3.7 DemoS_007 – Adding complexity

Description. The seventh step model is enhanced to capture more components of the energy balance, leading to a more comprehensive representation of the RES with more complex processes.

Objectives. The objective is to show how to model a more comprehensive RES covering more details of the energy balance with more complex processes along its two dimensions: number of commodities and the number of transformation steps in the whole supply-demand chain. In this step refined petroleum products are broken out into different commodities (e.g., gasoline, diesel, heavy fuel, etc.) to better describe the transport sector, where different types of vehicles are introduced. This enhancement of the RES requires the modelling of additional and more complex processes (e.g., refineries and dual demand cars) and the need to introduce the primary commodity group (PCG) concept.

Several more techniques are also introduced in this step:

- We present an easier way to account for combustion-based emissions, by directly linking emission coefficients with each unit of fuel burnt.
- We illustrate how to build end-use demand projections starting from base year values and different growth rates. This is done using the fill table feature to grab base year information from the initial files (e.g. B-Y Templates).
- We show how to build a user constraint that specific the minimum (or maximum) annual growth rate for a set of processes using the CAP, GROWTH attribute.
- Finally, we demonstrate how to use the elastic demand feature of TIMES, including how to generate the file containing the demand prices for base scenarios and how to use these prices for the constrained scenarios.

Attributes introduced:	Files updated
Share	SysSettings
ACTFLO	VT_REG1_PRI_v07
COM_VOC	VT_REG2_PRI_v07
COM_STEP	SubRES_NewTechs
COM_ELAST	Files created
UC_CAP	Scen_DemProj_DTCAR
	Scen_Refinery
	Scen_ElasticDem
	Scen_TRA_CO2_BOUND
	Scen_UC Growth

Files. The seventh step model is built:

1. by modifying the SysSettings file to add interpolation rules;
2. by modifying the two B-Y Template (VT_REG1_PRI_v07, VT_REG2_PRI_v07) and the SubRES file (SubRES_NewTechs) to add more commodities, more complex processes, and emission coefficients, and to introduce the PCG concept;
3. by creating a scenario file to project demand from base year values;
4. by creating a scenario file to update refinery attributes;
5. by creating a scenario file to include price-elasticities for demands;
6. by creating a scenario file with a limit on emissions from the transportation sector;
7. by creating a scenario file with a user constraint on growth rates of new cars (Figure 99).

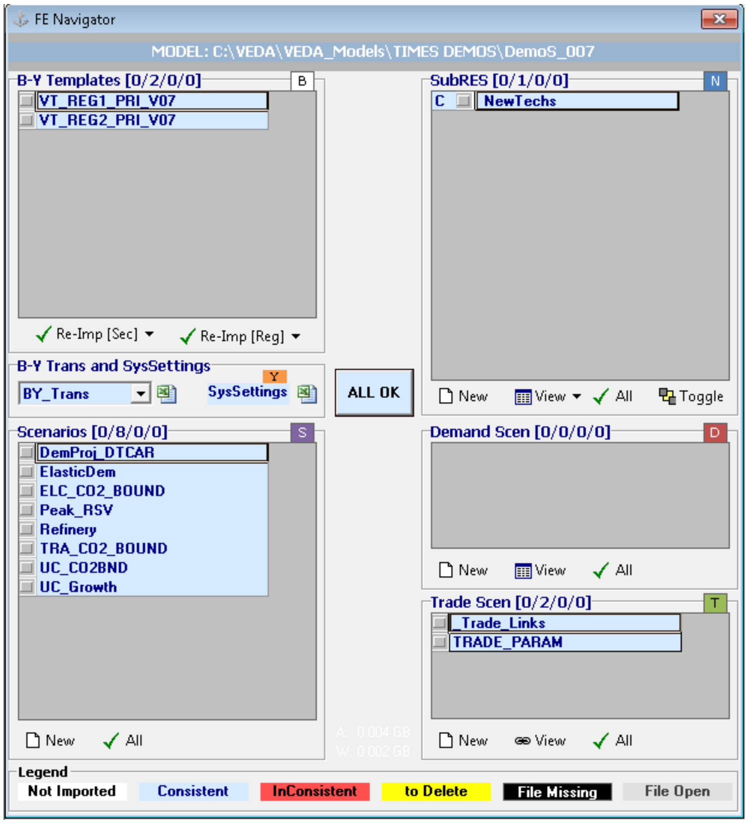


Figure 99. The files included in DemoS_007

3.7.1 SysSettings file

3.7.1.1 Interpol_Extrapol_Defaults

More interpolation/extrapolation rules are added to the transformation table (Figure 100). The same interpolation/extrapolation rule (number 5) is also used for the maximum input shares (Share-I) and the maximum output shares (Share-O) of all processes at once. These new attributes are defined in the next section.

~TFM_UPD					
TimeSlice	LimType	Attribute	Year	AllRegions	Pset_PN
	UP	ACT_BND	0	5	
	UP	Share-O	0	5	
	UP	Share-I	0	5	

Figure 100. Updated table for interpolation/extrapolation rules in the SysSettings file

3.7.2 B-Y Templates

3.7.2.1 EnergyBalance

At this step, the energy balance is disaggregated and includes a larger number of commodities. The crude oil category is disaggregated to track all refined products independently (Figure 101) to better describe the transport sector where different types of cars are introduced. A larger portion of the energy balance is covered in terms of the number of commodities and also of the number of transformation steps in the whole supply-demand chain, with the addition of the refining step.

		COA	GAS	OIL	DSL	KER	LPG	GSL	NAP	HFO	OPP	NUC	RNW	SLU	HET	ELC	
	REG2	Solid Fuels	Natural Gas	Crude oil	Diesel oil	Kerosenes	LPG	Motor spirit	Naphtha	Heavy Fuel Oil	Other Petroleum Products	Nuclear Energy	Renewable Energies	Industrial Wastes	Derived Heat	Electricity	Total
	PRIMARY																
MIN	Domestic Supply	0	7899	3761	0	0	0	0	0	0	0	10775	0	0	0	0	**22435**
IMP	Imports	0	13292	19354	3087	847	457	924	956	1511	836	0	0	0	0	584	**41848**
EXP	Exports	0	-2516	-2308	-2356	-414	-272	-2101	-561	-1735	-634	0	0	0	0	-563	**-13461**
TPS	**Total Primary Supply**	**0**	**18675**	**20807**	**730**	**433**	**184**	**-1177**	**395**	**-224**	**202**	**10775**	**0**	**0**	**0**	**20**	**50822**
	CONVERSION																
ESC	Energy Sector Consumption	0	-793	0	-23	0	-740	-230	-1	-288	0	0	0	-1	0	0	**-2076**
ELC	Electricity Plants	0	-5636	0	-42	0	-33	0	0	-735	-47	-10775	0	-16	869	5791	**-10625**
HPL	Heat Plants	0	-301	0	-11	0	0	0	0	-21	-2	0	0	-1	329	0	**-7**
REF	Petroleum Refineries	0	0	-22216	7982	1357	1521	4697	1358	3199	1820	0	0	0	0	0	**-281**
	Total Conversion	**0**	**-6730**	**-22216**	**7906**	**1357**	**747**	**4467**	**1358**	**2155**	**1771**	(Ctrl)	**0**	**-18**	**1198**	**5791**	**-12990**
	FINAL																
RSD	Residential	0	5160		1207	102	266	4	0	22	1	0	0	0	433	1436	**8631**
COM	Commercial	0	1752		516	2	44	8	0	27	0	0	0	1	127	1264	**3742**
IND	Industry	0	4437		418	51	200	11	62	400	268	0	0	59	317	2044	**8267**
AGR	Agriculture	0	201		513	1	23	2	0	19	0	0	0	0	8	10	**777**
TRA	Transport	0	21		5399	1467	132	3352	0	47	0	0	0	0	0	133	**10550**
OTH	Other	0	0		0	0	0	0	0	0	0	0	0	0	314	325	**639**
NEN	Non Energy	0	634		107	7	280	4	1259	73	1121	0	0	0	0	0	**3485**
BNK	Bunkers	0	0		206	0	0	0	0	1263	9	0	0	0	0	0	**1478**
TFC	**Total Final Consumption**	0	**12205**		**8366**	**1629**	**945**	**3382**	**1321**	**1851**	**1400**	**0**	**0**	**59**	**1198**	5211	**37568**

Figure 101. Disaggregated energy balance at start year 2005 for REG2 – Covered in DemoS_007

3.7.2.2 Con_REF – primary commodity group definition

A flexible refinery (REFEOIL00) is introduced in this sheet (Figure 102) to convert crude oil (OIL) into refined products (DSL, KER, LPG, GSL, etc.) that will be used in the transportation sector.

- The existing refinery is characterized with an efficiency (EFF) and an annual activity bound (ACT_BND) equivalent to the sum of the refined products produced at base year 2005 as given in the energy balance. In this example the efficiency is represented by the ratio of the crude oil in input to the refinery on the sum of the petroleum products in output. For this reason we get an efficiency greater than 1. This behaviour depends on the definition of the commodity group of a technology (see below for more details).
- This more complex process with multiple outputs commodities is also characterized with a new attribute: the maximum share for each commodity output in the total production (Share-O~UP). In this example, the maximum shares for all outputs sum to 100%, meaning that they are equivalent to fixed shares. It would be possible to have a sum of maximum shares greater than 100%, leaving some flexibility to the model to optimize the output mix.

The same approach is used to declare the new commodities and processes in their definition tables, where the refinery is declared as a PRE process, and the concept of Primary Commodity Group (PCG) is introduced (Figure 103). The activity of a standard process is equal to the sum of the commodity flow(s) on either the input side or the output side of a process, as defined by the PCG. The activity of a process is limited by the available capacity, so that the activity variable establishes a link between the installed capacity of a process and the maximum possible commodity flows entering or leaving the process during a year or a subdivision of a year.

		~FI_T			
TechName	Comm-IN	Comm-OUT	Share-O~UP	EFF	ACT_BND
*Technology Name	Input Commodity	Output Commodity	Output Share	Efficiency	Activity Bound
*Units			Pja		
REFEOIL00	OIL	DSL	36%	1.01	9400
		KER	6%		
		LPG	7%		
		GSL	21%		
		NAP	6%		
		HFO	15%		
		OPP	8%		

Figure 102. Refinery

In a simple process, one consuming a single commodity and producing a single commodity, the modeler simply chooses one of these two flows to define the activity, and thereby the process normalization (input or output). In complex processes, with several commodities (perhaps of different types) as inputs and/or outputs, the definition of the activity variable requires designation of the PCG to serve as the activity-defining group. The PCG is defined as a subset of the commodities of the same nature entering or leaving a process. For instance, the PCG may be the group of energy carriers, or the group of materials of a given type, on either the input or output side of the process. More about PCGs and their use can be found in Section 2.2.1 of Part II of the TIMES documentation.

VEDA-FE establishes default PCGs for any process involving multiple inputs and/or outputs, based upon the assumption first that all processes are output normalized and then according to the commodities' nature. In case of different commodity types on the output (or input) side, the default PCG is based on the following order:

- DEM
- MAT
- NRG
- ENV
- FIN

However, in some cases it is desirable/necessary to override these defaults, for instance to normalize a process with energy commodities inputs (NRGI) as for the refinery in this example. Indeed, the activity of a refinery is usually characterized based on the barrels of crude oil consumed.

~FI_Process								
Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
*								
PRE		REFEOIL00	Refinery Existing00	PJ	Pja		NRGI	

Figure 103. Overwrite default PCG for the refinery

3.7.2.3 Pri_PP

Import and export options for all refined petroleum products were added in this sheet; they are characterized with the COST and ACT_BND attributes as for any other primary fuels (solid fuels, natural gas, crude oil) (Figure 104). Note that by convention, the export prices are generally be slightly less than import prices, to avoid the model importing just to export.

3.7.2.4 Sector_Fuels

Additional sector fuel processes (FTE-*) are defined and characterized in this sheet (Figure 105), namely to produce the transportation sector fuels from primary refined products (e.g. GSL to TRAGSL). It is not always relevant to keep track of all primary fuels in a sector; multiple primary fuels can be aggregated into a single sector fuel in this case. In this example, several refined products are aggregated into a single electricity sector fuel (via FTE-ELCOIL). When more than one primary fuel are used to create one sector fuel, the shares of input fuels (Share-I~UP) need to be provided. As with Share-O, the maximum input shares may sum to greater than 100%, if desired, to provide some process flexibility.

				~FI_T			
TechName	Comm-IN	Comm-OUT	Year	LimType	CUM	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity			Reserves Cumulative Value	Cost	Annual Production Bound
*Units					PJ	M€2005/PJ	PJ
IMPDSL1		DSL				10.40	
IMPKER1		KER				11.20	
IMPLPG1		LPG				8.80	
IMPGSL1		GSL				11.20	
IMPNAP1		NAP				8.40	
IMPHFO1		HFO				8.40	
IMPOPP1		OPP				8.40	
EXPDSL1	DSL					10.30	1010
EXPKER1	KER					11.09	177
EXPLPG1	LPG					8.71	117
EXPGSL1	GSL					11.09	900
EXPNAP1	NAP					8.32	241
EXPHFO1	HFO					8.32	744
EXPOPP1	OPP					8.32	272

Figure 104. Imports and exports of refined petroleum products

		~FI_T				
TechName	Comm-IN	Comm-OUT	Share-I~UP	STOCK	EFF	LIFE
*Technology Name	Input Commodity	Output Commodity	Input Share	Existing Installed Capacity	Efficiency	Lifetime
*Units				PJa		Years
FTE-RSDGAS	GAS	RSDGAS			1.00	50
FTE-TRADSL	DSL	TRADSL			1.00	50
FTE-TRAKER	KER	TRAKER			1.00	50
FTE-TRALPG	LPG	TRALPG			1.00	50
FTE-TRAGSL	GSL	TRAGSL			1.00	50
FTE-TRAHFO	HFO	TRAHFO			1.00	50
FTE-TRAELC	ELC	TRAELC			1.00	50
FTE-TRAGAS	GAS	TRAGAS			1.00	50
FTE-ELCCOA	COA	ELCCOA			1.00	50
FTE-ELCGAS	GAS	ELCGAS			1.00	50
FTE-ELCOIL	DSL	ELCOIL	5%		1.00	50
	LPG		4%			
	HFO		86%			
	OPP		5%			
FTE-ELCRNW	RNW	ELCRNW			1.00	50
FTE-ELCNUC	NUC	ELCNUC			1.00	50

Figure 105. Additional sector fuel processes with multiple inputs

3.7.2.5 DemTechs_TRA

The single demand process consuming an energy commodity (TRAOIL) and producing directly the transport demand commodity (DTD1) is replaced with more sophisticated processes representing cars and characterized with non-energy units (Figure 106). The declaration of these processes is shown below (Figure 107): their activity units are in billions passengers-kilometres (BpK) rather than PJ, and their capacity units are in thousands of units (000_units) rather than PJa.

- The existing processes are characterized with their existing installed capacity (STOCK) in thousands of car units (000_units) as indicated above. The stock values correspond to the amount of fuel consumption (e.g. TRADSL) required to produce the transportation demand (DTCAR) as given by the energy balance and taking into account the efficiency (EFF), the annual availability factor (AFA) and the conversion between capacity unit and activity unit (CAP2ACT).
- The efficiency (EFF) is specified in terms of billions of vehicle-kilometres per petajoule (BVkm/PJ), and can be interpreted as the number of kilometres a vehicle can travel with 1 PJ of energy.
- The annual availability factor (AFA) represents the average thousand kilometres ('000 km) a car is traveling each year.
- A new attribute is introduced to capture the relation between the process activity and the commodity flow (ACTFLO), the commodity being the output demand, in terms of passengers per car unit (Passenger/Car). This TIMES parameter requires an additional index that is the specification of the commodity group: DEMO (demand out) in this example.
- The life time (LIFE) is specified in number of years as for the other processes.
- The conversion factor between capacity unit and activity unit (CAP2ACT) is not equal to 1 because the units are different: the activity is in billion vehicle-kilometres, the stock is in thousands of units (000_units or vehicles) and the utilization factor (AFA) is in thousand kilometres per vehicle. The CAP2ACT is translating mvkm into bvkm.

		~FI_T								
TechName	Comm-IN	Comm-OUT	STOCK	EFF	AFA	ACTFLO~DEMO	INVCOST	FIXOM	LIFE	CAP2ACT
*Technology Name	Input Commodity	Output Commodity	Existing Installed	Efficiency	Utilisation Factor	Activity to Flo	Invesotment Cost	Fixed O&M Cost	Lifetime	
*Units			000_Units	BV*km/PJ	'000 km	Passenger/Car	MI2005/000_Units	MI2005/000_Unitsa	Years	bvkm/mvkm
TCAREDSL	TRADSL	DTCAR	58069	0.41	17	1.25		0.16	10	0.001
TCARELPG	TRALPG	DTCAR	1550	0.38	14	1.25		0.16	10	0.001
TCAREGSL	TRAGSL	DTCAR	50466	0.40	12	1.25		0.15	10	0.001
TCAREGAS	TRAGAS	DTCAR	0	0.38	14	1.25		0.16	10	0.001

Figure 106. More complex processes in the transportation sector

~FI_Process								
Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process	Primary Commodity	Vintage Tracking
*								
DMD		TCAREDSL	Demand Technologies Transport Sector - Existing Cars - Diesel oil	BPkm	000_Units		DEMO	
		TCARELPG	Demand Technologies Transport Sector - Existing Cars - LPG	BPkm	000_Units		DEMO	
		TCAREGSL	Demand Technologies Transport Sector - Existing Cars - Motor spirit	BPkm	000_Units		DEMO	
		TCAREGAS	Demand Technologies Transport Sector - Existing Cars - Natural Gas	BPkm	000_Units		DEMO	

Figure 107. Declaration of more processes in the transportation sector

3.7.2.6 Demands

The demand for transportation by cars is updated and declared in the right units and correspond to the sum of billion passengers-kilometres (Bpass*km) for all types of cars (Figure 108):

- Demand (Bpass*km) = STOCK (000_units) * AFA (000_vehiclekm/unit) * ACTFLO~DEMO (Passengers/vehicle)* CAP2ACT(0.001bvkm/mvkm)

	~FI_T						
Attribute	CommName	*Unit	2005	2006	2010	2015	2020
*	Demand Commodity Name	Demand Unit	Demand Value				
*Units			PJ				
Demand	TPSCOA	PJ	3597				
Demand	DROT	PJ	0				
Demand	DTCAR	Bpass*km	1950				
Demand	TPSELC	PJ	5211	5264	5477	5757	6050

Figure 108. Demand for transportation by car in physical units

3.7.2.7 Emi

A new sheet is added to introduce a comprehensive and convenient approach to account for combustion emissions by sector. Indeed, the easiest way to account for combustion emissions is to directly associate the fuel-based emission coefficients with fuel consumption throughout the whole energy system.

A new ~COMEMI table is added (Figure 109) to define fuel-based emission coefficients instead of defining emission coefficients for each process in all ~FI_T tables. The special tag ~COMEMI is used to link emissions to commodity consumption through special processing in the VEDA-FE SYNC process. (The VEDA-TIMES parameters VDA-EMCB and FLO-EMIS provide alternative ways to declare consumption-linked emissions. See Part II of the TIMES documentation for more on the use of these parameters.)

In this example, emissions of TRACO2 are associated with six fuels (LPG, gasoline, kerosene, diesel, heavy fuel oil, natural gas,) for which a coefficient (kt/PJ) is provided. These coefficients are applied to all the fuel consumption by all the individual processes in the transportation sector.

~COMEMI						
CommName	TRALPG	TRAGSL	TRAKER	TRADSL	TRAHFO	TRAGAS
*Units	kt/PJ	kt/PJ	kt/PJ	kt/PJ	kt/PJ	kt/PJ
TRACO2	65.00	72.00	74.00	74.00	78.00	56.00

Figure 109. Combustion emissions from the transportation sector

3.7.3 SubRES_NewTechs

3.7.3.1 PRI_TRA

This sheet is updated to model the new cars using the same approach as described above for the existing cars.

3.7.4 Scenario files

Several scenario files are created at this seventh step.

3.7.4.1 Scen_DemProj_DTCAR

This scenario file is created to project transport demand using a fill table to grab base year values from B-Y templates (Figure 110). The **~TFM_FILL** table (see section 2.3.3 for more information) is a feature allowing a template to collect information from other templates. In this example, the table is collecting the base year values (YEAR=2005) from the B-Y templates (Scenario = BASE) for the transportation demand (Attribute=Demand) by cars (commodity = DTCAR). VEDA-FE fills in the REG1 and REG2 values in the blue highlighted cells each time the template is SYNCed.

~TFM_FILL								
Operation_Sum_Avg_Count	Scenario	TimeSlice	LimType	Attribute	Year	REG1	REG2	Cset_CN
A	BASE			Demand	2005	1950.24	4560.75	DTCAR

Figure 110. Grab base year demand values from B-Y templates

The DTCAR demand is then projected to 2020 in the **~TFM_INS** table using the base year values and some multipliers (2% for REG1 and 3% for REG2) defined by the user (Figure 111).

~TFM_INS							Demand Driver (annual growth)	
TimeSlice	LimType	Attribute	Year	REG1	REG2	Cset_CN	Reg1	Reg2
		Demand	2006	1989.2	2008.7	DTCAR	2%	3%
		Demand	2010	2153.2	2260.9	DTCAR		
		Demand	2015	2377.3	2621.0	DTCAR		
		Demand	2020	2624.8	3038.4	DTCAR		

Figure 111. Using base year values to project end-use demands

3.7.4.2 Scen_Refinery

This scenario file is created to update refinery attributes, again using a fill table to grab information from B-Y templates (Figure 112). In this example, the table is collecting the base year values (YEAR=2005) from the B-Y templates (Scenario = BASE) for the activity production bound (Attribute=ACT_BND) of the refinery (process = REFEOIL00).

~TFM_FILL								
Operation	Scenario	TimeSlice	LimType	Attribute	Year	REG1	REG2	Pset_PN
A	BASE			ACT_BND	2005	9400.42	21934.32	REFEOIL00

Figure 112. Grab base year attribute values from B-Y templates

The activity production is then projected to 2020 in the ~TFM_INS table using the base year values and some relaxation factors (25% for REG1 and 30% for REG2) defined by the user (Figure 113). In addition, the maximum (UP) shares of the refinery outputs (Attribute=SHARE-O) are all updated to 50%, creating flexibility for the model to optimize the mix of refined products (DSL, KER, LPG, etc.).

~TFM_INS															
TimeSlice	LimType	Attribute	Year	Other_Indexes	AllRegions	REG1	REG2	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
	UP	Share-O	2020	NRGO		50%	50%		REFEOIL00					DSL	
	UP	Share-O	2020	NRGO		50%	50%		REFEOIL00					KER	
	UP	Share-O	2020	NRGO		50%	50%		REFEOIL00					LPG	
	UP	Share-O	2020	NRGO		50%	50%		REFEOIL00					GSL	
	UP	Share-O	2020	NRGO		50%	50%		REFEOIL00					NAP	
	UP	Share-O	2020	NRGO		50%	50%		REFEOIL00					HFO	
	UP	Share-O	2020	NRGO		50%	50%		REFEOIL00					OPP	
	UP	ACT_BND	2020			11751	28515		REFEOIL00						

Figure 113. Using base year values to update refinery attributes

3.7.4.3 Scen_TRA_CO2_BOUND

This file is used to introduce bounds (limits) on the CO2 emissions from the transportation sector in REG1 and REG2. A transformation table ~TFM_INS is used (Figure 114) to declare upper bounds on annual emissions (Attribute = COM_BNDNET; LimType = UP), on the CO2 emissions from the transportation sector only (TRACO2) in REG1 and REG2. These upper bounds are calculated as percentage reduction targets from the transportation sector CO2 emissions in a reference scenario for 2010 (10%) and 2020 (20%). It is necessary to run the step model without any limit on emissions first to get the reference emission trajectory (run DemoS_007) and then calculate the bounds as a reduction targets from the reference emissions. An interpolation rule is used with the "0" flag in the Year column and the interpolation/extrapolation option in the region column where the bounds are declared; the code 5 means full interpolation and forward extrapolation.

~TFM_INS															
TimeSlice	LimType	Attribute	Year	AllRegions	REG1	REG2	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
	UP	COM_BNDNET	2010		279594	293300							TRACO2	
	UP	COM_BNDNET	2020		303192	350973							TRACO2	
	UP	COM_BNDNET	0		5	5							TRACO2	

Figure 114. Declaration of emission bounds for the transportation sector

3.7.4.4 Scen_UC Growth

This file shows another type of user constraint that specifies the maximum (or minimum) annual growth rate for a set of processes using the CAP, GROWTH attribute (Figure 115). (See Section 2.3.8 for more on user constraints.)

This user constraint imposes a maximum capacity (defined by UC_CAP) growth rate (CAP,GROWTH) of 1% per year (value in the column UC_CAP) for cars consuming TRADSL (these cars are identified using the two columns PSET_CO and PSET_CI). This constraint also provides a seed value of 1 (column UC_RHSRTS) to enable the capacity growth to start in case the existing capacity of diesel cars is zero.

~UC_Sets: R_E: AllRegions											
~UC_Sets: T_SUC:											
							~UC_T				
UC_N	Pset_Set	Pset_PN	Pset_CI	Pset_CO	UC_ATTR	Year	LimType	UC_CAP	UC_CAP~RHS	UC_RHSRTS	UC_RHSRTS~0
UC_GrowthOfNewCars			TRADSL	DTCAR	CAP, GROWTH		LO	1.01	1	-1	5

Figure 115. Specifying growth rates with a user constraint

3.7.4.5 Scen_ElasticDem

This file is used to introduce price-elasticities for end-use demands (Figure 116), so that demands can react to changes in their prices under a constrained energy system (e.g., under limits or tax on emissions, etc.). (See Section 4.2 of Part I of the TIMES documentation for more on the elastic demand formulation.)

In this example, price-elasticities are declared for the transportation demand by cars (DTCAR). Three attributes need to be declared:

- COM_ELAST: Elasticity of demand indicating how much the demand rises/falls in response to a unit change in the marginal cost of meeting a demand that is elastic.
- COM_VOC: Maximum possible variation of demand in both directions when using the elastic demand formulation (15% in this example).
- COM_STEP: Number of steps for the linear approximation of the demand curve (10 steps in this example).

~TFM_INS

TimeSlice	LimType	Attribute	Year	Cset_CN	AllRegions
	UP,LO	COM_VOC	2006	DTCAR	0.15
	UP,LO	COM_VOC	0	DTCAR	5
	UP,LO	COM_STEP		DTCAR	10

~TFM_INS

TimeSlice	LimType	Attribute	Year	Cset_CN	REG1	REG2
ANNUAL	LO	COM_ELAST	2006	DTCAR	-0.0330	-0.0330
ANNUAL	UP	COM_ELAST	2006	DTCAR	-0.0330	-0.0330
ANNUAL	LO	COM_ELAST	2020	DTCAR	-0.1500	-0.1500
ANNUAL	UP	COM_ELAST	2020	DTCAR	-0.0500	-0.0500

Figure 116. Declaring price-elasticities for end-use demands

In order to activate the elastic demand feature, there are few steps to follow:

- Generate a file with demand prices from a reference case, i.e. without any constraint or tax on emissions: in the Control Panel of the FE Case Manager, make sure the option “Write B Price for Elast Dem" is selected (Figure 117). This option is already selected in the DemoS_007.
- Solve a constrained case with price-elasticity:
    - Select the constrained scenarios you want to include in the model run (emission limits or taxes) as well as the elastic demand scenario.
    - In the FE Case Manager, click on **No Elast DEM** below Base Price and select the reference case that was run to get the demand base prices (see right side of Figure 118 where DemoS_007b is a constrained case run with elastic demands, while DemoS_007a on the left side is a constrained case run without elastic demands).

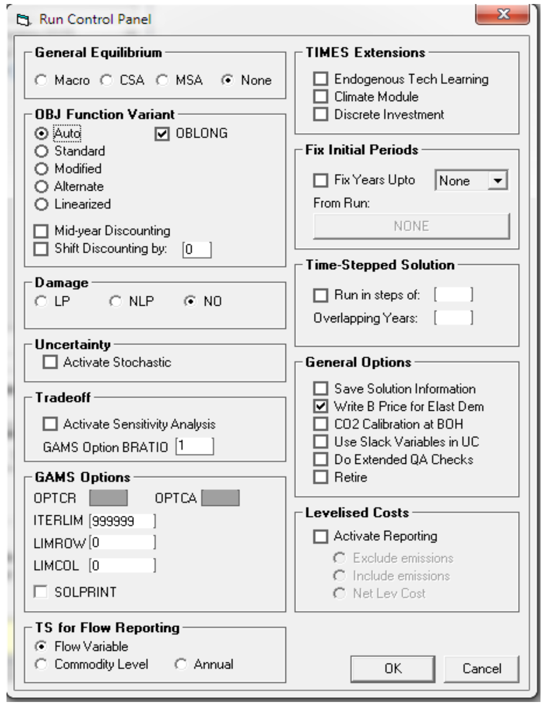


Figure 117. Write base prices for elastic demands

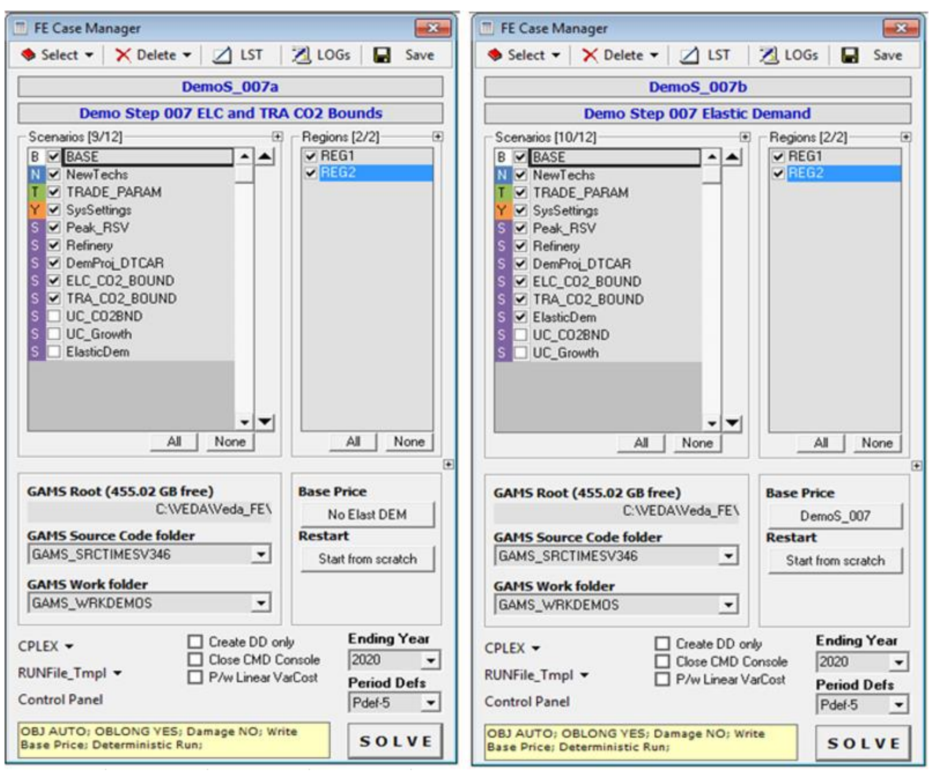


Figure 118. Activate the elastic demand in constrained runs

3.7.5 Results

The effect of price elasticities on the new projected demand for car transportation in thousand passengers-kilometres (kpass*km) to the 2020 horizon is visible (Figure 119) in the scenarios where it was activated (DemoS_007b and DemoS_007c). Demands are decreasing by about 9% in both regions, less than the maximum decrease of 15%, meaning than more cost-effective emission reduction options exist elsewhere in the system beyond that level.

The impacts of the emissions constraints and the growth rate constraint on the optimal process mix selected to meet the car transportation demand (kpass*km) is shown (Figure 120) for both regions together:

Demands

Original Units: Active Unit Data values filter:

Attribute | *Vintage* | TimeSlice | CommoditySet | *Process*

~Scenario~	-Commodity-	Region	2005	2006	2010	2015	2020
DemoS_007	Demand Transport Sector - Cars	REG1	1,950	1,989	2,153	2,377	2,625
		REG2	4,561	2,009	2,261	2,621	3,038
DemoS_007a	Demand Transport Sector - Cars	REG1	1,950	1,989	2,153	2,377	2,625
		REG2	4,561	2,009	2,261	2,621	3,038
DemoS_007b	Demand Transport Sector - Cars	REG1	1,950	1,989	2,056	2,235	2,389
		REG2	4,561	2,009	2,036	2,424	2,765
DemoS_007c	Demand Transport Sector - Cars	REG1	1,950	1,989	2,056	2,199	2,389
		REG2	4,561	2,009	2,036	2,424	2,765

Figure 119. Results - Effect of price elasticities for the car transportation demand in DemoS_007

Demands

Original Units: Active Unit Data values filter:

Attribute | *Vintage* | TimeSlice | CommoditySet | *Region*

~Scenario~	-Commodity-	Process	2005	2006	2010	2015	2020
DemoS_007	Demand Transport Sector - Cars	TCAREDSL	3,992	3,077	1,996		
		TCAREGAS	10	9	5		
		TCAREGSL	2,418	653	1,209		
		TCARELPG	90	24	26		
		TCARNDSL		234	1,178	4,998	5,663
		Total	6,511	3,998	4,414	4,998	5,663
DemoS_007a	Demand Transport Sector - Cars	TCAREDSL	3,992	3,077	1,752		
		TCAREGAS	10	9	5		
		TCAREGSL	2,418	653	1,209		
		TCARELPG	90	24	45		
		TCARNDSL		174	906	3,751	3,694
		TCARNELC			437	722	1,053
		TCARNGAS					245
		TCARNLPG		60	60	526	672
		Total	6,511	3,998	4,414	4,998	5,663
DemoS_007b	Demand Transport Sector - Cars	TCAREDSL	3,992	3,077	1,996		
		TCAREGAS	10	9	5		
		TCAREGSL	2,418	653	964		
		TCARELPG	90	24	45		
		TCARNDSL		174	906	3,751	3,694
		TCARNELC			115	382	543
		TCARNGAS					245
		TCARNLPG		60	60	526	672
		Total	6,511	3,998	4,092	4,659	5,153
DemoS_007c	Demand Transport Sector - Cars	TCAREDSL	3,992	3,077	1,996		
		TCAREGAS	10	9	5		
		TCAREGSL	2,418	653	964		
		TCARELPG	90	24	45		
		TCARNDSL		132	660	2,792	2,934
		TCARNELC			102	165	376
		TCARNGAS				934	1,148
		TCARNLPG		102	320	733	695
		Total	6,511	3,998	4,092	4,623	5,153

Figure 120. Results – Car transport process mix in DemoS_007

- In the reference case (DemoS_007), new diesel cars satisfy the entire demand for car transportation from 2015 and beyond. The output mix of the refinery is shown below (Figure 121).
- The limits on the transportation sector emissions (DemoS_007a) lead to a switch toward less polluting options such as electric, natural gas and LPG cars.
- The activation of elastic demand (DemoS_007b) leads to a reduction in the use of the most expensive option to meet demand – electric cars.
- The addition of a growth rate constraint on diesel cars (DemoS_007c) leads to a switch toward natural gas cars.

Refinery input and output

Original Units: PJ   Active Unit PJ   Data values filter:

Process | *Vintage* | TimeSlice | *Region*

~Scenario~	Attribute	Commodity	2005	2006	2010	2015	2020
DemoS_007	VAR_FIn	Crude Oil	21,002	16,000	18,172	17,907	16,544
		Total	21,002	16,000	18,172	17,907	16,544
	VAR_FOut	Diesel oil	9,074	7,684	7,303	9,513	8,167
		Heavy Fuel Oil	2,451	2,383	2,479	2,479	2,479
		Kerosenes	626	591	591	591	591
		LPG	654	440	443	389	389
		Motor spirit	6,223	2,992	5,419	3,001	3,001
		Naphtha	774	802	802	802	802
		Other Petroleum Products	933	906	906	906	906
		Total	20,737	15,798	17,942	17,680	16,335
DemoS_007c	VAR_FIn	Crude Oil	21,002	16,800	18,872	18,497	18,963
		Total	21,002	16,800	18,872	18,497	18,963
	VAR_FOut	Diesel oil	9,074	7,978	7,769	8,552	9,091
		Heavy Fuel Oil	2,451	2,479	2,479	2,479	2,479
		Kerosenes	626	591	591	591	591
		LPG	654	656	1,157	1,933	1,853
		Motor spirit	6,223	3,176	4,930	3,001	3,001
		Naphtha	774	802	802	802	802
		Other Petroleum Products	933	906	906	906	906
		Total	20,737	16,588	18,633	18,263	18,723

Figure 121. Results - Flexible refinery output in DemoS_007

Objective-Function = 5,484,966 M euros (see the _SysCost table in VEDA-BE) with 2,859,389 M euros for REG1 and 2,625,577 M euros for REG2. These costs are higher than those computed with the previous step model DemoS_006 because of the many components added to the RES. The total cost is 12% higher when emissions limits are imposed on the transportation sector (6,145,863 M euros), but only 7% higher with the activation of elastic demand as the model has more flexibility to reach the emissions targets (5,891,267 M euros). The addition of the growth rate constraint on diesel cars brings the system cost increase back up to 10% (6,025,956 M euros).

3.8 DemoS_008 - Split Base-Year (B-Y) templates by sector: demands by sector

Description. At the eighth step, the level of detail in the representation of the RES is expanded further, the base-year information is disaggregated into different B-Y Templates for each sector, and demands are projected through 2050. Each of these B-Y Templates utilizes only the relevant portion of the energy balance for its region and is linked to an additional single file containing the complete regional energy balances. This approach is convenient when different individuals work in parallel on different sectors. In addition, it encourages grouping of related commodities and processes, and as the size of a model grows it improves (and speeds up) the process of managing the model.

Objective. The objective is to give more examples on how to further expand the detail of the representation of the RES, in terms of the number of end-use demand segments and end-use devices as well as commodities. On the demand side, the idea is to cover the energy consumption by end-use in all sectors rather than by type of energy: agriculture (one end-use demand), commercial (three end-use demands), residential (three end-use demands), industrial (one end-use demands), and transport (two end-use demands). On the supply side, the idea is to break the renewables into more detail for wind, solar, hydro and biomass power. This enhancement of the RES requires the modelling of additional processes as well as the addition of emission coefficients for all sectors.

Another objective is to show how to impose a limit on power generation capacity: nuclear, for example. The scenario variants with nuclear maximum capacity, with different types of limits on emissions, and with and without the elastic demand feature, illustrate the impacts on the respective contribution of each sector to the target as well as on the electricity generation mix.

Attributes introduced:	Files updated
N.A.	SysSettings Scen_TRA_CO2_Bound Scen_ELC_CO2_Bound Scen_UC_CO2BND SubRES_NewTechs Files created VT_REG1_PRI_v08 VT_REG1_ELC_v08 VT_REG1_RCA_v08 VT_REG1_TRA_v08 VT_REG1_IND_v08 VT_REG2_PRI_v08 VT_REG2_RCA_v08 VT_REG2_ELC_v08 VT_REG2_TRA_v08 VT_REG2_IND_v08 Scen_UC_NUC_MaxCAP Files replaced VT_REG1_PRI_v07 VT_REG2_PRI_v07

Files. The eighth step model is built:

1. by modifying the SysSettings file to add more time periods;

2. by replacing the two B-Y Templates (VT_REG1_PRI_v07, VT_REG2_PRI_v07) by five B-Y Templates – one for each sector – in each region (VT_REG1_*_v08, VT_REG2_*_v08), and to add more energy commodities, energy processes, and emissions;
3. by completing the SubRES file;
4. by updating scenario files with limits on emissions;
5. by creating a scenario file with a user constraint on the maximum nuclear power capacity (Figure 122).

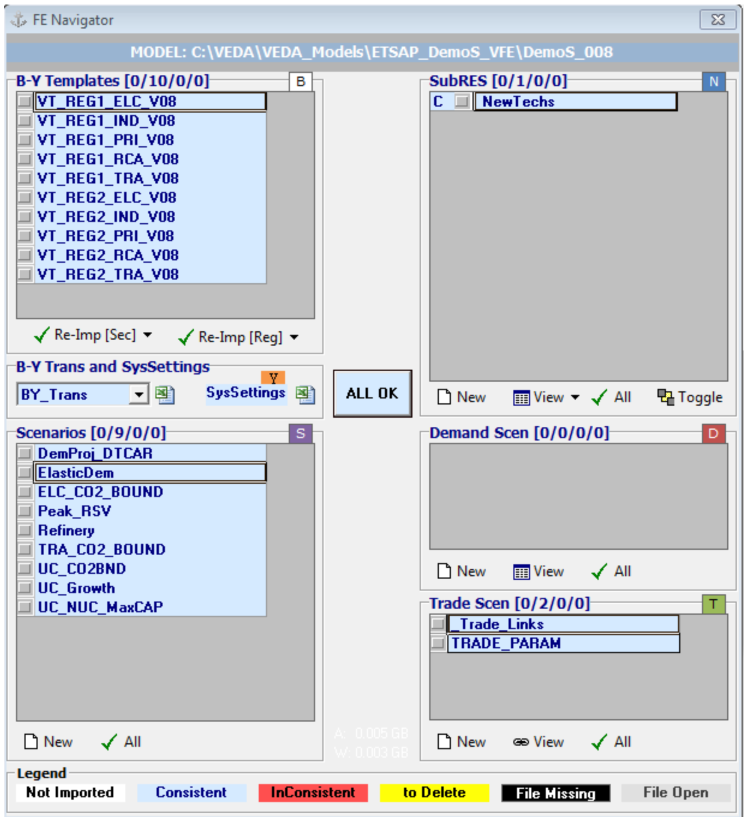


Figure 122. The files included in DemoS_008

3.8.1 SysSettings file

3.8.1.1 TimePeriods

The ~TimePeriods table is used to extend the time horizon of the model by adding six active periods of five years (Figure 123). These specifications are saved under a new time period definition (Pdef-11). The time horizon is extended to 2050 with the milestones years being 2005, 2006, 2010, 2015, 2020, 2025, 2030, 2035, 2040, 2045 and 2050. This can be seen in VEDA-FE, Advanced Functions menu, MileStone Years tab.

~TimePeriods		
Pdef-1	Pdef-5	Pdef-11
1	1	1
2	2	2
	5	5
	5	5
	5	5
		5
		5
		5
		5
		5
		5

Figure 123. New time periods definition in the SysSettings file

3.8.1.2 <u>Defaults</u>

The ~DefUnits table is used to specify the different default activity, capacity and commodity units for each sector in the model (Figure 124).

~DefUnits					
Option	PRI	ELC	IND	RCA	TRA
Process_ActUnit	PJ	PJ	PJ	PJ	kPk
Process_CapUnit	Pja	GW	GW	Pja	000_Units
Commodity_Unit	PJ	PJ	PJ	PJ	kPk

Figure 124. Default declarations in the SysSettings file

3.8.2 B-Y Template VT_REG*_PRI_V08

3.8.2.1 <u>EnergyBalance</u>

The energy balance is disaggregated further and includes a larger number of commodities. The renewable category is disaggregated to track several sources independently: biomass as well as hydro, wind, and solar energy (Figure 125). Moreover, the energy balances of both regions are now moved into a separate file (called EnergyBalance) and all B-Y Templates are linked to this file to grab the relevant sector data.

		COA	GAS	OIL	DSL	KER	LPG	(...)	NUC	BIO	HYD	WIN	SOL	SLU	HET	ELC	TOT
	REG1	Solid Fuels	Natural Gas	Crude Oil	Diesel oil	Kerosenes	LPG	(...)	Nuclear Energy	Biomass	Hydro power	Wind energy	Solar energy	Industrial Wastes	Derived Heat	Electricity	Total
	PRIMARY																
MIN	Domestic Supply	5264	3160	2686	0	0	0		4455	2262	503	264	126	0	0	0	18719
IMP	Imports	4201	5317	13824	2205	605	326		0	85	0	0	0	0	0	584	30166
EXP	Exports	-746	-1007	-1648	-1683	-295	-195		0	-54	0	0	0	0	0	-563	-9785
TPS	Total Primary Supply	8719	7470	14862	522	310	132	(...)	4455	2292	503	264	126	0	0	20	39100
	CONVERSION																
ESC	Energy Sector Consumption	-37	-317	0	-16	0	-529		0	-3	0	0	0	-1	0	0	-1275
ELC	Electricity Plants	-6239	-2254	0	-30	0	-24		-4455	-527	-503	-264	-68	-16	869	5791	-8279
HPL	Heat Plants	-105	-121	0	-8	0	0		0	-105	0	0	0	-1	329	0	-27
REF	Petroleum Refineries	0	0	-15868	5701	969	1086		0	0	0	0	0	0	0	0	-201
	Total Conversion	-6381	-2692	-15868	5647	969	533	(...)	-4455	-636	-503	-264	-68	-18	1198	5791	-9782
	FINAL																
RSD	Residential	232	2064		862	73	190		0	895	0	0	50	0	433	1436	6254
COM	Commercial	37	701		369	2	32		0	39	0	0	8	1	127	1264	2603
IND	Industry	1233	1775		299	36	143		0	541	0	0	0	59	317	2044	6976
AGR	Agriculture	29	80		367	0	16		0	47	0	0	0	0	8	10	573
TRA	Transport	0	8		3856	1048	94		0	121	0	0	0	0	0	133	7688
OTH	Other	773	0		0	0	0		0	0	0	0	0	0	314	325	1412
NEN	Non Energy	34	254		76	5	200		0	0	0	0	0	0	0	0	2324
BNK	Bunkers	0	0		147	0	0		0	0	0	0	0	0	0	0	1056
TFC	Total Final Consumption	2338	4882		5976	1164	675	(...)	0	1644	0	0	58	59	1198	5211	28886

* For purposes of clarity the energy balance is not presented totally and some columns are missing (for refined products).

Figure 125. Disaggregated energy balance at start year 2005 for REG1 – Covered in DemoS_008

3.8.2.2 Pri_COA, Pri_GAS, Pri_OIL, Pri_PP, Con_REF

The structure of these sheets have not changed, but the data is updated following a different commodity split between REG1 and REG2 in the energy balance.

3.8.2.3 Pri_RNW and Pri_NUC

Mining processes for the uranium resources and the new renewable potentials are characterized with a cost (Figure 126).

		~FI_T		
TechName	Comm-IN	Comm-OUT	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity	Cost	Annual Production Bound
*Units			M€2005/PJ	PJ
MINNUC1		NUC	0.25	

		~FI_T		
TechName	Comm-IN	Comm-OUT	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity	Cost	Annual Production Bound
*Units			M€2005/PJ	PJ
MINBIO1		BIO	4.05	
MINHYD1		HYD		
MINWIN1		WIN		
MINSOL1		SOL		

Figure 126. Description of new supply options for renewables

3.8.2.4 Pri_ELC

This sheet is created to capture the imports and exports of electricity (Figure 127). In the default process table, the operational level of these processes are declared as DAYNITE in the **Tslvl** column. Note that the ELC commodity is not declared in the default commodity table as it is already declared in the ELC B-Y Templates. Commodities need to be declared only once and then are available for all files (not only B-Y Templates).

		~FI_T		
TechName	Comm-IN	Comm-OUT	COST	ACT_BND
*Technology Name	Input Commodity	Output Commodity	Cost	Annual Production Bound
*Units			M€2005/PJ	PJ
IMPELC1		ELC	5.38	584
EXPELC1	ELC		6.00	563

Figure 127. Imports and exports options for electricity

3.8.3 B-Y Template VT_REG*_ELC_V08

3.8.3.1 Con_ELC

New power plants are added for each type of renewable energy (Figure 128) using the same approach as before. Their contribution to peak varies depending on the resources: 50% for hydro, 30% for wind, and 20% for solar. However, there is no emission coefficient associated with process anymore (in ~FI_T tables). All combustion emissions are tracked in a uniform manner at the sector level in a ~COMEMI table.

		~FI_T										
TechName	**Comm-IN**	**Comm-OUT**	**STOCK**	**STOCK~2030**	**EFF**	**AFA**	**INVCOST**	**FIXOM**	**VAROM**	**LIFE**	**CAP2ACT**	**Peak**
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Retirement Capacity	Efficiency	Utilisation Factor	Invesctment Cost	Fixed O&M Cost	Variable O&M Cost	Lifetime	Capacity to Activity Factor	% contribution to PEAK
*Units			GW	GW			M€2005/GW	M€2005/GW	M€2005/PJ	Years	PJ/GW	
ELCTECOA00	ELCCOA	ELC	89		0.38	0.85		40.00	0.50	30	31.536	1.00
ELCTEGAS00	ELCGAS	ELC	41		0.49	0.85		35.00	0.40	20	31.536	1.00
ELCTEOIL00	ELCOIL	ELC	6		0.25	0.85		20.00	0.20	30	31.536	1.00
ELCNENUC00	ELCNUC	ELC	52	52	0.33	0.90		38.00	0.27		31.536	1.00
ELCREBIO00	ELCBIO	ELC	8		0.28	0.60		25	0.35	25	31.536	1.00
ELCREHYD00	ELCHYD	ELC	32		1.00	0.50		50	2.00	50	31.536	0.50
ELCREWIN00	ELCWIN	ELC	24		1.00	0.35		35	0.50	20	31.536	0.30
ELCRESOL00	ELCSOL	ELC	7		1.00	0.30		60		15	31.536	0.20

Figure 128. New power plants for renewable electricity generation

3.8.3.2 Emi

A similar sheet is added in all sectors with a ~COMEMI table used to define fuel-based emission coefficients associated with fuel consumption in each sector (Figure 129).

~COMEMI			
CommName	**ELCCOA**	**ELCGAS**	**ELCOIL**
*Units	kt/PJ	kt/PJ	kt/PJ
ELCCO2	95.00	56.10	76.40

Figure 129. Combustion emissions from the electricity sector

3.8.4 BY Template VT_REG*_IND_V08

3.8.4.1 DemTechs_IND

The energy consumed in the industrial sector is captured through a single generic process (Figure 130) consuming the mix of industrial fuels as given in the energy balance and producing one end-use demand (DIDM1). A relaxation factor is used for the maximum input shares in 2050 to give more flexibility to the model over time to optimize the fuel mix. However, the value of the relaxation factor should remain realistic since most fuel switches involve process switches as well.

					Relaxation factor			
		~FI_T			20%			
TechName	**Comm-IN**	**Comm-OUT**	**STOCK**	**Share-I~UP**	**Share-I~2050~UP**	**EFF**	**AFA**	**LIFE**
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Input Share	Input Share	Efficiency	Utilisation Factor	Lifetime
*Units			PJa					Years
IDM1ETOT	INDCOA	DIDM1		19%	22%	1.00	0.95	30
	INDGAS			27%	32%			
	INDOIL			15%	18%			
	INDBIO			8%	10%			
	INDELC			31%	37%			

Figure 130. Multiple input shares process in the industrial sector

3.8.4.2 Emi

An emission commodity is created (Figure 131) and a ~COMEMI table is added in the Emi sheet to track all fuel-based emissions from the sector.

~FI_Comm								
Csets	Region	CommNam	CommDesc	Unit	LimType	CTSLvl	PeakTS	Ctype
*Commodity Set Membership	Region Name	Commodity Name	Commodity Description	Unit	Sense of the Balance EQN.	Timeslice Level	Peak Monitoring	Electricity Indicator
DEM		DIDM1	Demand Industry Sector - Demand 1	PJ				
ENV		INDCO2	Industry Carbon dioxide	kt				

Figure 131. New environmental commodity for industrial emissions

3.8.5 BY Template VT_REG*_RCA_V08

This B-Y Template includes the information related to three sectors: agriculture, commercial and residential.

3.8.5.1 DemTechs_AGR

The energy consumed in the agriculture sector is captured through a single generic process (as for the industrial sector) consuming the mix of agriculture fuels as given in the energy balance and producing one end-use demand (DAOT). A relaxation factor is also used for the maximum input shares in 2050 to give more flexibility to the model over time to optimize the fuel mix. However, the value of the relaxation factor should remain realistic since most fuel switches involve process switches as well.

3.8.5.2 DemTechs_RSD and DemTechs_COM

The energy consumed in the commercial and the residential sectors is modelled through specific processes (Figure 132). Multiple processes are in competition to satisfy each end-use demand (e.g., RSHE* to satisfy the DRSH demand). The existing processes are characterized with their existing installed capacity (STOCK) corresponding in this case to the energy consumption required to produce these energy services as given by the energy balance and the additional fuel split assumptions. The calculation of the existing stocks also takes into account availability factors (AFA) and are converted into GW using a capacity to activity factor (PRC_CAPACT equivalent to CAP2ACT). They also have an efficiency (EFF) and a life time (LIFE).

		~FI_T							
TechName	Comm-IN	Comm-OUT	STOCK	EFF	AFA	PRC_CAPACT	LIFE		Demand
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Utilisation Factor	Capacity to Activity Factor	Lifetime		Production
*Units			GW			Capacity Unit/PJ	Years		
RSHECOA	RSDCOA	DRSH	0	0.85	0.30	31.536	15		0
RSHEGAS	RSDGAS	DRSH	165	0.90	0.30	31.536	15		1563
RSHEOIL	RSDOIL	DRSH	92	0.80	0.30	31.536	15		867
RSHEBIO	RSDBIO	DRSH	86	0.80	0.30	31.536	15		814
RSHESOL	RSDSOL	DRSH	5	1.00	0.30	31.536	15		51
RSHEELC	RSDELC	DRSH	15	0.95	0.30	31.536	15		145
RAPEELC	RSDELC	DRAP	4351	1.00	0.30	1.00	5		1305
ROTECOA	RSDCOA	DROT	780	1.00	0.30	1.00	10		234
ROTEGAS	RSDGAS	DROT	1737	1.00	0.30	1.00	10		521
ROTEOIL	RSDOIL	DROT	963	1.00	0.30	1.00	10		289
ROTEBIO	RSDBIO	DROT	301	1.00	0.30	1.00	10		90
ROTEELC	RSDELC	DROT	0	1.00	0.30	1.00	10		0

Figure 132. Existing processes in the residential sector

3.8.5.3 Demands

The demand table includes all end-use demands for energy services from the three sectors (Figure 133). The values come from the process sheets where the values are already computed in the pink column (Figure 132): STOCK*AFA*PRC_CAPACT. This sheet also includes the fractional shares of each end-use demand by time slice (Figure 134). These shares are relevant to capture the annual variation in the electricity (ELC) consumption levels and prices, the only commodity tracked at the time slice level. In this example, the annual variations are significant for those end-use demands affected by seasonal changes (e.g. space heating).

	~FI_T		
Attribute	CommName	*Unit	2005
*	Demand Commodity Name	Demand Unit	Demand Value
*Units			PJ
Demand	DRSH	PJ	3440
Demand	DRAP	PJ	1305
Demand	DROT	PJ	1135
Demand	DCSH	PJ	904
Demand	DCAP	PJ	1149
Demand	DCOT	PJ	447
Demand	DAOT	PJ	565

Figure 133. Demand for energy services in the RCA sectors

		~FI_T	
Attribute	CommName	Time:	2005
*	Demand Commodity Name		
*Units			
COM_FR	DRSH	SD	0.00
COM_FR	DRSH	SN	0.00
COM_FR	DRSH	WD	0.60
COM_FR	DRSH	WN	0.40
COM_FR	DRAP	SD	0.30
COM_FR	DRAP	SN	0.25
COM_FR	DRAP	WD	0.20
COM_FR	DRAP	WN	0.20
COM_FR	DROT	SD	0.25
COM_FR	DROT	SN	0.25
COM_FR	DROT	WD	0.25
COM_FR	DROT	WN	0.25
COM_FR	DCSH	SD	0.10
COM_FR	DCSH	SN	0.10
COM_FR	DCSH	WD	0.40
COM_FR	DCSH	WN	0.40
COM_FR	DCAP	SD	0.25
COM_FR	DCAP	SN	0.25
COM_FR	DCAP	WD	0.25
COM_FR	DCAP	WN	0.25
COM_FR	DCOT	SD	0.25
COM_FR	DCOT	SN	0.25
COM_FR	DCOT	WD	0.25
COM_FR	DCOT	WN	0.25

Figure 134. Fractional shares for RCA energy service demands

3.8.5.4 Emi

An emission commodity is created in all three sectors and three ~COMEMI tables are added in the Emi sheet to track all fuel-based emissions from each of the three sectors.

3.8.6 BY Template VT_REG*_TRA_V08

3.8.6.1 DemTechs_TRA

The energy consumed in the transportation sector is disaggregated into two end-use demands: transportation by cars and public transport. Consequently, more existing processes are included to satisfy the demand for the new public transport demand, and they are modelled using the same approach as for cars (Figure 135).

		~FI_T								
TechName	**Comm-IN**	**Comm-OUT**	**STOCK**	**EFF**	**AFA**	**ACTFLO~DEMO**	**FIXOM**	**LIFE**	**CAP2ACT**	CALIBRATION
*Technology Name	Input Commodity	Output Commodity	Existing Installed Capacity	Efficiency	Utilisation Factor	Activity to Flo	Fixed O&M Cost	Lifetime	Capacity to Activity Factor	Demand
*Units			000_Units	MVkm/PJ	'000 km	PassengerCar	M$2005/000_Units	Years		kPass*km
TCAREGAS	TRAGAS	DTCAR	233	0.38	14	1.25	0.16	10	0.001	4
TCAREDSL	TRADSL	DTCAR	87103	0.41	17	1.25	0.16	10	0.001	1796
TCARELPG	TRALPG	DTCAR	2583	0.38	14	1.25	0.16	10	0.001	45
TCAREGSL	TRAGSL	DTCAR	84110	0.40	12	1.25	0.15	10	0.001	1209
TCAREBIO	TRABIO	DTCAR	3182	0.40	12	1.25	0.15	10	0.001	46
TCAREELC	TRAELC	DTCAR	0	0.40	12	1.25	0.15	10	0.001	0
TPUBEGAS	TRAGAS	DTPUB	0	0.10	50	15.00	0.24	30	0.001	0
TPUBEDSL	TRADSL	DTPUB	1168	0.15	50	15.00	0.24	30	0.001	876
TPUBELPG	TRALPG	DTPUB	0	0.10	50	15.00	0.24	30	0.001	0
TPUBEGSL	TRAGSL	DTPUB	0	0.15	20	15.00	0.23	30	0.001	0
TPUBEBIO	TRABIO	DTPUB	229	0.15	20	15.00	0.23	30	0.001	69
TPUBEELC	TRAELC	DTPUB	40	0.03	100	200.00	0.23	30	0.001	806

Figure 135. Existing processes in the transportation sector

3.8.6.2 Demands

The demand table includes both end-use demands (in Bpass-km) and the fractional shares of each end-use demand by time slice.

3.8.6.3 Emi

An emission commodity is created and a ~COMEMI table is added in the Emi sheet to track all fuel-based emissions from the sector.

3.8.7 SubRES_NewTechs

The structure of this file has not changed; this is a repository of new processes available for all the regions. The file includes one sheet for each sector: ELC, PRI, IND, RCA, TRA. (The sheet's names have changed and reflect each new sector's name).

The new process repository is completed with more new processes similarly as for the existing processes in the B-Y Templates, namely more processes for renewable power generation, public transport, and more energy services in the residential and commercial sectors (Figure 136).

3.8.7.1 IEA-ETSAP_ETechDS

This sheet contains a reference to the technology briefs (E-TechDS – Energy Technology Data Source) coordinated by the ETSAP-IEA. They are classified into two main categories:

energy supply technologies and energy demand technologies. They provide relevant data on the most important technical and economic attributes of numerous types of technologies.[3]

		~FI_T							
TechName	Comm-IN	Comm-OUT	START	EFF	AFA	CAP2ACT	INVCOST	FIXOM	LIFE
*Technology Name	Input Commodity	Output Commodity		Efficiency	Utilisation Factor	Capacity to Activity Factor	Invesctment Cost	Fixed O&M Cost	Lifetime
*Units						Capacity Unit/PJ	M€/Capacity Unit	M€/Capacity unit	Years
CSHNCOA1	COMCOA	DCSH	2006	0.88	0.30	31.536	400	10	20
CSHNGAS1	COMGAS	DCSH	2006	0.93	0.30	31.536	300	8	20
CSHNOIL1	COMOIL	DCSH	2006	0.83	0.30	31.536	250	6	20
CSHNBIO1	COMBIO	DCSH	2006	0.80	0.30	31.536	750	13	20
CSHNSOL1	COMSOL	DCSH	2006	1.00	0.30	31.536	1000	20	15
CSHNELC1	COMELC	DCSH	2006	0.96	0.30	31.536	400	10	20
CAPNELC1	COMELC	DCAP	2006	1.02	0.30	1.00	0.50		5
COTNCOA1	COMCOA	DCOT	2006	1.02	0.30	1.00	1.00		15
COTNGAS1	COMGAS	DCOT	2006	1.05	0.30	1.00	0.50		15
COTNOIL1	COMOIL	DCOT	2006	1.02	0.30	1.00	0.30		15
COTNBIO1	COMBIO	DCOT	2006	1.02	0.30	1.00	1.00		15
COTNELC1	COMELC	DCOT	2006	1.05	0.30	1.00	0.75		15
RSHNCOA1	RSDCOA	DRSH	2006	0.88	0.30	31.536	400	10	20
RSHNGAS1	RSDGAS	DRSH	2006	0.93	0.30	31.536	300	8	20
RSHNOIL1	RSDOIL	DRSH	2006	0.83	0.30	31.536	250	6	20
RSHNBIO1	RSDBIO	DRSH	2006	0.80	0.30	31.536	750	13	20
RSHNSOL1	RSDSOL	DRSH	2006	1.00	0.30	31.536	1000	20	15
RSHNELC1	RSDELC	DRSH	2006	0.96	0.30	31.536	400	10	20
RAPNELC1	RSDELC	DRAP	2006	1.02	0.30	1.00	0.50		5
ROTNCOA1	RSDCOA	DROT	2006	1.02	0.30	1.00	1.00		15
ROTNGAS1	RSDGAS	DROT	2006	1.05	0.30	1.00	0.50		15
ROTNOIL1	RSDOIL	DROT	2006	1.02	0.30	1.00	0.30		15
ROTNBIO1	RSDBIO	DROT	2006	1.02	0.30	1.00	1.00		15
ROTNELC1	RSDELC	DROT	2006	1.05	0.30	1.00	0.75		15

Figure 136. New processes in the residential and commercial sectors

3.8.8 Scenario files

3.8.8.1 Scen_UC_CO2BND

This user constraint is updated to introduce bounds (limits) on the CO2 emissions from all sectors in each region (REG1 and REG2). These upper bound are calculated as a percentage reduction target from the CO2 emissions (sum in kt) from all the sectors in a reference scenario for 2010 (10%) and 2020 (20%). It is necessary to run the step model without any limit on emissions first to get the reference emission trajectory (run DemoS_008) and to calculate the bounds as a reduction target from the reference emissions.

3.8.8.2 Scen_UC_NUC_MaxCAP

To build this scenario, a ~TFM_FILL table first collects information from the B-Y Templates for REG1 and REG2 (Figure 137): the installed capacity (STOCK) of the nuclear power plant (ELCNENUC00). These data are refreshed each time this file is synchronized (SYNC). Second, a user constraint is built to define an absolute upper limit on the total nuclear capacity by region (Figure 138). In 2015, the maximum capacity is fixed to the 2005 base year levels in both regions. Afterwards the capacity is kept constant for REG1 (using the interpolation rule

[3] http://www.iea-etsap.org/Energy_Technologies/Energy_Technology.asp

15=interpolation migrated at start, forward extrapolation), and in REG2 is limited to an additional 10% of the 2005 base year capacity in 2030 and an additional 50% in 2050.

~TFM_FILL									
Operation_Sum_Avg_Count	Scenario Name	TimeSlice	LimType	Attribute	Year	Other_Indexes	REG1	REG2	Pset_PN
A	BASE			STOCK			51.80	63.31	ELCNENUC00

Figure 137. Grab base information on nuclear power capacity

~UC_Sets: R_E: AllRegions														
~UC_Sets: T_E:														
								~UC_T:UC_RHSRT						
UC_N	Pset_Set	Pset_PN	Pset_CI	Pset_CO	Cset_CN	Attribute	Year	LimType	UC_CAP	REG1	REG2	UC_RHSRT~0	UC_Desc
AU_NUC_MaxCAP	ELE		ELCNUC				2015	UP	1	52	63	15	Max Nuclear Power Plants Capacity
	ELE		ELCNUC				2030	UP	1		70		
	ELE		ELCNUC				2050	UP	1		95		

Figure 138. Declare a maximum capacity for nuclear power plants with a user constraint

3.8.9 Results

The results for the electricity generation capacity (Figure 139) show the respective role of the new types of renewable power (biomass, hydro, wind and solar), the 2050 horizon, as well as the effects of the user constraint on nuclear capacity. Nuclear capacity remains constant for REG1 while it grows in REG2 up to the maximum bound in 2030, but not in 2050.

ELC plants capacity and new capacity

Original Units: GW　Active Unit: GW　Data values filter:

Attribute | *Process* | *Vintage*

~Scenario~	Region	ProcessSet	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_007	REG1	Coal Power Plants	137	160	175	181	168						
		Oil Power Plants	50	51	63	79	95						
		Renewable Power Plants	88	88	88	88	88						
	REG2	Gas Power Plants	104	100	114	144	193						
		Nuclear Power Plants	125	125	125	125	125						
		Oil Power Plants	8	8	7	5	4						
DemoS_008	REG1	Biomass power plants	8	7	6	5	3	2					
		Coal Power Plants	89	86	74	81	84	94	92	86	86	86	86
		Gas Power Plants	41	39	31	21	18	8	8	8	8	8	
		Hydro power plants	32	31	29	26	22	19	16	13	10	6	3
		Nuclear Power Plants	52	52	52	52	52	52	52	52	52	52	52
		Oil Power Plants	6	6	5	4	10	10	11	14	14	14	14
		Solar/PV power plants	7	7	5	2							
		Wind Power Plants	24	23	18	12	6						
	REG2	Biomass power plants	3	2	2	2	1	1					
		Coal Power Plants	48	47	40	42	52	67	62	54	54	54	54
		Gas Power Plants	62	59	47	31	23	7	7	7	7	7	
		Hydro power plants	32	31	29	26	22	19	16	13	10	6	3
		Nuclear Power Plants	63	63	63	63	65	68	70	71	71	71	71
		Oil Power Plants	6	6	5	4	3	2	1				
		Solar/PV power plants	7	7	5	2							
		Wind Power Plants	44	42	33	22	11						

Figure 139. Results - Power generation capacity by fuel type in DemoS_008

The emissions by sector (in Mt) are presented (Figure 140) for both regions, where it is possible to see the contribution of each sector to reaching the reduction targets. In DemoS_008c, with a limit on the total emissions, the additional reductions are coming from the electricity sector (replacing coal-fired with gas-fired power plants), as well as from the residential and the commercial sectors (replacing solid fuels with renewable energies).

Emissions by Sector

Original Units: Kt Active Unit Mt Data values filter:

Attribute *Process* *Vintage* *TimeSlice* CommoditySet *Region*

~-Scenario-~	Commodity	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_008	Agriculture Carbon dioxide	36	36	37	38	39	39	40	41	35	35	43
	Commercial Carbon dioxide	193	246	266	290	314	314	314	314	314	314	314
	Electricity Plants Carbon dioxide	1,049	962	932	956	981	1,045	988	893	889	877	852
	Industry Carbon dioxide	583	586	596	609	622	635	648	661	674	687	700
	Residential Carbon dioxide	563	701	784	887	990	990	990	990	990	990	990
	Transport Carbon dioxide	951	976	994	1,108	1,255	1,255	1,255	1,255	1,255	1,255	1,255
	Total	3,374	3,508	3,609	3,887	4,201	4,279	4,236	4,154	4,158	4,158	4,155
DemoS_008a	Agriculture Carbon dioxide	36	36	37	38	39	39	40	41	35	35	43
	Commercial Carbon dioxide	193	246	266	290	314	314	314	314	314	314	314
	Electricity Plants Carbon dioxide	1,049	962	925	962	866	907	850	755	752	739	804
	Industry Carbon dioxide	583	586	596	609	622	635	648	661	674	687	700
	Residential Carbon dioxide	563	701	784	887	990	990	990	990	990	990	990
	Transport Carbon dioxide	951	976	895	949	1,004	1,004	1,004	1,004	1,004	1,004	1,004
	Total	3,374	3,508	3,502	3,735	3,835	3,890	3,847	3,765	3,769	3,770	3,855
DemoS_008c	Agriculture Carbon dioxide	36	36	37	38	39	39	40	41	42	43	43
	Commercial Carbon dioxide	193	222	231	234	290	308	314	314	304	304	304
	Electricity Plants Carbon dioxide	1,049	962	864	876	784	686	601	480	466	408	376
	Industry Carbon dioxide	583	586	596	609	622	635	648	661	674	687	700
	Residential Carbon dioxide	563	608	665	757	913	926	933	949	936	954	938
	Transport Carbon dioxide	951	972	855	791	714	768	824	884	932	965	1,000
	Total	3,374	3,386	3,248	3,304	3,361	3,361	3,361	3,330	3,354	3,361	3,361

Figure 140. Results – Emissions by sector in DemoS_008

Objective-Function = 19,119,653 M euros (see the _SysCost table in VEDA-BE) with 9,068,703 M euros for REG1 and 10,050,950 M euros for REG2. These costs are again much higher to those computed in the previous step model DemoS_007 because of the expansion of the RES. The total cost is 4% higher with the emission limits for the electricity and the transportation sectors (19,358,261 M euros), and is only slightly reduced by the activation of the elastic demands (19,352,675 M euros). The additional user constraint on nuclear power increases the system cost by 11% (19,699,008 M euros).

3.9 DemoS_009 - SubRES sophistication (CHP, district heating) and Trans files

Description. At the ninth step, the model database is developed further by adding more SubRES with more complex processes. Because SubRES are used to add new processes in different sectors they can be considered as separate modules that can be included in model runs as part of the reference energy system or not. This approach is convenient when different individuals work in parallel on different sectors.

Objective. The objective is to give more examples of possible SubRES including more complex processes: one that introduces iron and steel production in the industrial sector, and one that introduces combined heat and power (CHP) processes, centralised heating plants, and heat exchanger + district heating network. Additional objectives include:

- To show how to use the BY Trans file to move or add data and reduce the size of tables in the B-Y Templates. Here we specify the availability factor by time slice for existing wind and solar processes and add an interpolation rule for new hydro capacity (NCAP_BND).
- To show how to use the transformation file associated with each SubRES to declare the availability or non-availability of each process in each region: new hydro power plants in this example.
- To give an example of a scenario used to insert/update information in the B-Y Templates and SubRES: the demands and the retirement profile for the iron and steel processes.
- To illustrate how to build a user constraint to limit the penetration of some processes, such as the district heating system between 2020 and 2050.

Attributes introduced:	Files updated
PASTI	VT_REG1_ELC_V09
CEH	VT_REG2_ELC_V09
CHPR	BY_Trans
UC_CAP	SubRES_NewTechs_Trans
UC_COMPRD	Files created
UC_FLO	SubRES_New-IND
	SubRES_New-CHP-DH
	Scen_IND_NewRes
	Scen_UC_DH_MinProd

Files. The ninth step model is built:

1. by modifying two B-Y Templates (VT_REG1_ELC_v09, VT_REG2_ELC_v09) to introduce past investment information;
2. by using the BY Transformation file (BY_Trans) to insert base year information (availability factor by time slice for existing wind and solar plants and interpolation rules);
3. by using a SubRES Transformation file (SubRES_NewTechs_Trans) to insert information for new processes (availability factor by time slice for new wind and solar plants) and to declare the availability or non-availability of each process in each region;
4. by building two new SubRES (one with an iron & steel sector; one with CHP processes and district heating);
5. by creating a scenario file to update information in the industrial sector;
6. by creating a scenario file with a user constraint on the minimum penetration of district heating in the residential sector (Figure 141).

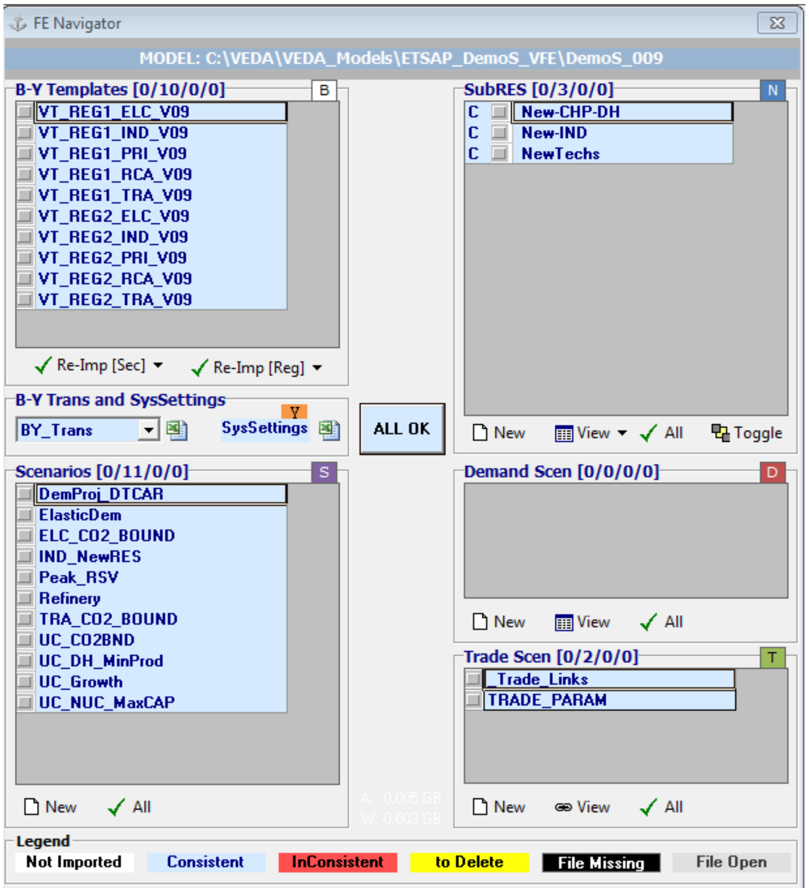


Figure 141. The files included in DemoS_009

3.9.1 B-Y Template VT_REG*_ELC_V09

The only B-Y Templates that are modified are the electricity ones (VT_REG1_ELC_V09 and VT_REG2_ELC_V09).

3.9.1.1 Con_ELC

The STOCK attribute for existing capacity can be replaced by another attribute (PASTI = past investments) to describe capacity installations that took place before the beginning of the model horizon (2005) and still exist during the modelling horizon. For any process, an arbitrary number of past investments may be specified to reflect the age structure in the existing capacity stock: the hydro power plants in this example (Figure 142). Each vintage of PASTI capacity will be constant until the end of its technical life, after which the capacity becomes zero in a single step. This allows a vintage-based retirement profile for the existing stock to be introduced into the model without the need to calculate and specify a STOCK in each future year.

		~FI_T				
TechName	Comm-IN	Comm-OUT	PASTI~1960	PASTI~1980	STOCK	STOCK~2030
*Technology Name	Input Commodity	Output Commodity	Past Investment	Past Investment	Existing Installed Capacity	Retirement Capacity
*Units			GW	GW	GW	GW
ELCTECOA00	ELCCOA	ELC			89	
ELCTEGAS00	ELCGAS	ELC			41	
ELCTEOIL00	ELCOIL	ELC			6	
ELCNENUC00	ELCNUC	ELC			52	52
ELCREBIO00	ELCBIO	ELC			8	
ELCREHYD00	ELCHYD	ELC	15	16		
ELCREWIN00	ELCWIN	ELC			24	
ELCRESOL00	ELCSOL	ELC			7	

Figure 142. Declare past investments that took place before 2005

3.9.1.2 BY_Trans

The BY_Trans file works like a scenario file, except that the rule-based filters and the update/insert changes apply only to those process and commodities already existing in the B-Y templates. In this example (Figure 143), the file is used to insert new information: the availability factor (AF) by time slice (SD, SN, etc.) for existing wind and solar plants (ELCREWIN00 and ELCRESOL00).

~TFM_INS									
TimeSlice	LimType	Attribute	Year	AllRegions	REG1	REG2	Pset_Set	Pset_PN	Pset_PD
SD		AF	2010		0.38	0.35		ELCREWIN00	
SN		AF	2010		0.25	0.30		ELCREWIN00	
WD		AF	2010		0.40	0.45		ELCREWIN00	
WN		AF	2010		0.35	0.25		ELCREWIN00	
SD		AF	2010		0.30	0.35		ELCRESOL00	
SN		AF	2010		0.20	0.15		ELCRESOL00	
WD		AF	2010		0.30	0.25		ELCRESOL00	
WN		AF	2010		0.20	0.15		ELCRESOL00	

Figure 143. Using the transformation file to insert new attributes for existing processes

The transformation file is also used to insert a new interpolation rule (2 = interpolation, but extrapolation with EPS (epsilon, or effectively zero), which inserts EPS in every year if no bound value is declared in any year) to avoid the installation of new capacity (NCAP_BND) after the base year for the existing hydro power plants (ELCREHYD00). VEDA-FE creates this entry by default for all technologies for which STOCK is declared. Since we have switched to using PASTI we need to declare it manually (Figure 144).

~TFM_INS								
TimeSlice	LimType	Attribute	Year	AllRegion	REG1	REG2	Pset_Set	Pset_PN
		NCAP_BND	0		2			ELCREHYD00

Figure 144. Using the transformation file to insert a new interpolation rule

3.9.2 SubRES_NewTechs_Trans

Similarly to the BY_Trans file, a transformation file exists for each of the SubRES created. They are used to update/insert information for new processes and commodities declared in the corresponding SubRES and to declare the availability or non-availability of each process in each region. In this example, the transformation file of the SubRES_NewTechs is used to insert the availability factor for new wind and solar plants (ELCRNWIN01 and ELCRNSOL01) exactly as for the existing ones.

To assign the availability of processes to regions, a new ~TFM_AVA table is created (Figure 145). The first line says that all processes (Pset_PN=*) are available in all regions. The second line modifies this to say that the new hydro power plant is not available in REG1 (1=available; 0=non-available).

~TFM_AVA			
Pset_PN	AllRegions	REG1	REG2
*	1		
ELCRNHYD01		0	

Figure 145. Using the SubRES transformation file to declare process availability

3.9.3 SubRES_New-IND

In the new SubRES_New-IND file, a simplified iron & steel sector is added to the model (Figure 146). This file includes two sheets (IND and PRI); sheet names need to start with the name of one of the model sectors.

Production of Finished Steel		~FI_T					
TechName	Comm-IN	Comm-OUT	Input	Output	Stock	FIXOM	VAROM
*					Mt-y	€/ton-a	
IDMIIS	IISRST		1.00		100	1	0.1
	INDOIL		0.03				
	INDGAS		1.15				
	INDELC		0.60				
		DIIS		1.00			

Production of Raw Steel		~FI_T					
TechName	Comm-IN	Comm-OUT	Input	Output	Stock	FIXOM	VAROM
*					Mt-y	€/ton-a	
ITIISBOF	IISIRO		1.00		100	1	0.1
	INDGAS		0.20				
	INDELC		0.20				
	INDOXY		0.09				
		IISRST		1.00			

Figure 146. Examples of processes in the iron & steel sector

For policy analysis, it is useful to develop the most energy-intensive industrial sectors, such as iron & steel, in more detail, using a process-oriented approach rather than using generic processes capturing the energy mix. Here the demand is expressed in millions tons (Mt) of finished steel production, and a series of processes are modelled to represent the main steps of the transformation chain, from raw material extraction to the production of finished products (with capacity and activity units in Mt). The last process (IDMIIS) is described like a demand process, while the others are described as (upstream) processes in the chain. This means that they consume energy commodities and/or materials to produce new materials useful for the iron &

steel chain production. The last process, which is a demand technology, finally consumes energy commodities and materials produced in the chain to satisfy the iron and steel demand (DIIS).

These processes use a mix of energy inputs and material inputs. These materials are declared as MAT commodities and tracked in Mt (Figure 147).

~FI_Comm

Csets	CommName	CommDesc	Unit	LimType	CTSLvl	PeakTS	Ctype
*Commodity Set Membership	Commodity Name	Commodity Description	Unit	Balance Equ Type Override	Timeslice Tracking Level	Peak Monitoring	Electricity Indicator
MAT	IISSIN	Industrial Sinter	Mt				
	IISIRO	Industrial Iron	Mt				
	IISRST	Industrial Steel	Mt				
DEM	DIIS	Demand Iron&Steel	Mt				

Figure 147. Energy and material input commodities for the iron & steel sector

3.9.4 SubRES_New-CHP-DH

This file includes two sheets (ELC_CHP and RCA), recalling that SubRES sheet names need to start with the name of one of the model sectors. The first sheet is used to add the combined heat and power (CHP) sector to the model (Figure 148). Cogeneration power plants, or combined heat and power plants (CHP), are plants that consume one or more commodities and produce two commodities, electricity (ELC) and heat (HET). The new CHP processes are characterized with additional attributes compared with conventional power plants.

- The new processes do not have an existing installed capacity, but they are available in the database to be invested in. They are characterized with an efficiency (EFF), an annual availability factor (AFA), fixed and variable O&M costs (FIXOM, VAROM), a life time (LIFE), a capacity to activity factor (CAP2ACT in PJ/GW), and an investment cost (INVCOST), as well as the year in which they become available (START). Maximum input shares (Share-I~UP) are also specified for the dual input process ELCBNGAB01 consuming a maximum of 60% of biomass.
- Two new attributes are introduced: the ratio of electricity lost to heat gained (CEH) as well as the ratio of heat produced to electricity produced (CHPR).

Two main types of cogeneration power plants can be distinguished according to the flexibility of the outputs: a back pressure process (ELCBNGAB01) and a condensing process (ELCCNGAS01).

- Back pressure turbines are systems in which the ratio of the production of electricity and heat is fixed, so that the electricity generation is directly proportional to the steam produced. In a real system, a back pressure turbine is defined using the electrical efficiency, the thermal efficiency, and the load utilization. The **CHPR** attribute is then fixed (FX), so the production of electricity and heat is in a fixed proportion, but one could also use a (LO) CHPR for defining the back-pressure point, if so desired (to allow by-passing the turbine to produce more heat). CEH can be either 0 (or missing) or 1:
    - If it is 0 (or missing) as in this example, the activity represents the electricity generation and the capacity represents the electrical capacity;

- If it is 1, the activity represents the total energy output and the capacity represents the total capacity (electricity + heat).
- The condensing pass-out or extraction turbines do not have to produce heat, permitting electricity only to be generated, and permitting the amount of heat generated to be directly adjusted to the heat demand, while the electricity generation is reciprocally proportional to heat generation (electricity losses because of heat extraction). They are thus described differently:
  - 1. Coefficient of electricity to heat, via attribute **CEH** such that: a) <= 1: electricity loss per unit of heat gained (moving from condensing to backpressure mode), indicating that activity is measured in terms of electricity, or b) >= 1: heat loss per unit of electricity gained (moving from backpressure to condensing mode), indicating that activity is measured in terms of total output (electricity plus heat).
  - 2. Efficiencies, according to 1: a) are specified for the condensing point, or b) are specified for backpressure point.
  - 3. Costs, according to 1: a) are specified based according to condensing mode, or b) are specified based on total electricity and heat output at backpressure point.
  - 4. Ratio of heat produced to electricity produced (**CHPR**): Ratio of heat to power at backpressure point; at least a maximum value is required, but in addition also a minimum value may be specified.
- See Section 4.1 of Part II of the TIMES documentation for more on CHP processes and their attributes.

The CHP processes are declared as CHP processes in the process declaration table with a time slice level of activity (DAYNITE). The heat (HET) is also declared as a new energy commodity in the commodity declaration table.

		~FI_T													
TechName	**Comm-IN**	**Comm-OUT**	**START**	**EFF**	**AFA**	**Share-I~UP**	**CEH**	**CHPR ~FX**	**CHPR ~UP**	**INVCOST**	**FIXOM**	**VAROM**	**LIFE**	**CAP2ACT**	**Peak**
*Technology Name	Input Commodity	Output Commodity		Efficiency	Utilisation Factor	Input share				Invesctment Cost	Fixed O&M Cost	Variable O&M Cost	Lifetime	Capacity to Activity Factor (Act Unit/Cap Unit)	% contribution to PEAK
*Units										M€/GW	M€/PJa	M€/PJ	Years		
ELCCNGAS01	ELCGAS	ELC	2015	0.40	0.85		0.20		1.20	950	30.00	0.35	30	31.536	1.00
		HET													
ELCBNGAB01	ELCGAS	ELC	2015	0.40	0.85			1.20		1100	40.00	0.40	30	31.536	1.00
	ELCBIO	HET				0.60									

Figure 148. Examples of combined heat and power processes

The RCA sheet is used to add a district heating option to the model (Figure 149): a process is created as the district heating option (RSHNHET1) and a sector fuel process (FTE-RSDHET) is created to produce sector heat (RSDHET) from primary heat (HET).

- They are characterized with an efficiency (EFF), an annual availability factor (AFA), fixed O&M costs (FIXOM), a life time (LIFE), a capacity to activity factor (CAP2ACT in PJ/GW), and an investment cost (INVCOST), as well as the year in which they become available (START).

		~FI_T								
TechName	**Comm-IN**	**Comm-OUT**	**START**	**EFF**	**AFA**	**CAP2ACT**	**INVCOST**	**FIXOM**	**LIFE**
*Technology Name	Input Commodity	Output Commodity		Efficiency	Utilisation Factor		Invesctment Cost	Fixed O&M Cost	Lifetime
*Units						Capacity Unit/PJ	M€/Capacity Unit	M€/Capacity unit	Years
RSHNHET1	RSDHET	DRSH	2015	0.96	0.30	31.536	250	5	20
FTE-RSDHET	HET	RSDHET	2015	0.95	1.00	1	1000	10	20

Figure 149. Demand for heat and district heating options

3.9.5 Scenario files

3.9.5.1 Scen_IND_NewRES

A transformation table is used to update the base year industrial demand (DIDM1): the base year valued defined in the B-Y Templates are multiplied by 0.9 (Figure 150). This essentially reduces the DIDM1 demand that was used to model all industrial sector energy consumption by an amount roughly corresponding to that consumed by the new iron and steel sector. (Although note that we are not trying to replicate calibration to the energy balance precisely in this simple example.)

Another transformation table is used to define the demand value for the new iron and steel demand (DIIS), activating this sector when the SubRES is included in a model run, and to specify the retirement profile for the iron and steel processes (STOCK in 2050). (In this case the STOCK has been introduced in a SubRES template so VEDA-FE will not create any interpolation rule to prohibit new investments.)

~TFM_UPD																
TimeSlice	**LimType**	**Attribute**	**Year**	**Attrib_Cond**	**Val_Cond**	**AllRegions**	**REG1**	**REG2**	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
		Demand	2005				*0.9	*0.9							DIDM1	
~TFM_INS																
TimeSlice	**LimType**	**Attribute**	**Year**	**Attrib_Cond**	**Val_Cond**	**AllRegions**	**REG1**	**REG2**	Pset_Set	Pset_PN	Pset_PD	Pset_CI	Pset_CO	Cset_Set	Cset_CN	Cset_CD
		Demand	2005				10	12							DIIS	
		STOCK	2050				100	100		*IIS*						

Figure 150. Update existing information and insert new information in the industrial sector

3.9.5.2 Scen_UC_DH_MinProd

A user constraint is built to specify the minimum district heating penetration requirement in specific years (2020 and 2050) with an interpolation/extrapolation rules between those years (rule 15=interpolation migrated at start, forward extrapolation) (Figure 151). The constraint says that the production of **DRSH** by processes that consume **RSDHET** (Pset_CI) must be the minimum (LimType=**LO**) percentage specified in each region/year combination of *all* production (table level declaration **UC_COMPRD**) of DRSH.

~UC_Sets: R_E: AllRegions														
~UC_Sets: T_E:														
								~UC_T: UC_COMPRD						
UC_N	**Pset_Set**	**Pset_PN**	**Pset_CI**	**Pset_CO**	**Cset_CN**	**Attribute**	**Year**	**LimType**	**UC_FLO**	REG1	REG2	UC_RHSRTS	UC_RHSRTS~0	**UC_Desc**
UC_DH_LOW	DMD		RSDHET		DRSH		2020	LO	1	-5%	-10%	0	15	Minimum district heating penetration
	DMD		RSDHET		DRSH		2050	LO	1	-10%				

Figure 151. Minimum district heating penetration using a user constraint

3.9.6 Results

The model variant DemoS_009d is solved with the new iron & steel sector. Figure 152 shows the demand production (DIIS in Mt) from the finished steel production process (IDMIIS), consuming industrial steel (IISRST in Mt) and a mix of energy in PJ.

The model variant DemoS_009e is solved with the new district heating option. Figure 153 shows the contribution of district heat in meeting the demand for residential space heating in both regions together.

Indusrial Iron&Steel Demand Technology

Original Units: **Active Unit**

Region | *Vintage* | TimeSlice

~-Scenario-~	Attribute	Process	Commodity	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_009d	VAR_FIn	IDMIIS	IISRST	22	22	22	22	22	22	22	22	22	22	22
			INDELC	13	13	13	13	13	13	13	13	13	13	13
			INDGAS	25	25	25	25	25	25	25	25	25	25	25
			INDOIL	1	1	1	1	1	1	1	1	1	1	1
	VAR_FOut	IDMIIS	DIIS	22	22	22	22	22	22	22	22	22	22	22
			INDCO2	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470	1,470

Figure 152. Results – Finished steel production in DemoS_009

Consumption by Sector and fuel

Original Units: PJ **Active Unit** PJ **Data values filter:**

Attribute | *Process* | *Vintage* | *TimeSlice* | *Region* | *CommoditySet* | -ProcessSet-

~-Scenario-~	Commodity	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_009	Residential Biomass	1,470	2,214	1,912	1,533	1,094	856	856	856	1,094	1,094	1,094
	Residential Natural Gas	5,663	4,498	3,626	2,537	1,517	1,412	1,491	1,346	1,861	2,334	2,373
	Residential Oil	2,760										
	Residential Solar energy	3,024	2,706	2,635	2,597	2,559	2,559	2,559	2,559	2,559	2,559	2,559
	Residential Solid Fuels	360	3,611	4,789	6,261	7,726	8,074	7,991	8,145	7,362	6,863	6,821
	Total	13,277	13,029	12,962	12,928	12,896	12,901	12,897	12,905	12,876	12,849	12,847
DemoS_009c	Residential Biomass	1,470	2,411	2,388	1,935	1,452	1,844	2,099	1,704	1,759	1,832	1,832
	Residential Natural Gas	5,663	4,885	5,177	4,792	4,288	4,512	3,334	3,214	3,214	3,214	4,076
	Residential Oil	2,760										
	Residential Solar energy	3,024	2,706	2,635	2,597	2,559	2,559	2,559	2,559	2,559	2,559	2,559
	Residential Solid Fuels	360	3,007	2,671	3,476	4,439	3,810	4,800	5,322	5,266	5,193	4,283
	Total	13,277	13,009	12,871	12,800	12,738	12,725	12,792	12,799	12,799	12,799	12,750
DemoS_009e	Residential Biomass	1,470	2,321	2,066	1,802	1,178	1,315	1,483	1,373	1,483	1,483	1,483
	Residential Natural Gas	5,663	4,553	4,778	4,431	3,519	3,866	2,923	2,329	2,329	2,329	3,043
	Residential Oil	2,760										
	Residential Solar energy	3,024	2,706	2,635	2,597	2,559	2,559	2,559	2,559	2,559	2,559	2,559
	Residential Solid Fuels	360	3,444	3,416	3,991	4,900	4,364	5,159	5,864	5,721	5,688	4,901
	Residential heat from district heating network					574	604	634	664	693	723	753
	Total	13,277	13,024	12,896	12,821	12,730	12,707	12,758	12,789	12,786	12,784	12,740

Figure 153. Results – Fuel used for residential space heating in DemoS_009

Objective-Function = 19,183,729 M euros (see the _SysCost table in VEDA-BE) with 9,084,193 M euros for REG1 and 10,099,536 M euros for REG2. These costs are similar to those computed with the previous step model DemoS_008. The total cost is 3% higher with the emission limits, growth rates, elastic demands, and the new iron and steel sector (19,721,879 M euros) and 5% with the new district heating option (20,187,883 M euros) and the new investment required to satisfy the minimum constraint on district heating penetration.

3.10 DemoS_010 - Demand projections and elastic demand

Description. At the tenth step, the model structure and database remain the same but energy service demands are projected using an internal VEDA-FE routine.

Objective. The objective is to show how to prepare the files required to automatically project end-use demands for energy services using demand drivers along with sensitivity and calibration series.

Attributes introduced:	Files updated
N.A.	Scen_ElasticDem Files created Dem_Alloc+Series ScenDem_DEM_Ref

Files. The tenth step model is built:

1. by creating one file that allocates a demand driver to each end-use demand (Dem_Alloc+Series) and defines sensitivity and calibration series, and one file (ScenDem_DEM_Ref) that defines demand drivers;
2. by modifying the elastic demand scenarios to cover all end-use demands for energy services (Figure 154).

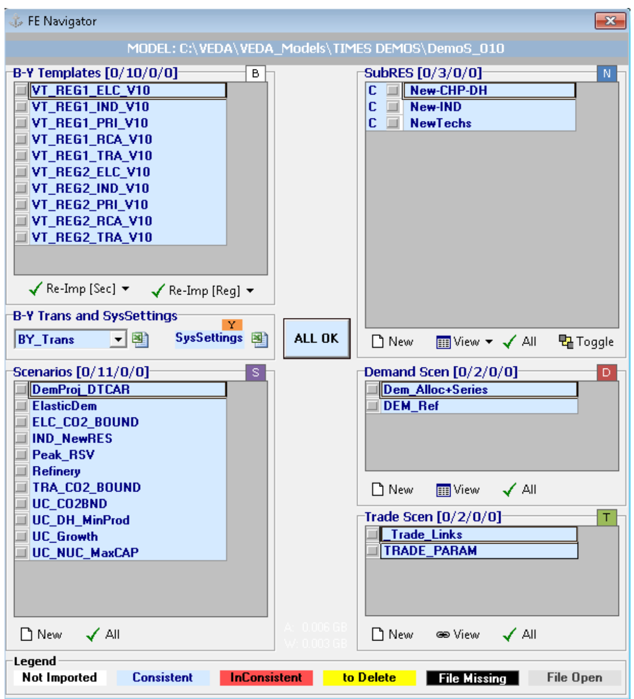


Figure 154. The files included in DemoS_010

3.10.1 Demand files

3.10.1.1 ScenDem_DEM_Ref

The ~DRVR_Table table is used to declare a coherent set of driver growth rates (or indexes, with 2005=1) to drive all end-use demands in all regions (Figure 155). These drivers can be more general, such as macroeconomic indicators, as in this example (Gross Domestic Product (GDP), population (POP), industrial output demand (INDD)), or more specific, like vehicle-kilometres for energy service demands in the transportation sector, for instance. It is possible to build multiple files called ScenDem_<file name> with different drivers to generate, for example, a reference case along with low and high growth cases.

	~DRVR_Scenario::DEM_Ref											
	~DRVR_Table											
Region	Driver	\~2005	\~2006	\~2007	\~2008	\~2009	\~2010	\~2011	(...)	\~2048	\~2049	\~2050
REG1	GDP	1.00	1.03	1.06	1.09	1.13	1.16	1.19	(...)	2.03	2.04	2.05
REG1	POP	1.00	1.01	1.02	1.03	1.04	1.05	1.06	(...)	1.37	1.38	1.38
REG1	GDPP	1.00	1.02	1.04	1.06	1.08	1.10	1.12	(...)	1.66	1.67	1.68
REG1	INDD	1.00	1.00	1.00	1.00	1.00	1.00	1.00	(...)	1.02	1.02	1.02
REG2	GDP	1.00	1.05	1.10	1.16	1.22	1.28	1.34	(...)	2.98	2.99	3.01
REG2	POP	1.00	1.02	1.04	1.06	1.08	1.10	1.13	(...)	1.67	1.67	1.68
REG2	INDD	1.00	1.00	1.00	1.00	1.00	1.01	1.01	(...)	1.04	1.04	1.05

Figure 155. Demand drivers for end-use demand projections

3.10.1.2 Dem_Alloc+Series

The ~Series table is used to define sensitivity and calibration series (Figure 156). The sensitivity series represents the sensitivity of each end-use demand to one unit change in its driver. The calibration series can optionally be used to provide additional control over the resulting demand levels.

The growth rates of the various drivers are applied to the 2005 base year demands using the following formula:

$$D_t = D_{t-1} * \left(Calibration + \left(\frac{Driver_t}{Driver_{t-1}} - 1 \right) * Sensitivity \right)$$

The ~DRVR_Allocation table is used to allocate a particular driver to each end-use demand in each region (Figure 157). Only one such allocation file, always named Dem_Alloc+Series, may be built. That is, it is envisioned that in different scenarios, the projection of the driver for each demand may change (higher or lower population growth, for example), but the association of each demand with a particular driver will not change. (For example, DRSH is always driven by population growth with the same sensitivity.) Only one driver series may be associated with each demand. However, one may easily create a composite series if combining two drivers is desired. In this example, the demand DAOT will be projected using the driver GDP, adjusted with calibration and sensitivity series (Constant; =1 over the whole model horizon).

~Series											
Series	\~2005	\~2006	\~2007	\~2008	\~2009	\~2010	\~2011	\~2012	\~2013	\~2014	\~2015	(...)
Constant	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	(...)
Ser_0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	(...)

Figure 156. Sensitivity and calibration series for end-use demand projections

~DRVR_Allocation

Region	Demand	Driver	Calibration	Sensitivity
REG1	DAOT	GDP	Constant	Constant
REG1	DCAP	GDP	Constant	Constant
REG1	DCOT	GDPP	Constant	Constant
REG1	DCSH	GDP	Constant	Constant
REG1	DIDM1	INDD	Constant	Constant
REG1	DRAP	POP	Constant	Constant
REG1	DROT	NewDriver	Constant	Constant
REG1	DRSH	POP	Constant	Constant
REG1	DTCAR	GDP	Constant	Constant
REG1	DTPUB	POP	Constant	Constant
REG2	DAOT	GDP	Constant	Constant
REG2	DCAP	GDP	Constant	Constant
REG2	DCOT	GDPP	Constant	Constant
REG2	DCSH	GDP	Constant	Constant
REG2	DIDM1	INDD	Constant	Constant
REG2	DRAP	POP	Constant	Constant
REG2	DROT	NewDriver	Constant	Constant
REG2	DRSH	POP	Constant	Constant
REG2	DTCAR	GDP	Constant	Constant
REG2	DTPUB	POP	Constant	Constant

Figure 157. Allocation of demand drivers and series for end-use demand projections

All the demands projected with the internal VEDA-FE module can also be managed from the menu: **Advanced Functions/Demand Master**. Changes made within the Demand Master will be reflected in the templates. For more information on the Demand Master function, see http://support.kanors-emr.org/.

3.10.2 Results

The resulting demand projections in the reference case (DemoS_010) using the driver and series allocation presented above are shown (Figure 158), as well as the demand reactions when including all additional constraints (limits on emissions, growth rates of cars, minimum penetration of district heating, etc.).

Demands

Original Units: **Active Unit** **Data values filter:**

Attribute | *Process* | *Vintage* | TimeSlice | CommoditySet | *Region*

Commodity	~-Scenario-~	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DAOT	DemoS_010	1,125	1,170	1,370	1,672	2,045	2,507	2,570	2,635	2,702	2,770	2,840
	DemoS_010e	1,125	1,170	1,329	1,599	2,014	2,349	2,393	2,487	2,544	2,624	2,669
DCAP	DemoS_010	2,297	2,389	2,798	3,415	4,178	5,123	5,252	5,384	5,520	5,660	5,803
	DemoS_010e	2,297	2,389	2,714	3,266	4,093	4,815	4,937	5,103	5,272	5,363	5,455
DCOT	DemoS_010	733	741	779	829	885	947	964	981	999	1,017	1,035
	DemoS_010e	733	735	749	789	867	888	903	929	934	959	968
DCSH	DemoS_010	2,280	2,376	2,805	3,457	4,270	5,285	5,418	5,555	5,695	5,839	5,986
	DemoS_010e	2,280	2,376	2,686	3,280	4,185	4,833	4,983	5,210	5,197	5,379	5,572
DIDM1	DemoS_010	13,159	13,169	13,209	13,258	13,308	13,358	13,408	13,459	13,509	13,560	13,611
	DemoS_010e	11,843	13,169	12,812	12,661	13,009	12,456	12,403	12,739	12,597	12,848	12,794
DIIS	DemoS_010e	22	22	22	22	22	22	22	22	22	22	22
DRAP	DemoS_010	2,610	2,650	2,813	3,033	3,272	3,532	3,621	3,713	3,806	3,903	4,001
	DemoS_010e	2,610	2,650	2,728	2,899	3,226	3,320	3,404	3,542	3,635	3,698	3,761
DROT	DemoS_010	2,250	2,250	2,250	2,250	2,250	2,250	2,250	2,250	2,250	2,250	2,250
	DemoS_010e	2,250	2,233	2,182	2,165	2,216	2,148	2,148	2,165	2,148	2,161	2,148
DRSH	DemoS_010	7,230	7,341	7,800	8,420	9,095	9,830	10,078	10,333	10,594	10,861	11,135
	DemoS_010e	7,230	7,341	7,458	7,940	7,731	8,355	8,566	8,493	8,210	8,010	7,797
DTCAR	DemoS_010	6,168	6,356	7,018	7,947	9,004	13,738	14,085	14,440	14,805	15,179	15,562
	DemoS_010e	6,168	6,356	7,018	7,827	8,869	13,532	13,787	14,170	14,472	14,780	15,095
DTPUB	DemoS_010	3,479	3,531	3,748	4,040	4,358	4,704	4,823	4,945	5,069	5,197	5,329
	DemoS_010e	3,479	3,531	3,692	3,951	4,328	4,563	4,606	4,759	4,904	4,999	5,169

Figure 158. Results – Demand projections in DemoS_010

Objective-Function = 24,831,217 M euros (see the _SysCost table in VEDA-BE) with 10,869,234 M euros for REG1 and 13,961,983 M euros for REG2. The total cost is 7% higher with all model variants (26,475,198 M euros).

3.11 DemoS_011 - Linking input templates and VEDA-BE sets

Description. At the eleventh step, the model structure and database still remain the same but process and commodity sets defined in VEDA-BE are linked with the VEDA-FE model.

Objective. The objective is to show how to link VEDA-BE sets with VEDA-FE models. It is possible to create sets of commodities and processes in VEDA-BE using filters and rules. These sets are generally used to build tables to view results in VEDA-BE, but it is also possible to link the VEDA-BE database with a VEDA-FE model to use these sets in VEDA templates. We also provide an example in which VEDA-BE sets are used in the VEDA-FE database: a user constraint on the minimum penetration of renewable power plants is built using a user defined set of renewable processes. (See Part V of the TIMES documentation for more on creating VEDA-BE sets and using them to view results within VEDA-BE.)

Attributes introduced:	Files created
N.A.	Scen_BOUNDS-UC_WSETS

Files. The eleventh step model is built:

1. by creating one scenario file that explains VEDA Sets specification and includes a user constraint using a VEDA-BE table.

3.11.1 Scen_Bounds-UC-wSets

The new scenario file contains two sheets: one that explains how to access VEDA-BE sets within VEDA-FE, and one that includes a user constraint using a VEDA-BE table (UC_Set). The steps to link the VEDA-BE database with a VEDA-FE database are:

- Click on **Veda_SnT.MDB** in the main menu bar of VEDA-FE.
- In the Options window, File Locations tab, click **Veda-BE Database** button (Figure 159), and locate the path where the VEDA-BE database is stored (e.g. C:\VEDA\Veda_BE\Databases\DemoS_VBE), and click **OK**.
    - The selected Veda_SnT.MDB path will now appear in the main menu bar of VEDA-FE (e.g. C:\VEDA\Veda_BE\Databases\DemoS_VBE\Veda_SnT.MDB).
- VEDA-BE sets will now appear in the VEDA-FE browser (See Section 2.4.1). The sets of commodities and processes can be viewed in the dropdown menus below the main Process and Commodity boxes. Selecting one of the sets will change the Process or Commodity list to show only the processes or the commodities included in this set: the PP_RENEW set that includes all renewable power plants in this example (Figure 160). Any of these sets can now be used directly in VEDA-FE files to insert or update information for a group of processes or commodities.

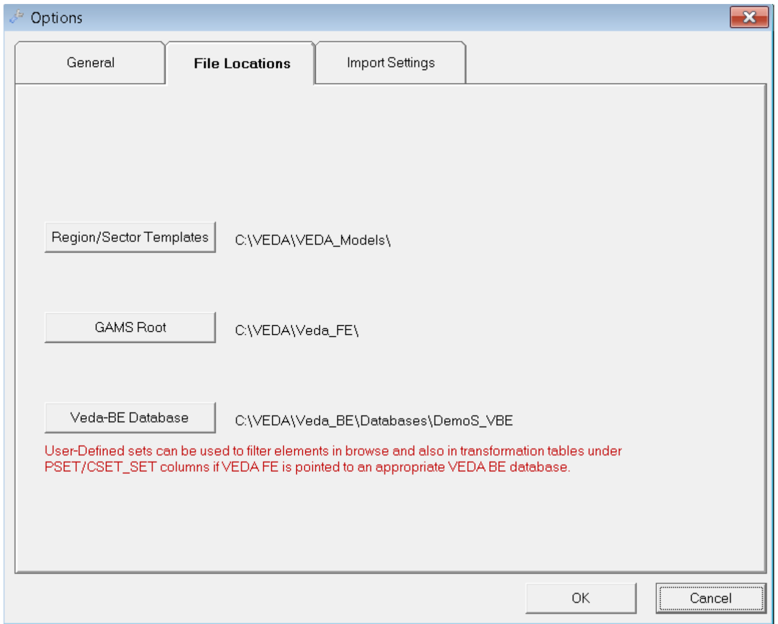


Figure 159. Create the link to VEDA-BE databases in VEDA-FE

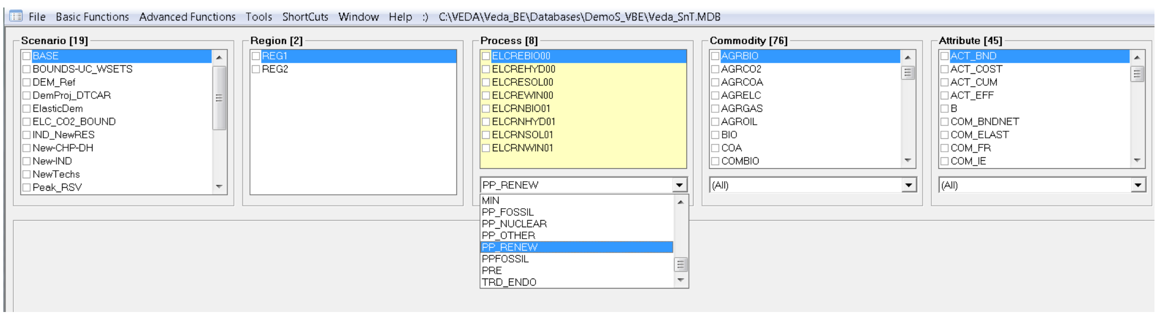


Figure 160. VEDA-BE sets in the VEDA-FE browser

As an example, a user constraint is built using the process set PP_RENEW (column PSet_SET) that includes all renewable power plants: it specifies a minimum renewable penetration share of 10% in 2020 and 15%-20% in 2050, depending on the region, along with an interpolation/extrapolation rule (Figure 161).

~UC_Sets: R_E: AllRegions														
~UC_Sets: T_E:														
								~UC_T: UC_COMPRD						
UC_N	Pset_Set	Pset_PN	Pset_CI	Pset_CO	Cset_CN	Attribute	Year	LimType	UC_FLO	REG1	REG2	UC_RHSRTS	UC_RHSRTS~0	UC_Desc
UC_RNW-PP_LOW	PP_RENEW				ELC		2020	LO	1	-10%	-10%	0	15	Minimum renewable power plants penetration
	PP_RENEW				ELC		2050	LO	1	-20%	-15%			

Figure 161. User constraint on renewable power using a VEDA-BE set

3.11.2 Results

Figure 162 shows the impact of the new user constraint on the renewable share of total power generation. While the share of renewables is going to 0 without the user constraint in the previous reference case (DemoS_010), it reaches 18% across both regions in 2050 in the new reference case (DemoS_011), and 20% when including all additional constraints (limits on emissions, growth rates of cars, minimum penetration of district heating, etc.).

Electric Generation by Fuel Group

Original Units: PJ Active Unit PJ Data values filter:

Attribute | Commodity | *Process* | *Vintage* | *TimeSlice* | CommoditySet | *Region*

Period

~Scenario~	ProcessSet	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_010	Fossil Power Plants	4,966	4,492	5,700	6,988	8,230	9,652	9,918	9,808	9,666	9,528	9,395
	Nuclear Power Plants	3,267	3,267	3,267	3,267	3,330	3,393	3,457	3,828	4,018	4,207	4,397
	Renewable Power Plants	1,818	1,774	1,368	936	688	505	217				
	Total	10,051	9,533	10,335	11,191	12,248	13,550	13,591	13,636	13,684	13,735	13,792
DemoS_010e	Fossil Power Plants	4,549	4,517	5,286	5,918	7,344	7,570	7,530	7,597	7,447	6,861	6,538
	Nuclear Power Plants	3,267	3,267	3,267	3,267	3,330	3,393	3,457	3,828	4,018	4,207	4,397
	Renewable Power Plants	1,818	1,774	1,472	1,108	811	1,097	1,024	807	807	1,226	1,304
	Total	9,633	9,558	10,024	10,294	11,485	12,060	12,011	12,232	12,271	12,294	12,239
DemoS_011	Fossil Power Plants	4,966	4,492	5,700	6,988	7,576	8,506	8,300	7,780	7,558	7,129	6,812
	Nuclear Power Plants	3,267	3,267	3,267	3,267	3,330	3,393	3,457	3,828	3,920	4,207	4,397
	Renewable Power Plants	1,818	1,774	1,368	936	1,342	1,651	1,834	2,034	2,206	2,407	2,582
	Total	10,051	9,533	10,335	11,191	12,248	13,550	13,591	13,642	13,684	13,743	13,792
DemoS_011e	Fossil Power Plants	4,549	4,517	5,222	5,711	6,928	7,117	6,974	6,608	6,394	5,783	5,643
	Nuclear Power Plants	3,267	3,267	3,267	3,267	3,330	3,393	3,457	3,828	4,018	4,207	4,397
	Renewable Power Plants	1,818	1,774	1,580	1,375	1,331	1,734	1,838	1,972	2,193	2,269	2,527
	Total	9,633	9,558	10,069	10,354	11,589	12,244	12,269	12,408	12,605	12,259	12,567

Figure 162. Results – Demand projections in DemoS_011

Objective-Function = 24,867,969 M euros (see the _SysCost table in VEDA-BE) with 10,886,683 M euros for REG1 and 13,981,286 M euros for REG2. The total cost is 6% higher with all model variants (26,483,468 M euros).

3.12 DemoS_012 – More modelling techniques

Description. At the twelfth step, taxes and subsidies are added to the model database and a new modelling technique is introduced, namely the lumpy investment concept.

Objective. The objective is to show how to add taxes and subsidies for processes or commodities, such as a tax on diesel and total CO2 for all sectors and regions, as well as a subsidy on solar power plants in this example. Another objective is to show how to use the lumpy investment feature of TIMES through discrete capacity for the new nuclear power plants.

Attributes introduced:	Files updated
N.A.	VT_REG1_PRI_v12 VT_REG2_PRI_v12 SubRES_NewTechs Files created Scen_TRADSL_Tax Scen_CO2_Tax Scen_Solar_Subsidies Scen_UC_CO2_Regions Scen_NUC_DiscInv

Files. The twelfth step model is built:

1. by updating two B-Y Templates (VT_REG1_PRI_v12, VT_REG2_PRI_v12) to create an aggregated CO2 emission commodity;
2. by updating the SubRES_NewTechs file to specify discrete investment options;
3. by creating scenario files for introducing taxes, subsidies, and an emission constraint for all sectors and regions, as well as for discrete investments for nuclear power plants.

3.12.1 B-Y Template VT_REG*_Pri_V12

The only B-Y Templates that are modified are the primary energy ones (VT_REG1_PRI_V12 and VT_REG2_PRI_V12).

3.12.1.1 TOTCO2

A sheet is added with a **~COMAGG** table is that is used to define an aggregated commodity (TOTCO2), including all sectoral CO2 emissions using multipliers of 1. This is equivalent to making TOTCO2 the sum of all sectoral CO2 emissions (Figure 163). It is possible to add more aggregated commodities and change multipliers. For instance, when there are different types of GHG emissions (CH4, N2O, etc.), an aggregated commodity can be created in CO2-equivalent to account for their respective global warming potential (CH4=36; N2O=298).

~COMAGG						
CommName	AGRCO2	COMCO2	RSDCO2	ELCCO2	TRACO2	INDCO2
TOTCO2	1	1	1	1	1	1

~FI_Comm								
Csets	Region	CommNar	CommDes	Unit	LimType	CTSLvl	PeakTS	Ctype
*Commodity Set Membership	Region Name	Commodity Name	Commodity Description	Unit	Sense of the Balance EQN.	Timeslice Level	Peak Monitoring	Electricity Indicator
ENV		TOTCO2	Total CO2	kt				

Figure 163. Aggregation of emission commodities

3.12.2 SubRES_NewTechs (ELC sheet)

The first step necessary to enable lumpy investments is to specify discrete investment options in the default process table, for new nuclear power plants in this example (ELCNNUC01), by changing the process set from ELC to ELC, DSCINV (Figure 164).

~FI_Process								
Sets	Region	TechName	TechDesc	Tact	Tcap	Tslvl	PrimaryCG	Vintage
*Process Set Membership *	Region Name	Technology Name	Technology Description	Activity Unit	Capacity Unit	TimeSlice level of Process Activity	Primary Commodity Group	Vintage Tracking
ELE		ELCTNCOA01	Power Plants New 1 - Solid Fuels	PJ	GW	SEASON		
		ELCTNOIL01	Power Plants New 1 - Oil	PJ	GW			
		ELCTNGAS01	Power Plants New 1 - Natural Gas	PJ	GW			Yes
		ELCRNBIO01	Power Plants New 1 - Biomass	PJ	GW			
		ELCRNHYD01	Power Plants New 1 - Hydro power	PJ	GW			
		ELCRNWIN01	Power Plants New 1 - Wind energy	PJ	GW			
		ELCRNSOL01	Power Plants New 1 - Solar energy	PJ	GW			
ELE,DSCINV		ELCNNNUC01	Power Plants New 1 - Nuclear	PJ	GW	ANNUAL		

Figure 164. Discrete investment option for nuclear power plants

3.12.3 Scenario files

3.12.3.1 Scen_NUC_DiscInv – lumpy investments

The second step necessary to enable lumpy investments is to specify allowable discrete capacity investments (NCAP_DISC) in specific years for new nuclear power plants (ELCNNUC01). In this example (Figure 165) the capacity installed for this process can be a module of 1 GW in 2015, while in 2033 the model can install 2 GW or 3 or 4 or 5 GW.

~TFM_INS						
TimeSlice	LimType	Attribute	Year	Other_Indexes	AllRegions	Pset_PN
		NCAP_DISC	2015	1	1	ELCNNNUC01
		NCAP_DISC	2033	2	2	ELCNNNUC01
		NCAP_DISC	2033	3	3	ELCNNNUC01
		NCAP_DISC	2033	4	4	ELCNNNUC01
		NCAP_DISC	2033	5	5	ELCNNNUC01
		NCAP_DISC	0	1	5	ELCNNNUC01
		NCAP_DISC	0	2	5	ELCNNNUC01
		NCAP_DISC	0	3	5	ELCNNNUC01
		NCAP_DISC	0	4	5	ELCNNNUC01
		NCAP_DISC	0	5	5	ELCNNNUC01

Figure 165. Discrete capacity at specific years for nuclear power plants

In summary, the TIMES lumpy investment variant can be enabled following four steps:

1. Specify the SET DSCINV for the process for which lumpy investment is to be enabled (here new power plants (ELCNNUC01) in the ELC sheet of the SubRES_NewTechs file).
2. Build a scenario file with the discrete capacity modules to be allowed: capacities for the new power plants (ELCNNUC01) in the NUC_DSCINV sheet of the Scen_NUC_DiscInv scenario.
3. Before solving the model, it is necessary to enable the variant discrete investment in VEDA-FE. From the FE Case Manager, select the **Control Panel** button, check the box for Discrete Investment at the top right in the TIMES Extensions section (Figure 166), and click the **OK** button. Back in the FE Case Manager, the inscription DSC YES in the yellow section at the bottom of the window shows that the option is enabled.
4. In the Control Panel, set OPTCR (optimization criterion, or tolerance) to 0, in order to get a truly optimal solution. For example, if you leave OPTCR at its default value 0.1, in most models this will leave room for very different MIP solutions that would satisfy the optimality tolerance, and thus you could see lots of flip-flopping between model runs (even when using exactly the same scenario data).

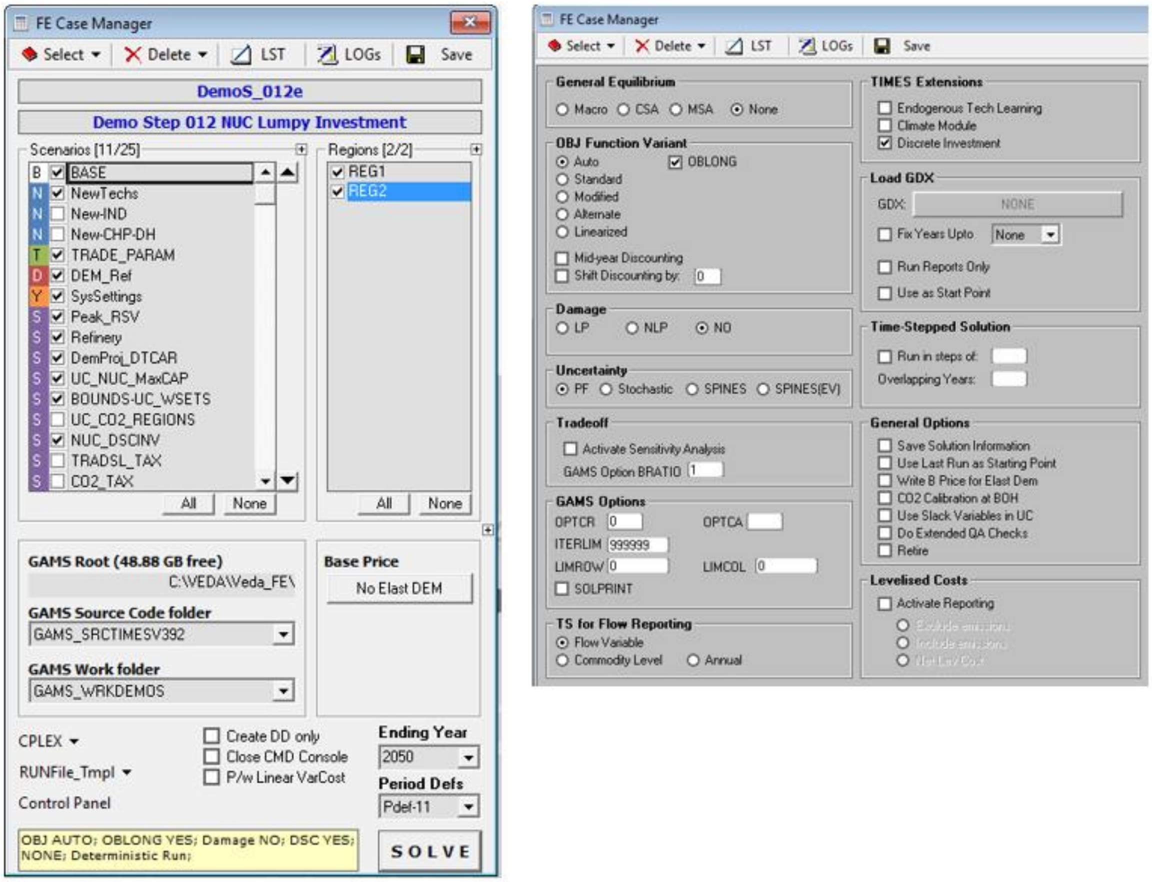


Figure 166. Enable the variant discrete investment in VEDA-FE

3.12.3.2 Scen_TRADSL_Tax

This file is used to introduce a flow tax (FLO_TAX) on processes and commodities (input/output) (Figure 167). This is a new attribute that allows imposing an incremental cost of using/producing a commodity by a process (cost in Currency per unit of commodity produced or consumed). Here it is used to impose a flow tax on all the transportation processes (T*) consuming the diesel commodity (TRADSL) at specific years in each region.

~TFM_INS							
TimeSlice	LimType	Attribute	Year	REG1	REG2	Pset_PN	Cset_CN
		FLO_TAX	2015	10	10	T*	TRADSL
		FLO_TAX	2050	30	20	T*	TRADSL
		FLO_TAX	0	5	5	T*	TRADSL

Figure 167. Flow tax on commodities

3.12.3.3 Scen_CO2_Tax

This file is used to introduce a tax on a net quantity of commodity (COM_TAXNET). Here we impose a tax on the new emission aggregated commodity (TOTCO2) created in B-Y Templates (VT_REG*_PRI_V12) at specific years (Figure 168).

~TFM_INS						
TimeSlice	LimType	Attribute	Year	REG1	REG2	Cset_CN
		COM_TAXNET	2015	20	15	TOTCO2
		COM_TAXNET	2050	50	50	TOTCO2
		COM_TAXNET	0	5	5	TOTCO2

Figure 168. Tax on net quantity of commodities

3.12.3.4 Scen_Solar_Subsidies

This file is used to introduce a flow subsidy (FLO_SUB) on commodities (Figure 169). This is a new attribute that allows creating a credit for using/producing a commodity by a process (cost in Currency per unit of commodity produced or consumed). Here a flow subsidy on the electricity (ELC) commodity produced by all processes consuming the solar energy commodity (ELCSOL) is created with various values at specific years in each region.

~TFM_INS							
TimeSlice	LimType	Attribute	Year	REG1	REG2	Pset_CI	Cset_CN
		FLO_SUB	2010	15	10	ELCSOL	ELC
		FLO_SUB	2050	25	25	ELCSOL	ELC
		FLO_SUB	0	5	5	ELCSOL	ELC

Figure 169. Flow subsidy on commodities

3.12.3.5 Scen_UC_CO2_Regions

This file introduces a new user constraint that imposes limits on all CO2 emissions, summed over all regions and sector emissions. These upper bounds (or limits) are calculated as a percentage reduction target from the total CO2 emissions (TOTCO2 in kt) in a reference scenario for 2020 (10%) and 2050 (15%). It is necessary to run the step model without any limit on emissions first to get the reference emission trajectory (run DemoS_012) and to calculate the bounds as reduction from the reference emissions.

Comparing this scenario with Scen_UC_BND, the differences are the ~UC_Sets (using R_S: AllRegions rather than R_E: AllRegions) and the declaration (UC_RHSTS rather than UC_RHSRTS).

~UC_Sets: R_S: AllRegions							
			~UC_T				
UC_N	Cset_CN	Year	LimType	UC_COMNET	UC_RHSTS	UC_RHSTS~0	UC_Desc
AU_CO2BND_AllRegions	TOTCO2	2020	UP	1	4078637	5	CO2 upper bound on the toal emissions of All Regions
	TOTCO2	2050	UP	1	4632568		

Figure 170. User constraint on the aggregation of emission commodities

3.12.4 Results

This model is mainly run to show the impacts of the different taxes and subsidies, as well as the effects of the lumpy investment feature of TIMES through the discrete capacity requirement for the new nuclear power plants. Regarding fuel consumption in transportation (Figure 171):

- The tax on diesel consumption in the transportation sector (DemoS_012a) leads to a rapid decrease in refined products, reaching zero by 2025, to the benefit of renewable energies, which meet most of the demand by 2050.
- The tax on total CO2 emissions (DemoS_012b) leads to an even more drastic decrease of refined products, reaching zero by 2010, to the benefit of renewable energies.
- The limit on total CO2 emissions (DemoS_012d) does not have an impact on the transportation fuel mix but affects other parts of the whole energy system. The tax puts much higher pressure on the energy system than the limit.

Regarding the electricity generation capacity (Figure 172):

- The tax on total CO2 emissions (DemoS_012b) has important impacts on the electricity sector as well, where most of the thermal generation capacity is replaced with wind power.
- The subsidy on solar power (DemoS_012c) leads to a more diversified mix, as part of the wind power is replaced with solar power.
- The declaration of discrete capacity for nuclear power plants (DemoS_012e) limits the nuclear growth, with only 1 GW of new capacity addition in 2020, 2025, 2030 and 10 GW in 2035 compared with 121 GW in the reference case (Figure 173).

Consumption by Sector and fuel

Original Units: PJ **Active Unit** PJ **Data values filter:**

Attribute | *Vintage* | *TimeSlice* | *Commodity* | *Region* | *Process*

~-Scenario-~	-ProcessSet-	CommoditySet	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_012	Fuel Consumption Transport Sector	Electricity	269	260	590	638	689	745	765	785	805	825	846
		Natural Gas	21	19	11								
		Petroleum & Products	12,816	13,190	13,329	14,767	16,731	25,529	26,174	26,835	27,512	28,207	28,919
		Renewables	163	150	98	34	25	17	8				
		Total	13,269	13,619	14,029	15,439	17,446	26,291	26,947	27,620	28,317	29,032	29,765
DemoS_012a	Fuel Consumption Transport Sector	Electricity	269	260	590	638	689	745	765	785	805	825	846
		Natural Gas	21	19	11								
		Petroleum & Products	12,816	13,190	13,329	8,331	1,575						
		Renewables	163	150	98	6,749	15,679	26,184	26,837	27,506	28,200	28,912	29,642
		Total	13,269	13,619	14,029	15,718	17,943	26,929	27,601	28,290	29,005	29,737	30,488
DemoS_012b	Fuel Consumption Transport Sector	Electricity	269	260	590	638	689	745	765	785	805	825	846
		Natural Gas	21	19									
		Petroleum & Products	12,816	13,190									
		Renewables	163	150	13,412	15,170	17,175	26,184	26,837	27,506	28,200	28,912	29,642
		Total	13,269	13,619	14,003	15,808	17,864	26,929	27,601	28,290	29,005	29,737	30,488
DemoS_012d	Fuel Consumption Transport Sector	Electricity	269	260	590	638	689	745	765	785	805	825	846
		Natural Gas	21	19	11								
		Petroleum & Products	12,816	13,190	13,329	14,767	16,731	25,529	26,174	26,835	27,512	28,207	28,919
		Renewables	163	150	98	34	25	17	8				
		Total	13,269	13,619	14,029	15,439	17,446	26,291	26,947	27,620	28,317	29,032	29,765

Figure 171. Results – Fuel consumption for transportation in DemoS_012

ELC plants capacity and new capacity

Original Units: GW   Active Unit: GW   Data values filter:

Attribute | *Process* | *Vintage* | *Region*

~-Scenario-~	ProcessSet	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_012	Coal Power Plants	137	133	127	146	123	103	91	71	71	71	66
	Gas Power Plants	104	98	86	115	160	215	219	219	211	196	188
	Hydro power plants	62	62	45	32	32	32	14				
	Nuclear Power Plants	115	115	115	115	117	119	121	128	131	140	147
	Oil Power Plants	11	11	10	8	16	19	21	25	26	26	28
	Solar/PV power plants	14	13	10	5							
	Wind Power Plants	68	65	51	34	75	103	147	184	200	218	234
	Total	512	498	443	454	522	590	612	628	639	652	663
DemoS_012b	Coal Power Plants	137	133	115	92	69	46	23				
	Gas Power Plants	104	98	78	52	26						
	Hydro power plants	62	62	45	32	32	32	14				
	Nuclear Power Plants	115	115	115	115	117	119	121	128	134	140	147
	Oil Power Plants	11	11	10	8	6	36	61	84	87	90	93
	Solar/PV power plants	14	13	10	5							
	Wind Power Plants	68	65	599	732	861	1,012	1,072	1,099	1,123	1,148	1,173
	Total	512	498	971	1,035	1,111	1,245	1,291	1,311	1,344	1,378	1,413
DemoS_012c	Coal Power Plants	137	133	127	146	123	101	89	70	70	70	66
	Gas Power Plants	104	98	86	115	160	216	220	220	212	199	188
	Hydro power plants	62	62	45	32	32	32	14				
	Nuclear Power Plants	115	115	115	115	117	119	121	128	131	140	147
	Oil Power Plants	11	11	10	8	16	19	21	25	28	28	34
	Solar/PV power plants	14	13	10	5					48	85	125
	Wind Power Plants	68	65	51	34	75	103	147	184	161	148	131
	Total	512	498	443	454	522	590	612	628	649	670	691
DemoS_012e	Coal Power Plants	137	133	127	146	123	102	90	124	124	124	113
	Gas Power Plants	104	98	86	115	161	218	224	295	287	275	268
	Hydro power plants	62	62	45	32	32	32	14				
	Nuclear Power Plants	115	115	115	115	116	117	118	13	18	23	33
	Oil Power Plants	11	11	10	8	16	19	21	20	20	20	23
	Solar/PV power plants	14	13	10	5							
	Wind Power Plants	68	65	51	34	75	103	147	184	200	218	234
	Total	512	498	443	454	523	590	613	635	648	660	671

Figure 172. Results – Electricity generation capacity in DemoS_012

ELC plants new installed capacity in each period

Original Units: GW   Active Unit: GW   Data values filter:

Attribute | *Process* | *Region*

~Scenario~	ProcessSet	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_012	Coal Power Plants	12	42		3	11	4			7
	Gas Power Plants	8	55	71	81	4			40	63
	Nuclear Power Plants			2	2	2	121	3	10	6
	Oil Power Plants			10	4	4	6	1		2
	Wind Power Plants			57	45	44	38	73	64	59
DemoS_012e	Coal Power Plants	12	42		2	11	57			1
	Gas Power Plants	8	55	72	83	6	71		44	65
	Nuclear Power Plants			1	1	1	10	5	5	10
	Oil Power Plants			10	4	4	1			4
	Wind Power Plants			57	45	44	37	73	64	59

Figure 173. Results – New capacity investments for electricity generation in DemoS_012

Energy Technology Systems Analysis Programme

http://www.iea-etsap.org/web/Documentation.asp

Documentation for the TIMES Model

PART V

October 2016

Authors: Evelyn Wright
Amit Kanudia
Richard Loulou
Gary Goldstein

General Introduction

This documentation is composed of five Parts.

<u>Part I</u> provides a general description of the TIMES paradigm, with emphasis on the model's general structure and its economic significance. Part I also includes a simplified mathematical formulation of TIMES, a chapter comparing it to the MARKAL model, pointing to similarities and differences, and chapters describing new model options.

<u>Part II</u> constitutes a comprehensive reference manual intended for the technically minded modeler or programmer looking for an in-depth understanding of the complete model details, in particular the relationship between the input data and the model mathematics, or contemplating making changes to the model's equations. Part II includes a full description of the sets, attributes, variables, and equations of the TIMES model.

<u>Part III</u> describes the organization of the TIMES modeling environment and the GAMS control statements required to run the TIMES model. GAMS is a modeling language that translates a TIMES database into the Linear Programming matrix, and then submits this LP to an optimizer and generates the result files. Part III describes how the routines comprising the TIMES source code guide the model through compilation, execution, solve, and reporting; the files produced by the run process and their use; and the various switches that control the execution of the TIMES code according to the model instance, formulation options, and run options selected by the user. It also includes a section on identifying and resolving errors that may occur during the run process.

<u>Part IV</u> provides a step-by-step introduction to building a TIMES model in the VEDA-Front End (VEDA-FE) model management software. It first offers an orientation to the basic features of VEDA-FE, including software layout, data files and tables, and model management features. It then describes in detail twelve Demo models (available for download from the ETSAP website) that progressively introduce VEDA-TIMES principles and modeling techniques.

<u>Part V</u> describes the VEDA Back-End (VEDA-BE) software, which is widely used for analyzing results from TIMES models. It provides a complete guide to using VEDA-BE, including how to get started, import model results, create and view tables, and create and modify user sets, and step through results in the model Reference Energy System. It also describes advanced features and provides suggestions for best practices.

PART V: VEDA Back-End

TABLE OF CONTENTS FOR PART V

1 Introduction

Part V of the TIMES documentation provides a user manual for the **VE**rsatile **D**ata **A**nalyst **B**ack-**E**nd (VEDA[1]-BE) software, which is widely used for analyzing results from TIMES models. VEDA-BE is a flexible, user-friendly tool for the construction of analyst tables and graphs from structured datasets of any kind.

VEDA-BE is particularly suitable for analyzing results from GAMS-based models built on the concepts of *sets* for identifying and grouping model components and *parameters* for the management of numeric values, including those obtained from the model solution (e.g. the primal and dual solution values obtained from optimization). Any GAMS-based model can be linked to VEDA-BE by means of the GDX2VEDA utility and the associated <application>.VDD VEDA data definition file (see Appendix B for more).

VEDA-BE has additional features to work with the TIMES family of models that rely on a Reference Energy System (RES) network structure, where nodes in the network represent processes, and the arcs linking these nodes represent commodities produced and/or consumed by these processes. By means of the GDX2VEDA utility, VEDA-BE is made aware of a TIMES model's network topology, allowing the user to trace flows of commodities through the network by means of process inputs and outputs.

VEDA-BE may be used in conjunction with TIMES models built and managed in either the VEDA Front-End or ANSWER-TIMES model management systems. This document describes how to use VEDA-BE with both systems. It uses the VEDA-TIMES DemoS model described in Part IV as an example throughout; however, the instructions are general, and use of this demo model is not required in order to follow along.

Part V is organized as follows:

Chapter 1	Introduction and installation: introduces VEDA-BE and provides instructions to download and install the software.
Chapter 2	Getting started: provides an overview of the user environment, describes how to create a new database and how to import TIMES results, and introduces how to view results in ExRES and table views.
Chapter 3	Sets: describes the role of sets in viewing results in VEDA-BE. Explains how to construct and modify sets, how to use them in tables, and how to move them between VEDA-BE databases.
Chapter 4	Tables: describes how to build, customize, and manage tables; how to move tables between VEDA-BE databases; and some advanced features for working with tables.
Chapter 5	Units handling: describes the VEDA-BE units handling facility.
Chapter 6	Working with Excel and producing reports: describes ways to use VEDA-BE to produce analysis reports, including using VEDA-BE in conjunction with Excel and using the VEDA-BE batch mode.
Chapter 7	Best practices: provides some best practice suggestions.
Appendix A	VEDA-BE Attribute Reference Guide.
Appendix B	The GDX2VEDA utility and the VEDA data definition file.

[1] Veda [Sanskrit,=knowledge, cognate with English wit, from a root meaning know], oldest scriptures of Hinduism and the most ancient religious texts in an Indo-European language. [The Columbia Encyclopedia, 6th ed. New York: Columbia University Press, 2000.]

1.1 Software installation

Instructions for installing VEDA-BE are available at http://support.kanors-emr.org/.

2 Getting started with VEDA-BE for TIMES modeling

This section provides a guide to getting started with VEDA-BE for TIMES modeling, including creating a new database, importing results into an existing database, and viewing simple tables and graphs.

2.1 Bringing TIMES results into VEDA-BE

VEDA-BE requires as input four text files containing full information on each model run that the user wishes to process and examine. These files are produced from model run GAMS Data eXchange (GDX) files by the GAMS GDX2VEDA utility (see Appendix B). The generic names and role of these files are as follows:

- <scenarioname>.VD, containing the application VEDA-BE header directives (controlling the appearance of the main VEDA-BE table specification form) and actual model data;
- <scenarioname>.VDE, containing the list of individual set member elements for each index managed by VEDA-BE with their descriptions;
- <scenarioname>.VDS, where the set grouping information is provided for the various sub-sets associated with each index managed by VEDA-BE, and
- <scenarioname.VDT>, containing the topology (RES) information, for MARKAL/TIMES models only.

These files are deposited by the GDX2VEDA utility in the user's GAMS Work folder. (The default path is \VEDA\VEDA-FE\GAMS_WrkTIMES for VEDA-FE users, and \AnswerTIMESv6\GAMS_WrkTI for ANSWER users, but the user may choose to set up different Work folders, for example, for different projects.)

Following a model run, in order to analyze results in VEDA-BE, these files must first be brought into VEDA-BE. This may be done (occasionally, for example, when starting a new project) by creating a new VEDA-BE database from the run results, or (far more often) by importing run results into an existing database. Both operations are described in this section, along with other results management facilities, after quick instructions on how to follow along with the DemoS example database used in this document.

2.1.1 The DemoS working example

Throughout Part V, we will work with the TIMES DemoS_012 model as an example. The user may choose to follow along by downloading the Demo model package.[2] The ETSAP_DemoS_VFE folder should be copied into the user's /VEDA/Veda_Models directory, and the DemoS_VBE folder should be copied into the user's /VEDA/VEDA-BE/Databases folder.

To create an example run, we have run the DemoS_012 case, changing the run name so that we can compare the results to the DemoS_012 run already in the DemoS_VBE database, as shown in Figure 1. We have also created the Gams_WrkDEMO folder to receive our results files. (See Part IV for more on running the Demo models in VEDA-FE.)

---

[2] Available on the ETSAP website along with the TIMES documentation at http://iea-etsap.org/index.php/documentation.

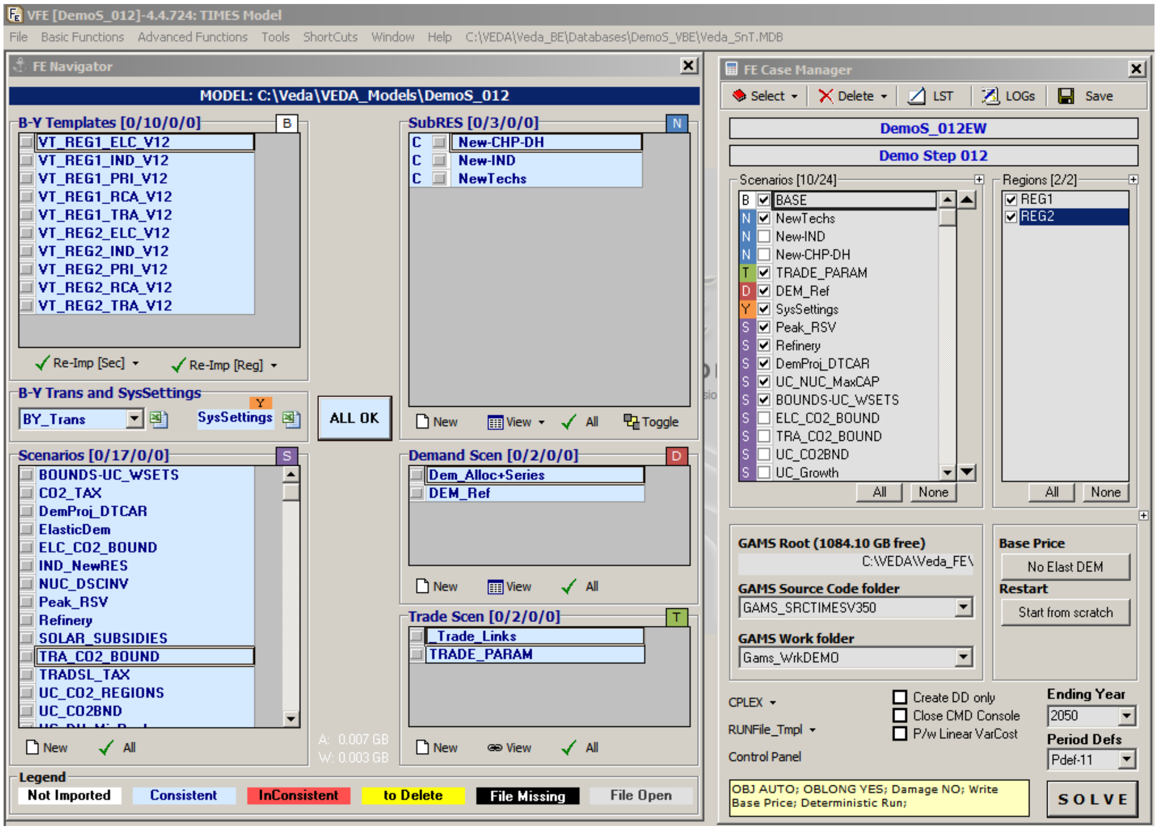


Figure 1: Creating an example DemoS_012 run

2.1.2 Creating a new VEDA-BE database

To create a new VEDA-BE database, from the **File** menu choose **New Database** and then select **Local**. Depending on your set-up, you may now see a window titled "**Flavors**". If so, see instructions starting at Figure 5 below. You may also see the following message:

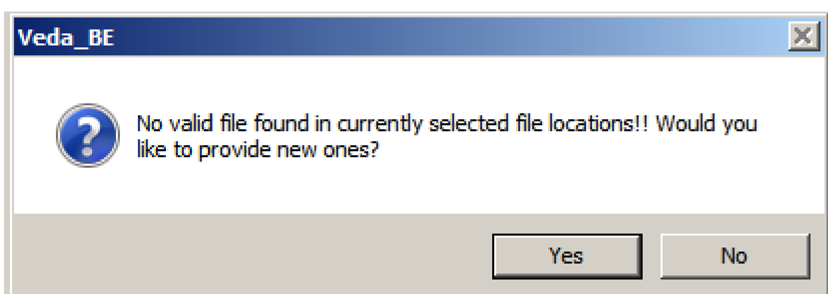


Figure 2: Prompt to set file locations

Click "**Yes**", and you will be delivered to the **Input Files Locations** window:

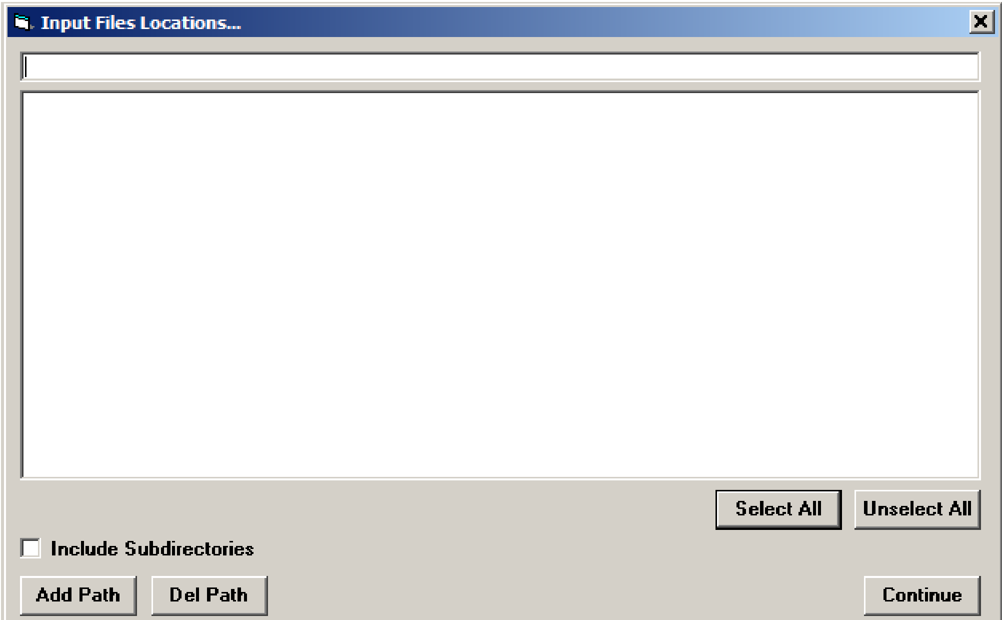


Figure 3: The Input Files Locations window

Click **Add Path**, which will bring up a folder **Browse** window. Scroll to the Veda_FE folder and click the "+" next to it. Then select the GAMSWrk folder you have used in your model run, as shown in Figure 4, and click **OK**[3].

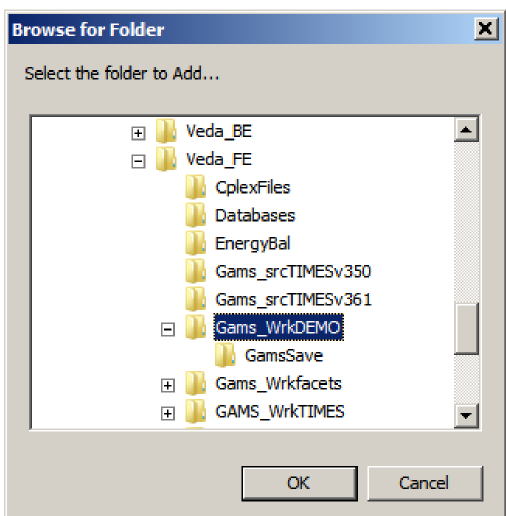


Figure 4: Selecting your GAMSWrk folder in the folder Browse

[3] If you use ANSWER to produce TIMES model runs, the Input Files Location path is set by navigating in this browse to your GAMS Work folder (default GAMS_WrkTI) within your AnswerTIMESv6 folder. Although the DemoS_012 model is a VEDA-TIMES model, you may follow along with the examples in this guide using the pre-existing DemoS_VBE database.

This should return you to the **Input Files Locations** window, where clicking **Continue** will bring you to the **Flavors** window:

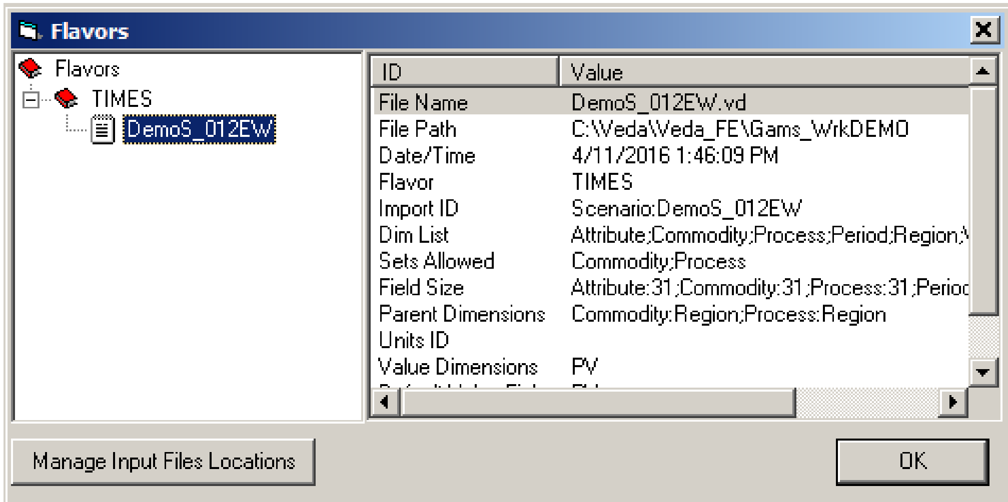


Figure 5: Selecting the run to create a new database in the Flavors window

In the **Flavors** window, click the "+" next to TIMES (the "flavor" of VEDA model that we will be working with), and your model run should appear. Select your run and Click **OK**. You will be prompted for a database name (e.g., *DemoS_VBE* here), whereupon VEDA-BE will create the new database and import your run. When the import is complete, you should see your scenario listed on the **Scenarios** tab.

A note on terminology: In VEDA-FE and ANSWER, model runs are called *cases*, and are assembled from one or more *scenarios* consisting of input data that are layered to form the input assumptions for a particular case. However, in VEDA-BE, the model runs upon import are referred to as *scenarios*. Because VEDA-BE is generally used only to review TIMES model output data, this change in terminology should not cause any (lasting) confusion.

2.1.3 Importing results into an existing database

It is generally more convenient to work from an existing database, if one has been previously created for your model/project, because it will likely be already populated with Sets and Tables that are useful for analyzing your model. (Throughout this document, we will work with the DemoS_VBE database that is distributed with the DemoS model.)

Upon starting, VEDA-BE will open the last used database, if any. To open a desired database, choose **Open Database** from the **File** menu. You will see a folder Browse of all the databases (folders) in the (default) /VEDA/VEDA-BE/Databases folder, from which you can select the desired database[4].

[4] You may store your databases in any location. Each "database" will always be a folder consisting of several Access databases, as described in Section 2.1.5.

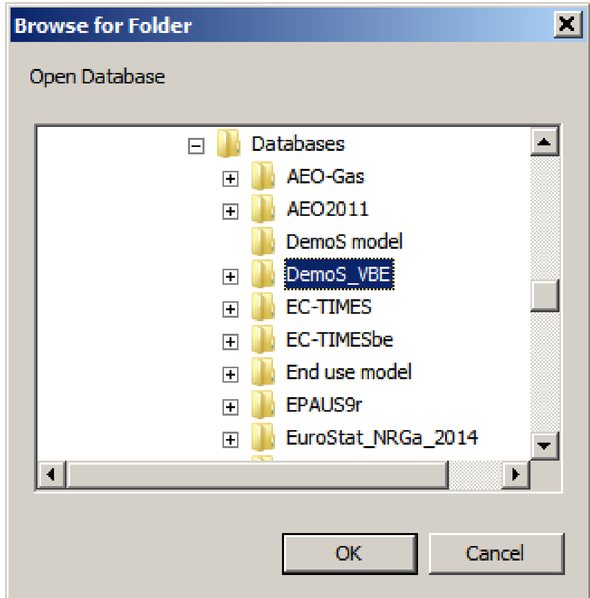


Figure 6: Opening a database

To import a run or runs into an existing database, from the **Results** menu choose **Import/Archive**. This brings up the following window:

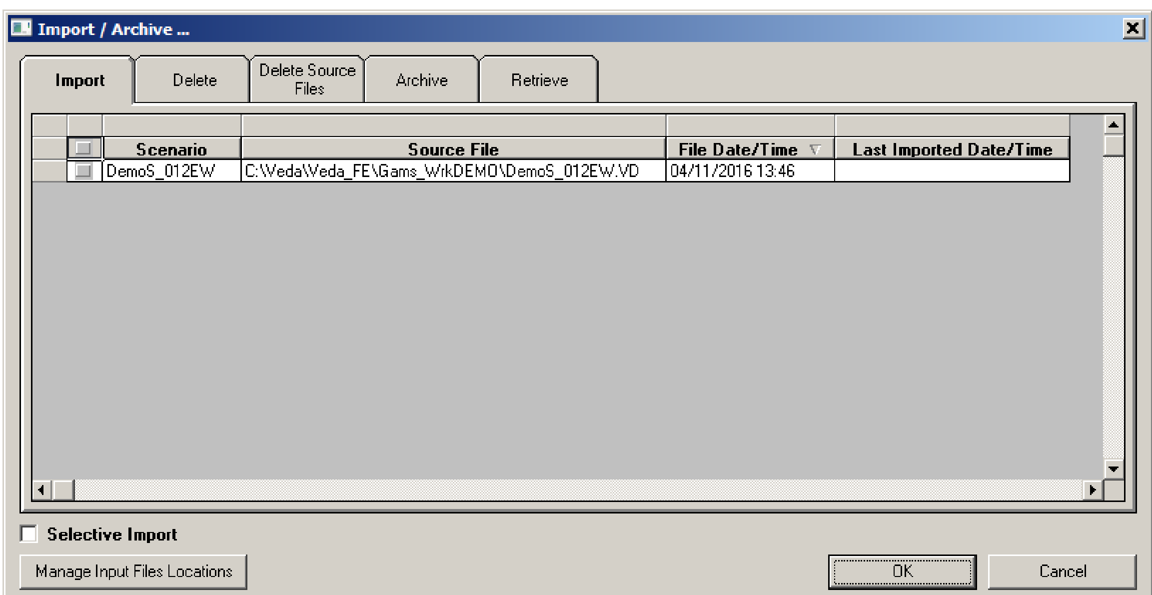


Figure 7: The Import/Archive window

You may see your model run listed here. If so, click on the checkbox next to its name to select it, and click **OK**.

Alternatively, the Import file list may be blank, because the path to the GAMSWrk folder has not been set yet. In this case, click **Manage Input Files Locations**. This will take you to the **Input Files Locations** window shown in Figure 3, where you can click **Add Path** and navigate to and select your GAMSWrk folder as described in Section 2.1.2. This should be a one-time operation, unless you change GAMSWrk folders, and the Import file list should be populated with new results as you conduct runs from now on.

Note that the **Input Files Locations** box can also be used to remove a path from a VBE database, if desired (for example, if you decide to create separate GAMSWrk folders for different projects later on.)

Once you have selected your run(s) to import and clicked **OK**, the **Tasks** window should appear, showing you the progress of the import process. Depending on the size of your model, this process may take a few seconds (for the DemoS model) up to several minutes for very large models.

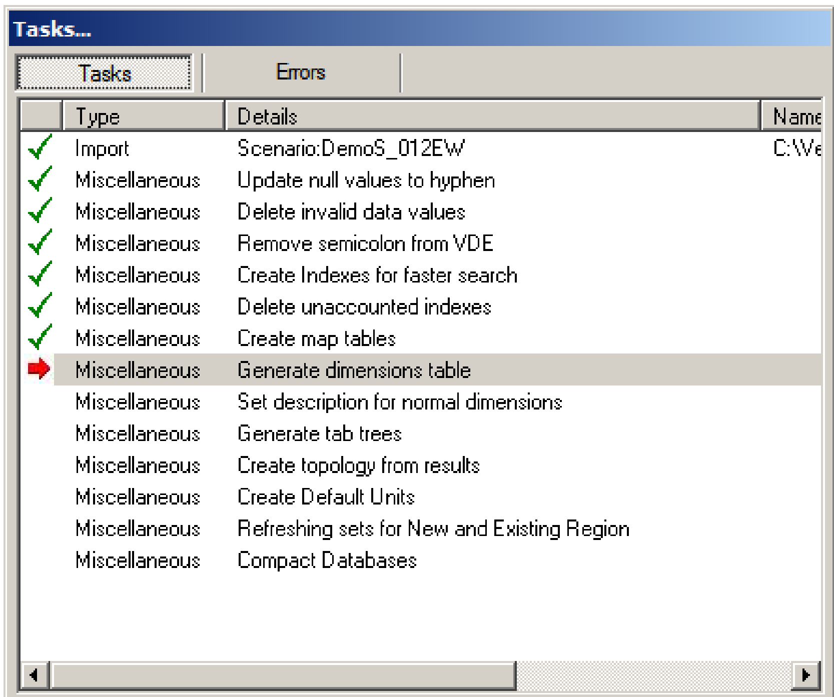


Figure 8: The Tasks window for importing and deleting scenarios

2.1.4 Managing scenarios in a VEDA-BE database

From the **Import/Archive** window, one may perform additional tasks to manage the scenarios stored in your VEDA-BE database. Clicking on the **Delete** tab will bring up a list of all the scenarios that are currently in the database. Select the scenarios you want to delete by checking the checkboxes next to their names and click **CONTINUE**. This will bring you back to the **Import** tab, where you can select scenarios for Import, if desired, and then click **OK** to start the Delete (and Import, if any) process.

To streamline the DemoS_VBE database for our use, we'll delete all the scenarios from DemoS 1-11. With the **Import/Archive** window open, click on the **Delete** tab to bring up the list of all scenarios in the database. Clicking on the checkbox next to "Scenario" (see the circled area in Figure 9) will select *all* scenarios in the list. You may sort the list by scenario name by clicking on **Scenario**, then scroll to the bottom and unselect the DemoS_012 scenarios by clicking on each of their checkboxes to clear the checks. Click **CONTINUE** and then **OK** to start the delete.

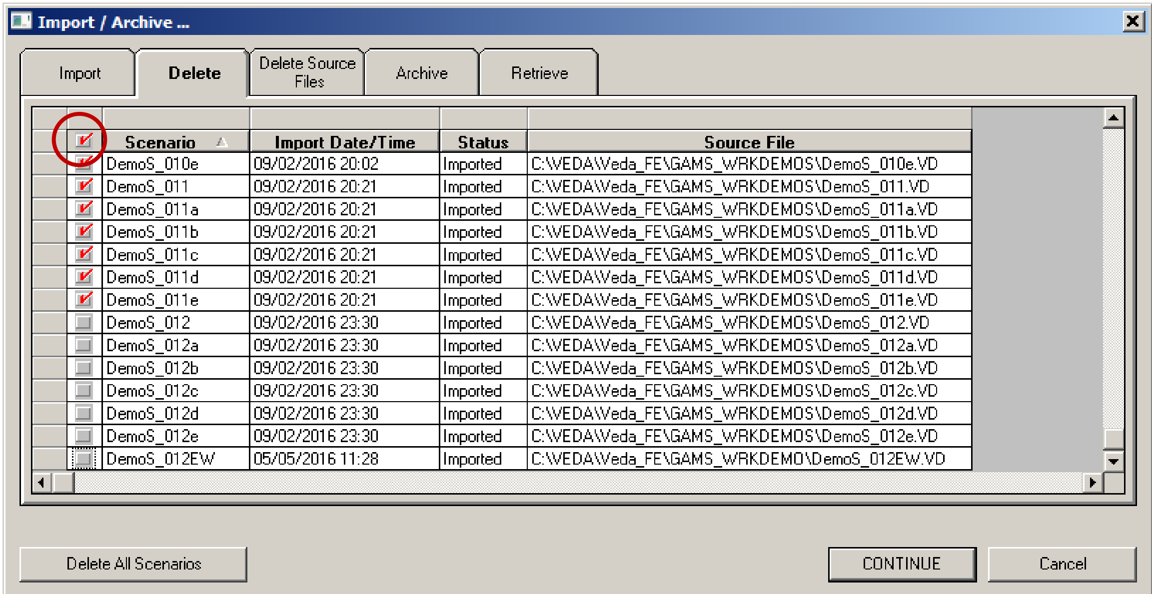


Figure 9: Deleting scenarios from the database

From the **Import/Archive** window, you may also delete old VD* files from your GAMSWrk folder, by clicking on the **Delete Source Files** tab and selecting the scenarios for deletion. Once you have imported scenarios into your VEDA-BE database, the VD* files are no longer needed and should be periodically deleted to save disk space.

Note that if, upon opening the **Import/Archive** window, VEDA-BE finds results files with the same name but a later time stamp than a scenario already in the database, indicating that you have run a new version of a previously imported scenario, it will automatically select that scenario for import. You may deselect it, if desired, by clearing the checkbox next to its name. If you request the import to proceed, it will first delete the previous version of the scenario.

2.1.5 The structure of a VEDA-BE database

A VEDA-BE database is a folder residing by default in the /VEDA/VEDA-BE/Databases folder, containing several Microsoft Access databases (.mdb files). These include:

- One Access database named VEDA_R_<scenarioname> corresponding to each scenario that is currently in the VEDA-BE database;
- Access databases named VEDA-R and VEDA-Y. These hold the structure of the scenario-specific files, and the scenario and table dimensions, and are meant for the internal use of the application. Users should never modify these files; and
- An access database named VEDA_SnT.

The SnT file is of particular importance, as it can be used to transfer Sets and Tables from one VEDA-BE database to another, as described in Sections 3.4 and 4.4. **The other Access files should not be moved or copied by the user from one location to another.**

2.1.6 Moving/sharing a VEDA-BE database

The recommended procedure to move an existing VEDA-BE database from one computer to another and/or share it with a colleague is to create a new database from one model run, and then import any needed scenarios, sets, and tables as follows:

Select a model run on the old machine, locate the four VD* files from that run in your GAMSWrk folder, and compress and move them to the GAMSWrk folder on the other machine. Also locate the VEDA_SnT file in the VEDA-BE database folder and copy it to a known location on the other machine, such as the Desktop. Next create a new database on this machine from the VD* files, following the procedure described in Section 2.1.2. Finally, import desired sets and tables from the SnT file, following the procedures described in Sections 3.4 and 4.4. Additional scenarios may be imported into the new database by copying their VD* files to the new machine's GAMSWrk folder and using the **Import/Archive** window.

Merging scenarios, sets, and/or tables from existing one database into another may be accomplished by following the same procedure, minus the creation of a new database.

Note that it is not recommended to short-cut this procedure by moving the entire database folder and/or the SnT file to the new machine. Differences in VEDA-FE settings between the two machines may cause the required SnT file format to be slightly different, resulting in incompatibilities.

2.2 Orientation to the main VEDA-BE window

If you have imported and deleted scenarios from the DemoS_VBE database as described in Section 2.1, your screen should now look something like that shown in Figure 10.

The arrows in Figure 10 call out the primary features of the main VEDA-BE window. Below the **Main menu** bar (described in Section 2.2.2), the window is divided in two. The left panel is the **Table definition** panel, which contains at its top a dropdown list of existing tables (if any have been previously defined). The drop-down menu also contains the option to define a new table, as shown in Figure 10. Underneath the menu is the list of dimensions and elements that describe the selected table (if any). The user can choose (and then optionally modify) a pre-defined table from the list, or create a new one altogether. Note that the list of tables shown is the one in the folder that has been selected under *\View\Open folder(s)* in the main menu (see Section 4.4). If no folder has been selected, all tables are shown. The table definition process is described in Section 4.1.

The right panel holds the **Dimension Tabs**, which display the detailed lists of all elements in the database. The available dimensions for TIMES models are described in Section 2.2.1.

Additional features of note in the main window include:

- The **Units** drop-down menu allows the user to specify the native model units that are associated with the table. If the user has defined custom units and conversion factors (discussed in Section 5), the display units of the table may be modified inside the table view.
- The **View table(s)** button processes the table definition, and produces the tabular cube view (see Section 2.3) according to the specified criteria.
- The **Global Filter** status bar at the bottom indicates whether global filters have been set, and if so, for which dimensions and elements. This feature allows the user to set filters that will apply to *all* tables viewed and then later to easily remove them for all tables,

without having to redefine the tables. See Section 2.3.5 for instructions on this extremely useful feature.

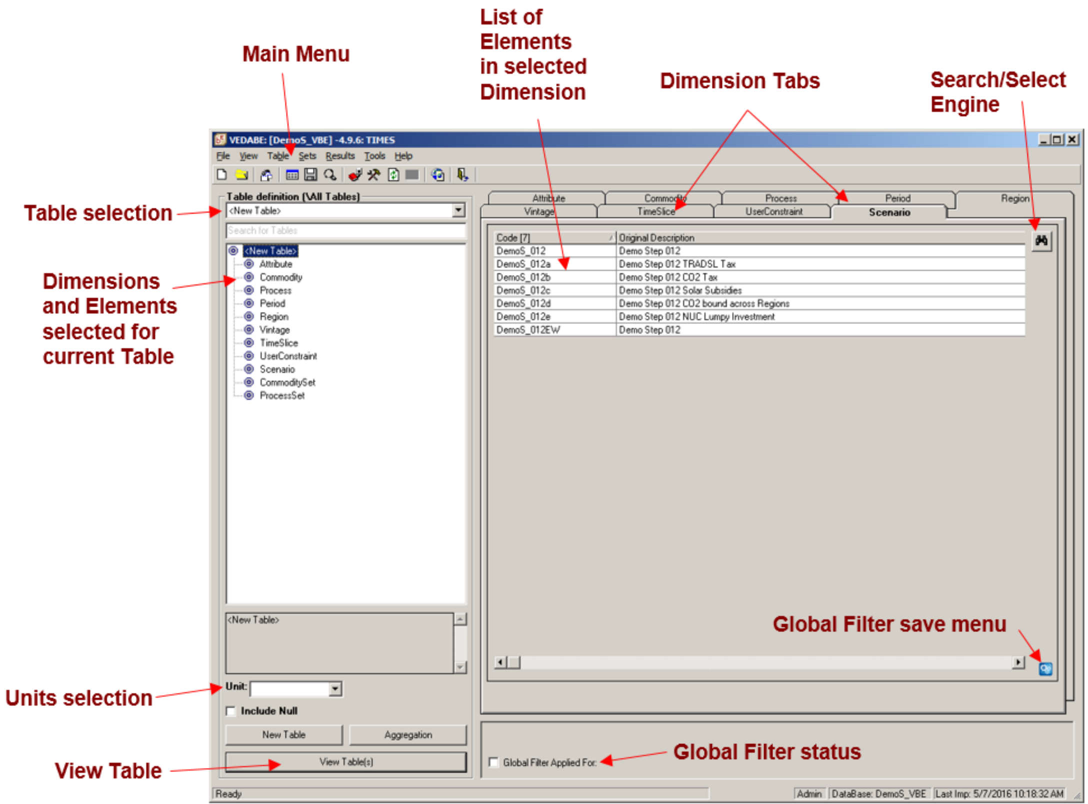


Figure 10: The main VEDA-BE window

2.2.1 The dimension tabs and items lists

The **Dimension tabs** organize all the items that are available to view in the VEDA-BE database. Each tab lists the available elements corresponding to that dimension. Note that for the **Process** and the **Commodity** dimensions, the tab is divided into lists of *sets* of items that TIMES and/or the user have defined (above) and the items themselves (below), as shown in Figure 11.

Each element in a dimension has a *code* and a *description*. The description comes in two flavors: the original description provided from the model, and the Veda description. You may toggle between the Veda and Original descriptions by right-clicking with the cursor in the description column and selecting **Display Type** from the pop-up menu. Initially, these two descriptions are identical, but the user may change the Veda description at will by selecting **Edit description** from this pop-up menu. This feature is particularly useful for scenario descriptions, and may also by useful for set descriptions or region names, in order to adapt the heading of table columns or rows to the terminology of a particular report.

For the TIMES models, the 9 dimensions are:

- **Attribute**: describing the kind of results data to be displayed. There are many VEDA attributes handled for the TIMES models, including process capacities and investments, commodity flows, costs, and marginals. See Appendix A for a full listing.

- **Commodity and Commodity Set**: a *commodity* is an energy form, a material, an emission, or a demand category. In short, a commodity represents a link in the RES network (whereas a process, as described below, represents a node of the RES). A *commodity set* is a group of commodities defined by the model or by the user. Commodity sets are convenient ways to avoid having to list many commodities when defining a table. They will be discussed in detail in Section 3.
- **Process and Process Set (**Figure 11): a *process* is the generic name for a technology or an energy source or energy/emission sink. A *process set* is a group of processes defined by the model itself or by the user. Process sets are a convenient way to avoid having to list many processes when defining a table. They will be discussed in detail in Section 3.

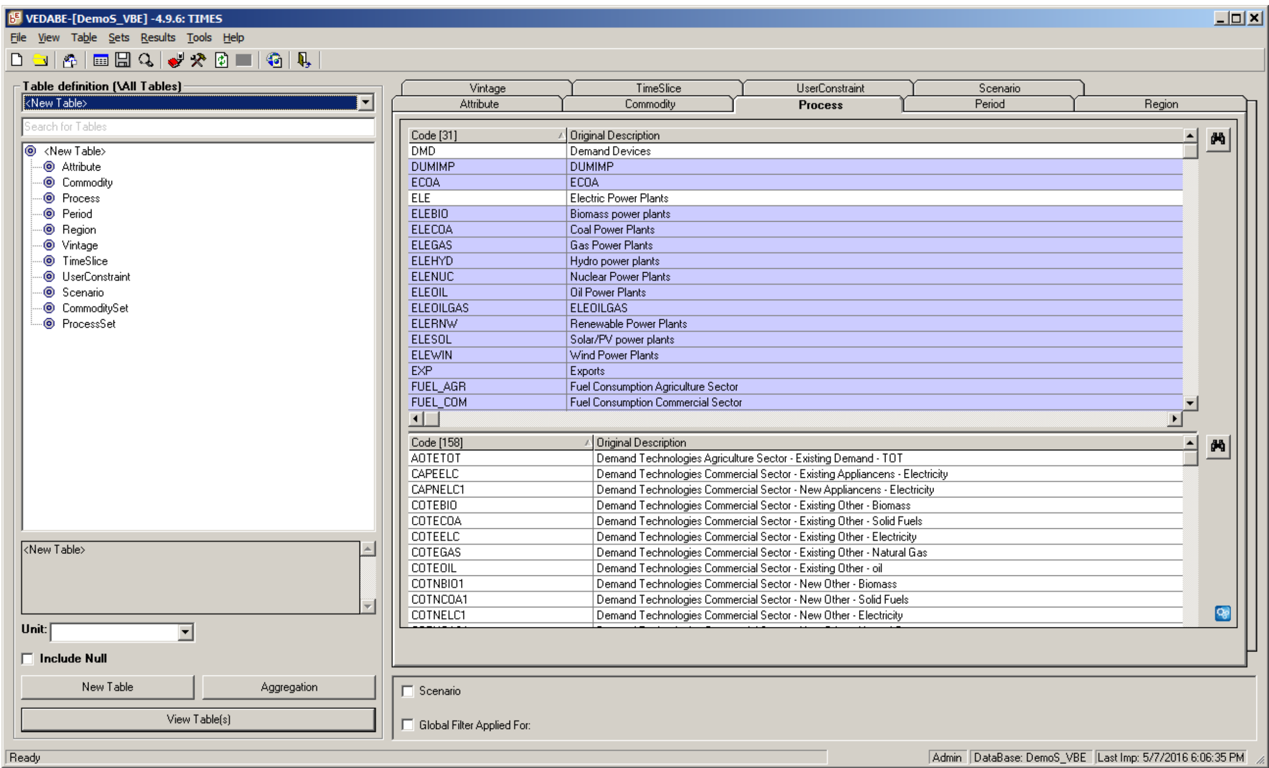


Figure 11: The *Process* tab, showing *process sets* above and *processes* below

- **Period**: lists all the periods in the model, by milestone year.
- **Region**: the various regions represented in the model.
- **Vintage**: the vintage year corresponds to the period in which a process was initially installed, as opposed to the period in which activity occurs. This view may be useful for technologies whose characteristics change with vintage. This dimension also contains the reserved tags 0, -, and ¤ for use in characterizing capacity additions and retirements (see Appendix A).
- **Time Slice**: sub-divisions of model periods according to seasons and time-of-day.
- **User Constraint**: lists all user constraints in the model, along with reserved TIMES tags that distinguish objective function components (see Appendix A).
- **Scenario**: all model runs currently imported in the database, as described in Section 2.1.

2.2.2 VEDA-BE main menus and toolbar

This section summarizes the features available under each of the main menus, as follows.

- **File**: contains commands to create a ***New*** database and ***Open*** an existing one; to ***Backup*** all files in the current database as a single Zip archive and to ***Restore*** a database from such an archive; and to ***Quit*** VEDA-BE.
- **View**: besides the usual view choices such as showing or hiding parts of the screen, this menu has two important functions: a) to allow the user to select subsets of tables in specific ***folders***, and b) to choose the ***execution mode*** (interactive or batch). See Sections 4.4 and 6.2 for these advanced features.
- **Tables**: this menu allows the user to define a ***new*** table, to ***save*** or ***delete*** a table, to ***import*** tables defined in another database and ***export*** tables to share with another database, to erase ***Cube files*** (the final form in which a table is organized), and to open the ***Table Master***. See Section 4 for defining and managing tables.
- **Sets**: this menu allows the user to create ***new*** sets, to ***edit, delete***, or ***refresh*** existing sets, to ***import*** sets defined by another user, and to ***export*** sets to another location. See Section 3 for all operations on sets.
- **Results**: this menu allows ***importing*** and ***deleting*** results, as well as defining the directory where result files are stored, as described in Section 2.1.
- **Tools**: this menu offers a diverse array of tools, including user ***Options*** for setting default table layouts and formats (discussed in Section 4.2) and settings for working with ***Exports*** and ***Batch Mode*** (Section 6); operations on ***Cube Files*** (Section 2.3.7); the ***Units*** facility (Section 5); and the ability to ***Update Excel files*** containing calculations and graphs based on VEDA-BE tables (Section 6.3).
- **Help**: this menu provides links to the KanORS Veda Support website and Veda Support Forum. The Forum has an active user community and is a very good source of information for beginners as well as advanced users.

2.3 Viewing tables

We are now ready to work with the tables that have been defined in the DemoS_VBE database. We'll begin by verifying that our test run of the DemoS_012 scenario has succeeded in duplicating the same solution, by selecting the ***_SysCost*** table from the drop-down menu in the **Table definition** panel, as shown in Figure 12. Note that this table is so useful in initial assessment of model runs that its creator has chosen to begin its name with an underscore (**_**) character, so that it will always sort at the beginning of the table list.

The **Table definition** panel now shows us how this table is constructed. Its specifications consist of only a single entry: the attribute ***Reg_obj***. Clicking on the **Attributes** tab shows us the description of this item: *Regional total discounted system cost*, as shown in Figure 13. Note that this item appears in bold, because it is part of the currently selected table's definition.

Below the **Table Definition** form, we see the table's description, assigned by its creator: *Objective Function Value*. We also see that the table's creator has set the Units for this table to M Euro. These are the *native* units in the model for the costs shown in this table. Having set the native units in the table definition process allows the user to make use of the Units function to later change the *display* units of the table to any units of their choosing.

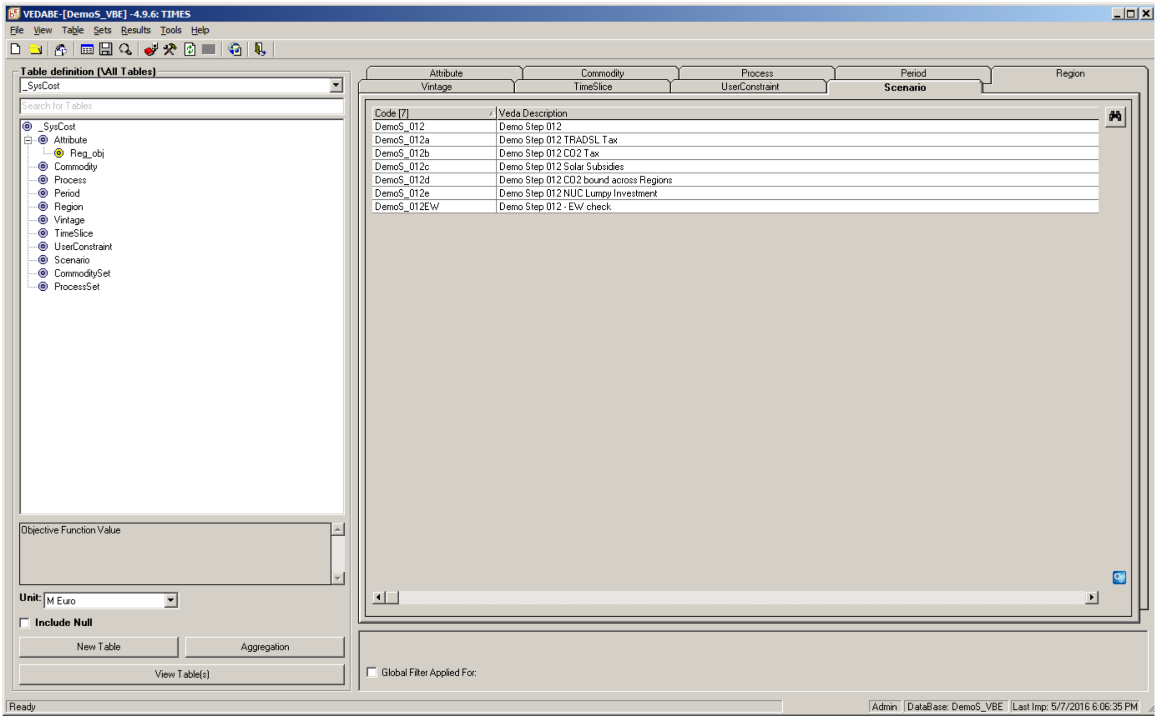


Figure 12: The *_SysCost* table definition

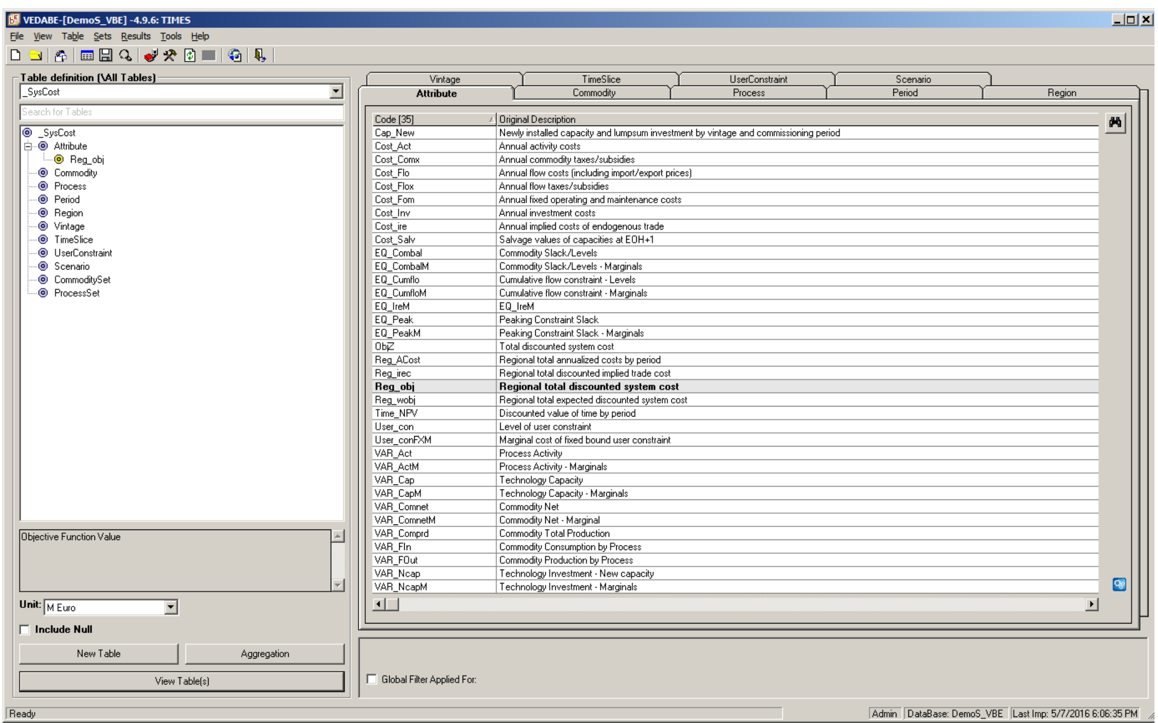


Figure 13: The Attributes tab

Clicking on **View Table(s)** brings us into the table **View window**, as shown in Figure 14. VEDA-BE tables are highly dynamic data *cubes* that may be readily rearranged. Table dimensions are shown in either rows (as for *Scenario* in this example), columns (*Region*), or "on the page", meaning sitting at the top, as *Attribute* is here. Dimensions in rows or columns are enumerated there, while dimensions on the page are summed over. In this case, there is only one *Attribute* in the table (***Reg_obj***), so having it on the page merely gets it out of the way without changing the results shown.

Hovering the cursor over any (black) dimension header will cause a cross-hair to appear, allowing you to hold down the cursor (left-mouse) and drag this dimension to any other position in the table (where a green line will appear). Dragging *Region* to the top will remove the enumerated *Region* data from the table and show only the sum across regions, as shown in Figure 15. Note that the *Region* header is now enclosed in asterisks (*), to alert us that multiple values are being summed over.

The user must take care that dimensions on the page are only those that it is meaningful to sum over. For example, it rarely makes sense to sum over *Attribute* (with the exception of the ***Cost_*** components), so when there is more than one *Attribute* in a table, it should be shown in a row or column. Similarly, summing over *Scenario* is never meaningful, and so *Scenario* should always be in a row or column (with the exception of tables used in single scenario Excel workbooks for update using the features described in Section 6.3, to be used with the Global Filter selecting a single scenario.)

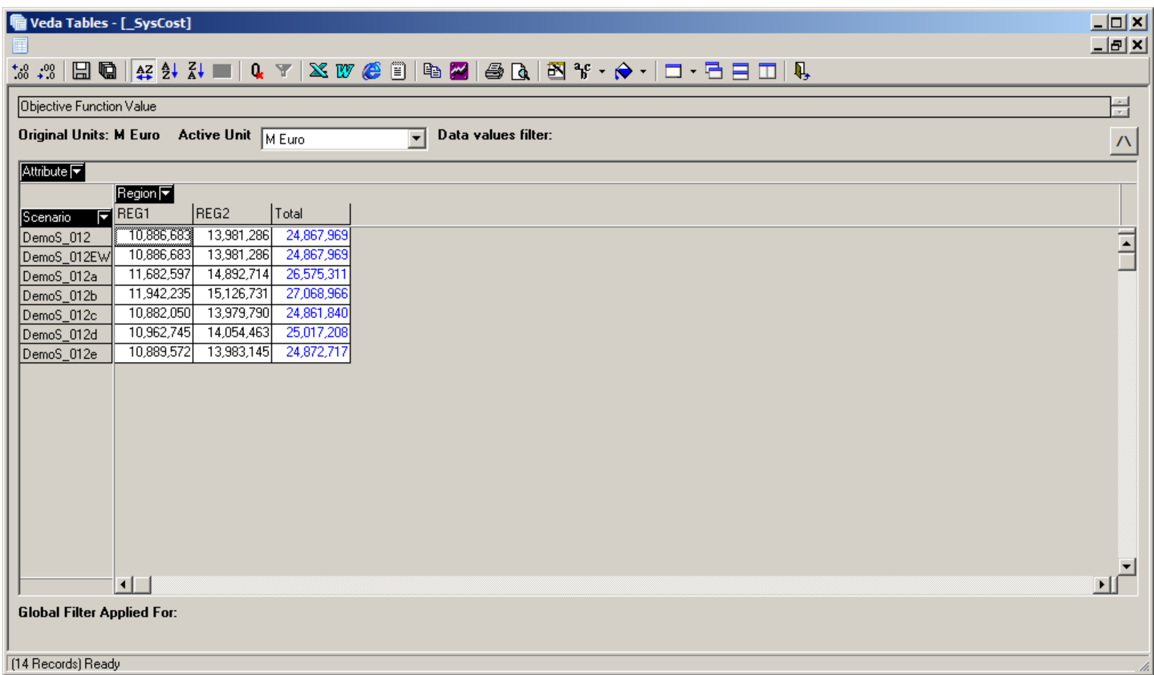


Figure 14: Viewing the _*SysCost* table

The order of multiple dimensions in row and column positions determines the order in which data is nested. (See Section 2.3.6 for an example.)

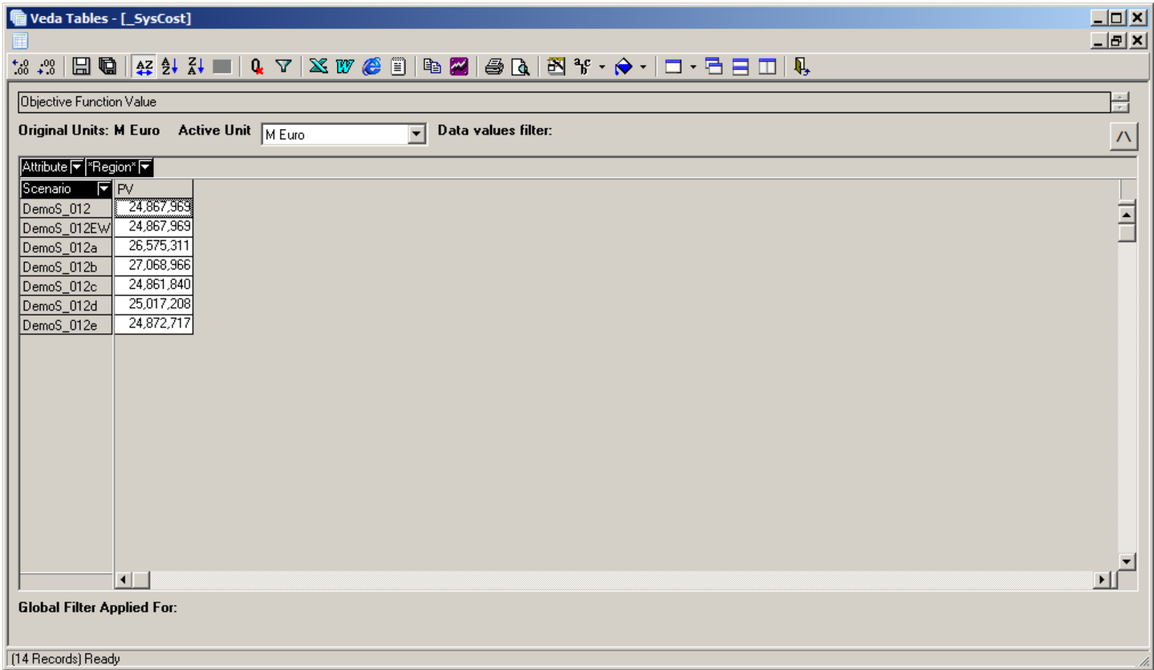


Figure 15: The _*SysCost* table with *Region* summed

With this table, we can verify that our test run of DemoS_012 has produced the same objective value, and hence the same solution as the original run. However, let us use this simple table to explore a few more features in the View window.

2.3.1 Sub-totals

Dragging *Region* back to its original column position allows us to see (as in Figure 14) that the creator of this table opted to have the displayed values summed over *Region.* This option is toggled by right-clicking while the cursor lies in the row or column listing for the desired dimension (in this case, on one of the region names) and selecting **Totals** from the menu that appears. As with dimensions held on the page (rather than in rows or columns), the user must take care that the items summed produce a meaningful total.

2.3.2 Changing item display from name to description

From the same menu, one may choose the **Display Type** for the selected dimension. As shown in Figure 16, the **Display Type** menu offers the option to view items in the selected dimension by their *Code* (short name), *Description*, or both, in either order. Switching *Scenarios* here to a display type that includes the description may help us keep track of which scenario is which, for example. Note that when only the *Code* or *Description* is selected, the unselected option appears as a tooltip when you hover over a scenario entry.

2.3.3 Unit conversion

If a table's native *units* and relevant *unit conversion factors* have been specified, a table's display units can be converted in the cube (and saved, if desired). Clicking on the drop-down menu next to **Active Unit** above the top of the table reveals that, in addition to the table's native *M Euro* unit, another relevant unit (and its conversion factor with MEuro) has been defined: *B Euro*.

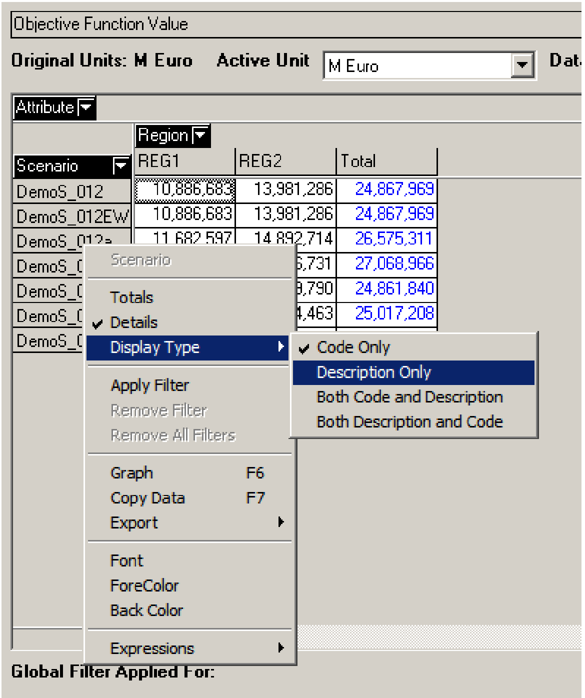


Figure 16: Choosing the display type

Selecting *B Euro* from the drop-down menu converts all data in the table. Defining units and conversion factors is discussed in Section 5.

2.3.4 View window toolbar

The view window has a toolbar with several icons. The name of each button is shown when the cursor is hovered over the corresponding button. Moving from left to right, here is a brief description of the tool bar buttons:

- Decrease or increase the number of digits after decimal;
- Save table changes. The first button saves the highlighted table only, and the next button saves all tables currently on screen (in the case of multiple tables). If changes were made while in view mode, the user is asked to provide a new name or to use the old name for the table (writing over its previous specifications). If you begin entering a new name, a checkbox will appear to allow you to retain the old table (with its original name) or substitute the new name for the old. Note that these save options save the *definition* of the table, as modified in view mode, as well as any layout modifications. They do not save the *data values* in the table. These are regenerated from the table definition each time the table is opened by choosing **View Table(s)** in the main window, unless the **Save Cube Files** user option has been selected, as described in Section 2.3.7.
- Sort the table. **Ascending** and **Descending** sort table rows by the data in the first column (only), while **Default** returns to sort by the row name. This feature can be useful when trying to identify the largest flows/costs in a table;
- Hide/Show blank/zero values;

- Export table (four options are provided: Excel, Word, Html, Text, of which Excel is the recommended option) or copy a portion of the table on to the clipboard. When the table is exported to a file, the date and time are appended to the table's name to constitute the name of the saved file. See Section 6 for more on using VEDA-BE in conjunction with Excel.
- Produce a graph of a selected table area: first block the table cells desired, then press the graph button (discussed further in Section 2.3.8);
- Print and Preview the current table;
- Set layout preferences (appearance, font, colors);
- Set the way multiple windows are shown (cascade, horizontally, vertically). These options are useful when several tables are viewed simultaneously.
- Close the view window. (You may also close individual tables by clicking on the X in the upper righthand corner of that table's window).

The /\ button located on the far right below the toolbar toggles showing and hiding the table's description in the header.

2.3.5 Global Filtering

While we could easily verify, in this simple table, that the objective function values for the two scenarios we wished to compare were identical, in fact we made this task harder for ourselves by including four extraneous scenarios in the table. In general, efficient model results analysis is all about controlling *and limiting* what you see to only the essential items. The ***global filter*** feature supports this practice.

Global filters are set in the main VEDA-BE window. Close the current table (by clicking on the **X** in the upper right corner of the window or by clicking the **Exit** icon). Because we have modified the table layout, we are prompted to choose whether to **Save** our changes, **Exit** without saving, or **Cancel** and keep the table open. As with the Save toolbar buttons described in Section 2.3.4, clicking the **Save** button will give you the further choice of saving with a new name (optionally retaining the original table) or writing over the original table.

Back in the main window, there are two methods to set a global filter. Both start from the **Scenarios** tab:

- Option 1: hold down the **Control** key, and click on the desired scenarios in the scenarios list. Their rows will turn yellow, to indicate that they have been set within the global filter.
- Option 2: right click anywhere within the scenarios list, and choose **Global Filter** from the menu that appears. This brings up the selection window shown in Figure 17. Here you may select individual scenarios by clicking on them (turning them **bold blue**), filter scenarios by code or description, and **Select** or **Unselect All** from the filtered list.

Setting the global filter for the original and re-run *DemoS_012* scenarios results in the window appearance in Figure 18. The global filter status at the bottom now shows us that the global filter has been applied for the selected scenarios. The checkboxes allow you to remove the filters by dimension (if filters have been set for one or more dimensions) while retaining their specifications. The filters may be reengaged later by re-checking the box.

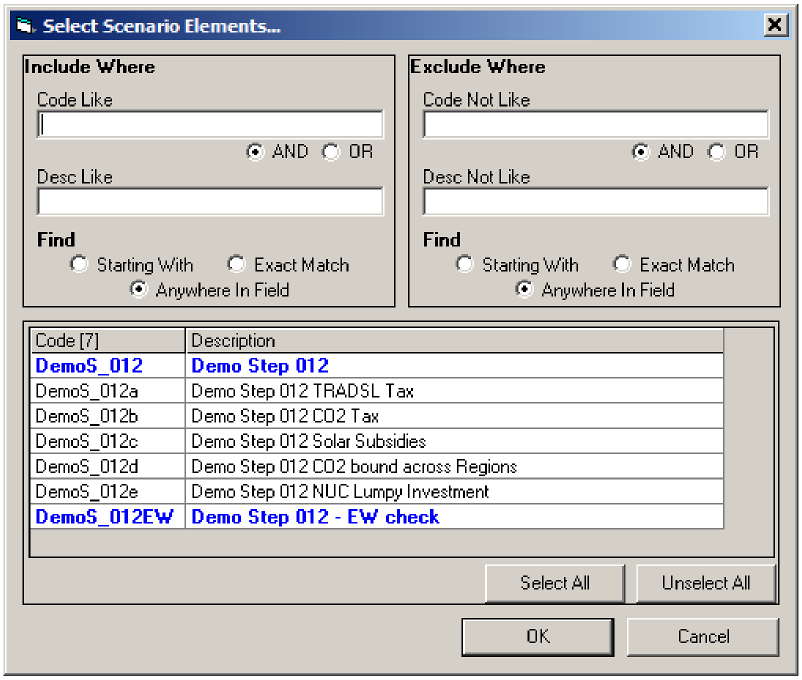


Figure 17: Selecting scenarios for a global filter

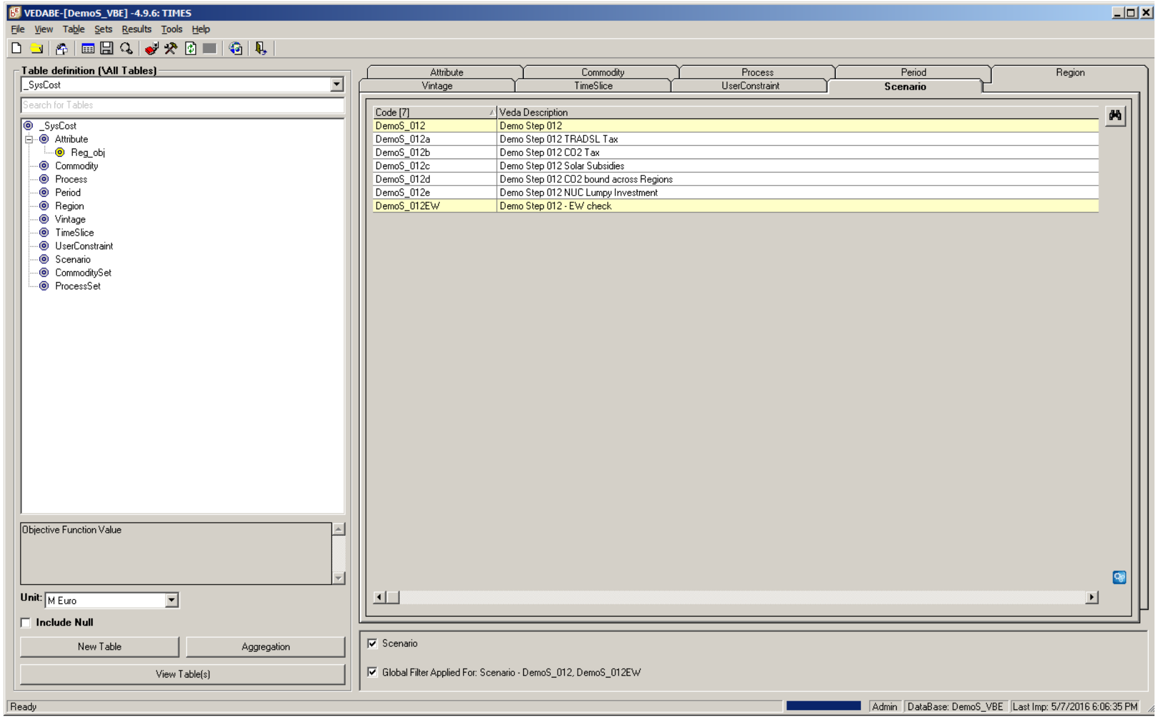


Figure 18: Main window with global filter set on scenarios

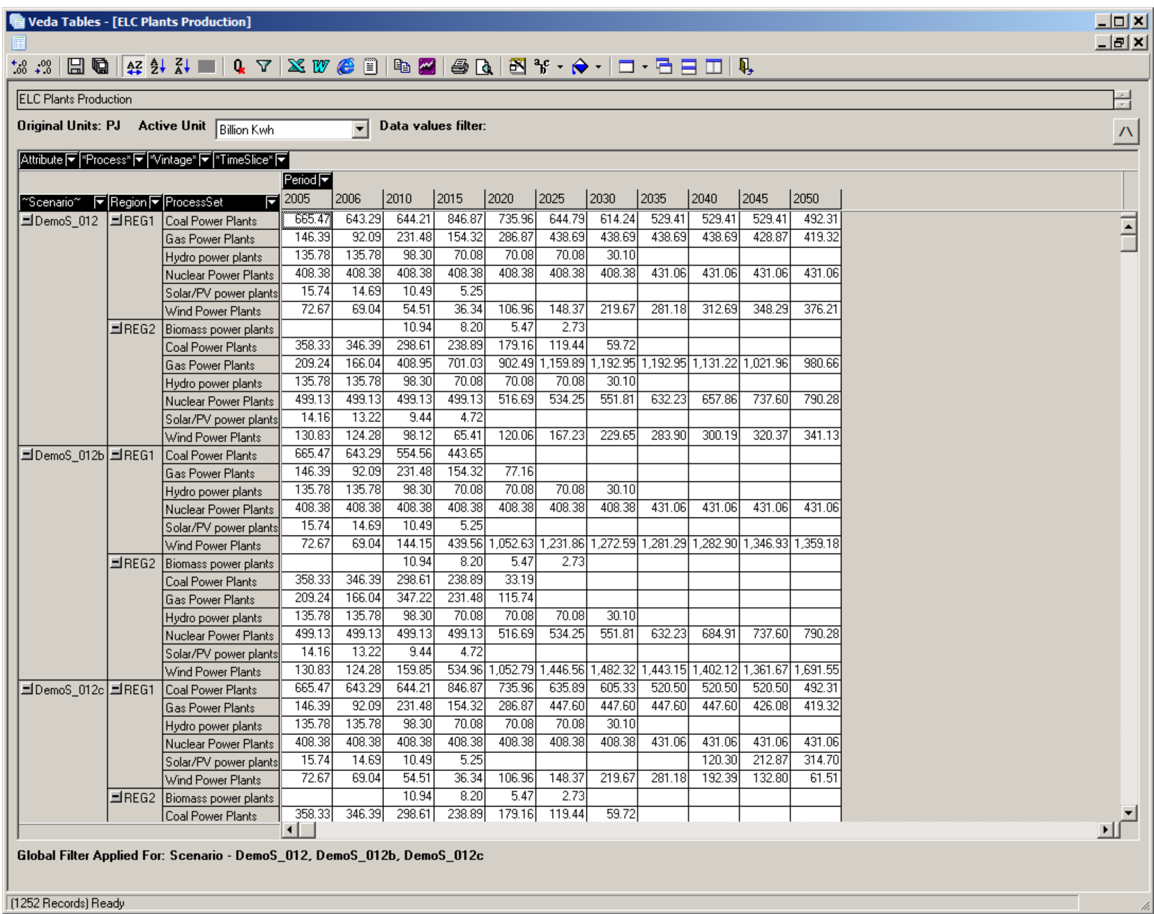
When we click **View Table(s)** with the filter engaged, we now see only the two selected scenarios. This filter will remain active for any subsequent tables created, until it is removed. Tables already open will not be affected.

Because scenarios and scenario names change often, and the global filter facilitates viewing only the scenarios you choose, it is recommended to *never* include scenarios in a table definition, and to simply use the global filter to select scenarios for viewing as you work. Global filters may also be set for other dimensions: *Region* may be convenient for analyzing results in large multi-region models, and *Attribute* can be useful in conjunction with the ExRES feature described in Section 2.4, to limit results shown in the full attributes mode.

To save a global filter for later re-use, click on the **Gear** icon in the lower right corner of the main window. Choose **New**, enter a name for the filter, and click the “check” icon. The filter may now be selected at any time by clicking on its name from the drop-down menu under the Gear icon.

2.3.6 Additional functions in the table view

To fully appreciate the power of the VEDA-BE table viewer, we need to work with a more complex table. Set a global filter for scenarios *DemoS_012*, *DemoS_012b* (carbon tax), and *DemoS_012c* (solar subsidy), and select the ***ELC Plants Production*** table. This table consists of the attribute ***Var_Act***, or process activity, and several process sets, each named *ELExxx*. Click **View Table(s)** to open the table view. When we first open the table, it looks like Figure 19.

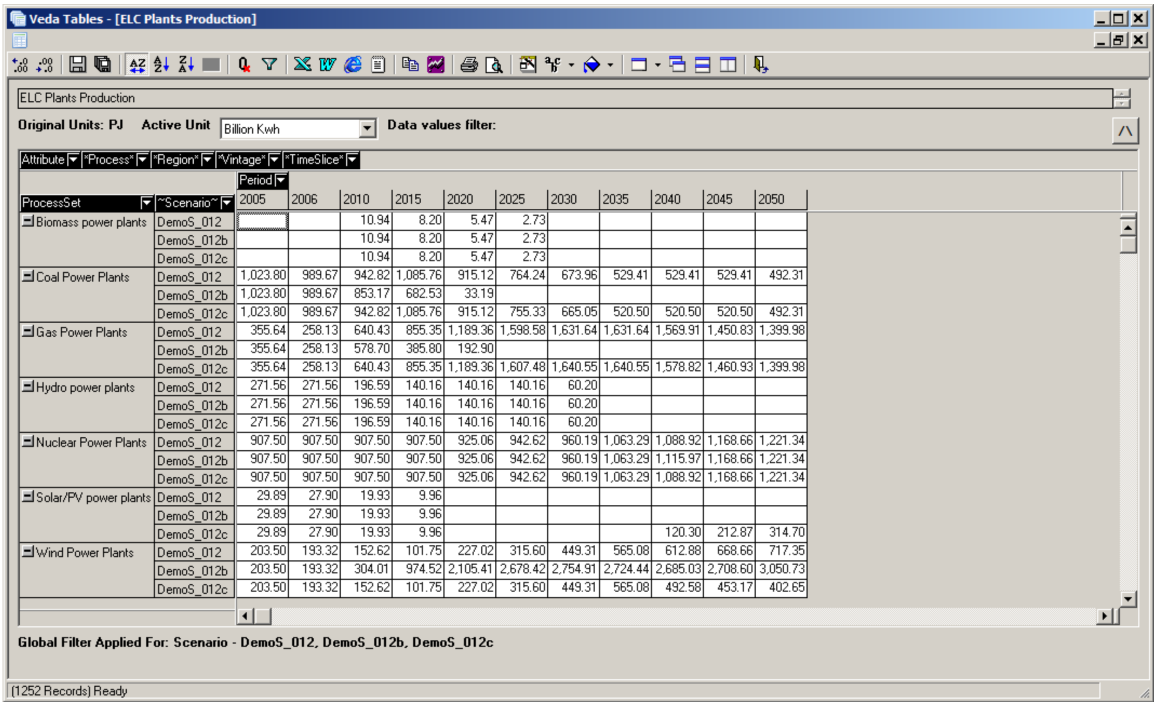


~Scenario~	Region	ProcessSet	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_012	REG1	Coal Power Plants	665.47	643.29	644.21	846.87	735.96	644.79	614.24	529.41	529.41	529.41	492.31
		Gas Power Plants	146.39	92.09	231.48	154.32	286.87	438.69	438.69	438.69	438.69	428.87	419.32
		Hydro power plants	135.78	135.78	98.30	70.08	70.08	70.08	30.10				
		Nuclear Power Plants	408.38	408.38	408.38	408.38	408.38	408.38	408.38	431.06	431.06	431.06	431.06
		Solar/PV power plants	15.74	14.69	10.49	5.25							
		Wind Power Plants	72.67	69.04	54.51	36.34	106.96	148.37	219.67	281.18	312.69	348.29	376.21
	REG2	Biomass power plants			10.94	8.20	5.47	2.73					
		Coal Power Plants	358.33	346.39	298.61	238.89	179.16	119.44	59.72				
		Gas Power Plants	209.24	166.04	408.95	701.03	902.49	1,159.89	1,192.95	1,192.95	1,131.22	1,021.96	980.66
		Hydro power plants	135.78	135.78	98.30	70.08	70.08	70.08	30.10				
		Nuclear Power Plants	499.13	499.13	499.13	499.13	516.69	534.25	551.81	632.23	657.86	737.60	790.28
		Solar/PV power plants	14.16	13.22	9.44	4.72							
		Wind Power Plants	130.83	124.28	98.12	65.41	120.06	167.23	229.65	283.90	300.19	320.37	341.13
DemoS_012b	REG1	Coal Power Plants	665.47	643.29	554.56	443.65							
		Gas Power Plants	146.39	92.09	231.48	154.32	77.16						
		Hydro power plants	135.78	135.78	98.30	70.08	70.08	70.08	30.10				
		Nuclear Power Plants	408.38	408.38	408.38	408.38	408.38	408.38	408.38	431.06	431.06	431.06	431.06
		Solar/PV power plants	15.74	14.69	10.49	5.25							
		Wind Power Plants	72.67	69.04	144.15	439.56	1,052.63	1,231.86	1,272.59	1,281.29	1,282.90	1,346.93	1,359.18
	REG2	Biomass power plants			10.94	8.20	5.47	2.73					
		Coal Power Plants	358.33	346.39	298.61	238.89	33.19						
		Gas Power Plants	209.24	166.04	347.22	231.48	115.74						
		Hydro power plants	135.78	135.78	98.30	70.08	70.08	70.08	30.10				
		Nuclear Power Plants	499.13	499.13	499.13	499.13	516.69	534.25	551.81	632.23	684.91	737.60	790.28
		Solar/PV power plants	14.16	13.22	9.44	4.72							
		Wind Power Plants	130.83	124.28	159.85	534.96	1,052.79	1,446.56	1,482.32	1,443.15	1,402.12	1,361.67	1,691.55
DemoS_012c	REG1	Coal Power Plants	665.47	643.29	644.21	846.87	735.96	635.89	605.33	520.50	520.50	520.50	492.31
		Gas Power Plants	146.39	92.09	231.48	154.32	286.87	447.60	447.60	447.60	447.60	426.08	419.32
		Hydro power plants	135.78	135.78	98.30	70.08	70.08	70.08	30.10				
		Nuclear Power Plants	408.38	408.38	408.38	408.38	408.38	408.38	408.38	431.06	431.06	431.06	431.06
		Solar/PV power plants	15.74	14.69	10.49	5.25					120.30	212.87	314.70
		Wind Power Plants	72.67	69.04	54.51	36.34	106.96	148.37	219.67	281.18	192.39	132.80	61.51
	REG2	Biomass power plants			10.94	8.20	5.47	2.73					
		Coal Power Plants	358.33	346.39	298.61	238.89	179.16	119.44	59.72				

Figure 19: The *ELC Plants Production* table, as first opened

We have *Scenario*, *Region*, and *ProcessSet* on rows, and *Period* on columns. The *ProcessSet* display has been set to *Description Only*, and the units have been set to *Billion Kwh*, from their original *PJ*.

Let us first drag *Region* up above to simplify the table. Then drag *Scenario* to the right of *ProcessSet.* This makes the table appear as in Figure 20 and allows us to compare power plant activity by type across scenarios more easily. We see, for example, that coal and gas plants entirely shut down by 2025 in the *DemoS_012b* (carbon tax) scenario, replaced by wind and solar plants, and that in the *DemoS_012c* (solar subsidy) scenario, solar plants take market share primarily away from wind plants, leaving the others largely unchanged.



ProcessSet	~Scenario~	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
Biomass power plants	DemoS_012			10.94	8.20	5.47	2.73					
	DemoS_012b			10.94	8.20	5.47	2.73					
	DemoS_012c			10.94	8.20	5.47	2.73					
Coal Power Plants	DemoS_012	1,023.80	989.67	942.82	1,085.76	915.12	764.24	673.96	529.41	529.41	529.41	492.31
	DemoS_012b	1,023.80	989.67	853.17	682.53	33.19						
	DemoS_012c	1,023.80	989.67	942.82	1,085.76	915.12	755.33	665.05	520.50	520.50	520.50	492.31
Gas Power Plants	DemoS_012	355.64	258.13	640.43	855.35	1,189.36	1,598.58	1,631.64	1,631.64	1,569.91	1,450.83	1,399.98
	DemoS_012b	355.64	258.13	578.70	385.80	192.90						
	DemoS_012c	355.64	258.13	640.43	855.35	1,189.36	1,607.48	1,640.55	1,640.55	1,578.82	1,460.93	1,399.98
Hydro power plants	DemoS_012	271.56	271.56	196.59	140.16	140.16	140.16	60.20				
	DemoS_012b	271.56	271.56	196.59	140.16	140.16	140.16	60.20				
	DemoS_012c	271.56	271.56	196.59	140.16	140.16	140.16	60.20				
Nuclear Power Plants	DemoS_012	907.50	907.50	907.50	907.50	925.06	942.62	960.19	1,063.29	1,088.92	1,168.66	1,221.34
	DemoS_012b	907.50	907.50	907.50	907.50	925.06	942.62	960.19	1,063.29	1,115.97	1,168.66	1,221.34
	DemoS_012c	907.50	907.50	907.50	907.50	925.06	942.62	960.19	1,063.29	1,088.92	1,168.66	1,221.34
Solar/PV power plants	DemoS_012	29.89	27.90	19.93	9.96							
	DemoS_012b	29.89	27.90	19.93	9.96							
	DemoS_012c	29.89	27.90	19.93	9.96					120.30	212.87	314.70
Wind Power Plants	DemoS_012	203.50	193.32	152.62	101.75	227.02	315.60	449.31	565.08	612.88	668.66	717.35
	DemoS_012b	203.50	193.32	304.01	974.52	2,105.41	2,678.42	2,754.91	2,724.44	2,685.03	2,708.60	3,050.73
	DemoS_012c	203.50	193.32	152.62	101.75	227.02	315.60	449.31	565.08	492.58	453.17	402.65

Figure 20: The *ELC Plants Production* table, as rearranged

To get additional detail, pull *Process* down to the row headers, between *ProcessSet* and *Region*, so that we may still compare scenarios directly against each other. We see that most of the process sets in the table contain more than one process that operates in these scenarios, generally an existing process that retires partway through the model horizon, and a new process whose activity grows over time. (We will see in Section 3 how these process sets are created.)

We can focus the view further by removing some items temporarily from view. Each (black) dimension header contains a triangle revealing a drop-down menu of all items under that dimension in the table definition when clicked. Individual items can be deselected and reselected by clearing and re-checking the checkbox next to their names, and the entire list can be selected or cleared by holding down the **Control** key while clicking a single checkbox.

For example, by selecting only *Solar/PV* and *Wind power plants* from the *ProcessSet* list, and only *DemoS_012* and *DemoS_012c* from the *Scenarios* list, and then pulling both *ProcessSet* and *Process* up above, we now see the total generation from wind and solar plants in these two scenarios, and can see just how close the totals are (Figure 21).

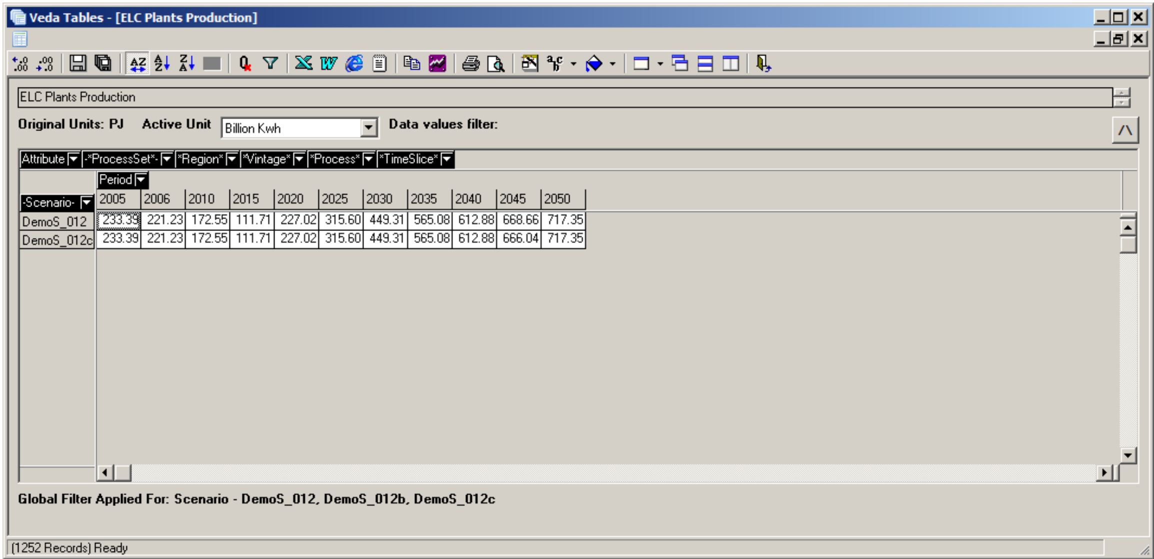


Figure 21: The *ELC Plants Production* table, rearranged to show total solar and wind plant activity

Note that if you choose to **Save** the table after these modifications, both the process set selections and the scenario selections will be saved in the table definition, because explicit choices have been made for both.

An alternative to removing items from a table while managing the volume of information displayed is to selectively collapse and open sections of the table using the "+" and "-" icons next to elements that have dimensions nested beneath them. For example, returning to the original ***ELC Plants Production*** view shown in Figure 19 and clicking "-" icon next to *DemoS_012* in the Scenarios column collapses all the results for that scenario, showing just the total over all collapsed rows. You may collapse all the table sections associated with a particular dimension by right clicking on any item in that dimension and choosing **Details** from the drop-down menu that appears. You may then selectively reopen desired sections by clicking on the relevant "+" icon.

2.3.7 Saving data cubes

The table **Save** operation described in the previous section saves the *definitions* and *layout options* for tables, which will be recreated from these saved specifications the text time the table is viewed. Another kind of "save" operation is possible: saving the *data cube* itself as it has been created. The option to do so is a global user preference, set by choosing **Options** from the **Tools** menu in the main window, and then checking the box labeled **Save Cube Files**. If selected, this option will reopen an existing table almost instantly, as long as the content of the table has not changed since the last request. VEDA-BE keeps track of any potential changes and recreates the cube when necessary. This option can greatly speed up work if you expect to view the same tables repeatedly.

When viewing saved cube files, the pivot, select and deselect, and subtotal options are all available, but swapping code and description is not. All cubes are deleted whenever scenarios are imported or deleted. Choosing **Delete Cube File(s)** from the **Tools** menu will manually force a delete of all saved cubes.

2.3.8 The VEDA-BE graphing function

A final feature available in the view window is the graphing function. An entire data table or any portion thereof may be graphed by selecting the desired data points and either right clicking and choosing **Graph** from the drop-down menu or clicking the graph button on the toolbar. (If no data is selected, the entire table will be graphed.) For example, returning the *ELC Plants Production* table to its original Figure 19 configuration, selecting all the data rows for *REG1* in scenario *DemoS_012*, and clicking the **Graph** button yields the graph shown in Figure 22.

The toolbar at the top of the **Graph** window contains options to change the type of graph, switch rows and columns, copy to the clipboard or export to Excel, and toggle the legend, as well as a **Graph Settings window** with numerous options to format the graph, as well as to hide or remove individual series. These simple graphs can help the user quickly get a visual feel for one or more data series, but for creating and maintaining more complex or customized graphs for presentations and reports, the **Update Excel File** feature discussed in Section 6.3 is recommended.

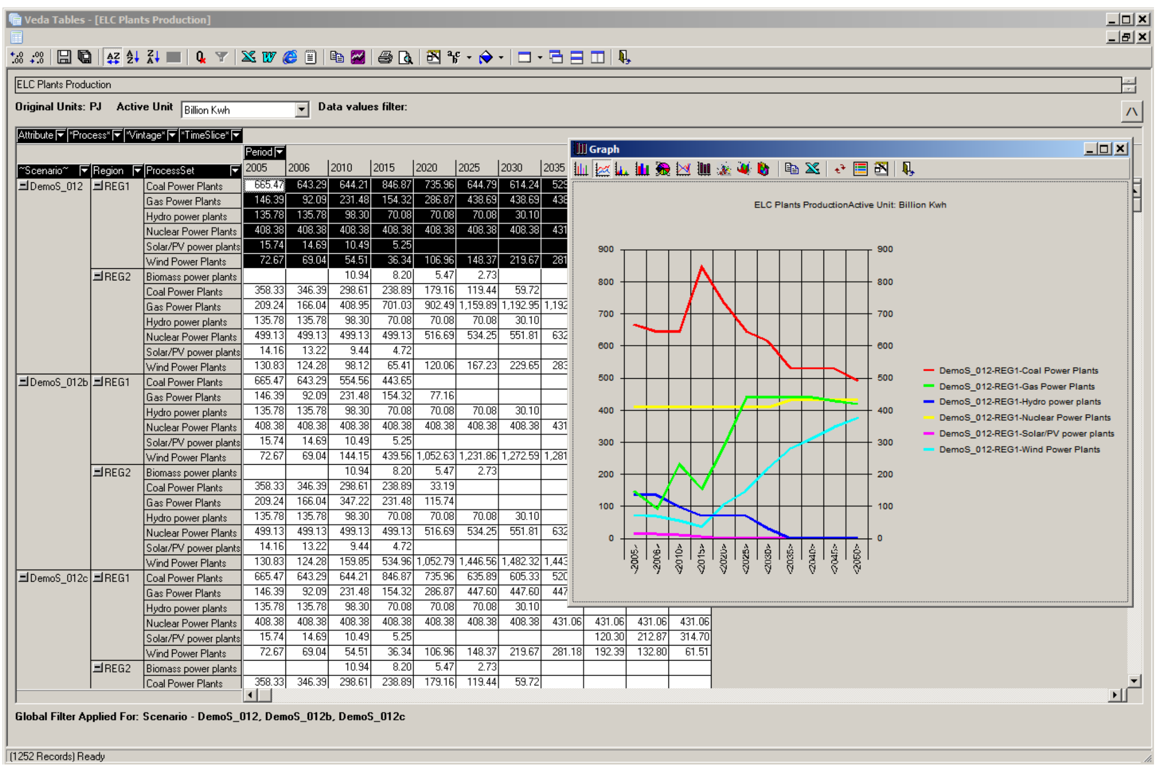
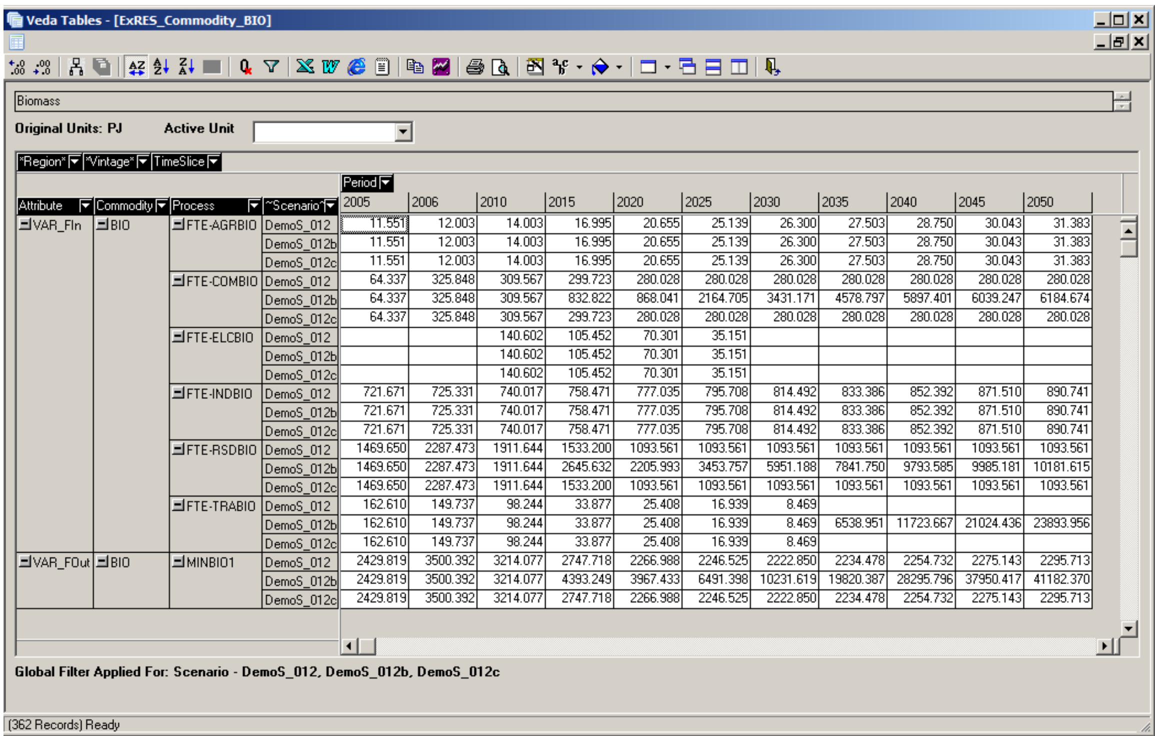


Figure 22: Creating a simple graph in VEDA-BE

2.4 Viewing results in the ExRES

It is not necessary to define a table every time you wish to view results. Another equally powerful, but quite different method for viewing results is the **ExRES** (for "Extended RES"). This tool lets you zero in on model results *as they are connected in the model's Reference Energy System (RES).* Starting from the main window, the ExRES can be invoked for any process or process set, commodity or commodity set, or user constraint to rapidly get results information for that item or set.

For example, to view results data for commodity *BIO (Biomass)*, right click on its listing on the *Commodity* tab and choose **ExRES** from the menu that appears. This will create the table shown in Figure 23 for all scenarios in the currently set global filter. It shows all data associated with the ***VAR_Fin*** (*Commodity Consumption by Process*, or uses) and ***VAR_Fout*** (*Commodity Production by Process*, or sources) attributes for the selected commodity. (Had we launched the ExRES from a process, *VAR_Fin* and *VAR_Fout* would still be the displayed attributes, now representing all inputs to and outputs from the selected process.)


Veda Tables - [ExRES_Commodity_BIO]

Biomass

Original Units: PJ Active Unit

Region *Vintage* TimeSlice

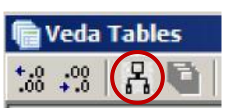
Attribute	Commodity	Process	~Scenario~	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
VAR_FIn	BIO	FTE-AGRBIO	DemoS_012	11.551	12.003	14.003	16.995	20.655	25.139	26.300	27.503	28.750	30.043	31.383
			DemoS_012b	11.551	12.003	14.003	16.995	20.655	25.139	26.300	27.503	28.750	30.043	31.383
			DemoS_012c	11.551	12.003	14.003	16.995	20.655	25.139	26.300	27.503	28.750	30.043	31.383
		FTE-COMBIO	DemoS_012	64.337	325.848	309.567	299.723	280.028	280.028	280.028	280.028	280.028	280.028	280.028
			DemoS_012b	64.337	325.848	309.567	832.822	868.041	2164.705	3431.171	4578.797	5897.401	6039.247	6184.674
			DemoS_012c	64.337	325.848	309.567	299.723	280.028	280.028	280.028	280.028	280.028	280.028	280.028
		FTE-ELCBIO	DemoS_012			140.602	105.452	70.301	35.151					
			DemoS_012b			140.602	105.452	70.301	35.151					
			DemoS_012c			140.602	105.452	70.301	35.151					
		FTE-INDBIO	DemoS_012	721.671	725.331	740.017	758.471	777.035	795.708	814.492	833.386	852.392	871.510	890.741
			DemoS_012b	721.671	725.331	740.017	758.471	777.035	795.708	814.492	833.386	852.392	871.510	890.741
			DemoS_012c	721.671	725.331	740.017	758.471	777.035	795.708	814.492	833.386	852.392	871.510	890.741
		FTE-RSDBIO	DemoS_012	1469.650	2287.473	1911.644	1533.200	1093.561	1093.561	1093.561	1093.561	1093.561	1093.561	1093.561
			DemoS_012b	1469.650	2287.473	1911.644	2645.632	2205.993	3453.757	5951.188	7841.750	9793.585	9985.181	10181.615
			DemoS_012c	1469.650	2287.473	1911.644	1533.200	1093.561	1093.561	1093.561	1093.561	1093.561	1093.561	1093.561
		FTE-TRABIO	DemoS_012	162.610	149.737	98.244	33.877	25.408	16.939	8.469				
			DemoS_012b	162.610	149.737	98.244	33.877	25.408	16.939	8.469	6538.951	11723.667	21024.436	23893.956
			DemoS_012c	162.610	149.737	98.244	33.877	25.408	16.939	8.469				
VAR_FOut	BIO	MINBIO1	DemoS_012	2429.819	3500.392	3214.077	2747.718	2266.988	2246.525	2222.850	2234.478	2254.732	2275.143	2295.713
			DemoS_012b	2429.819	3500.392	3214.077	4393.249	3967.433	6491.398	10231.619	19820.387	28295.796	37950.417	41182.370
			DemoS_012c	2429.819	3500.392	3214.077	2747.718	2266.988	2246.525	2222.850	2234.478	2254.732	2275.143	2295.713

Global Filter Applied For: Scenario - DemoS_012, DemoS_012b, DemoS_012c

(362 Records) Ready

Figure 23: The ExRES for commodity *BIO*

This input-and-output-flows view of the ExRES may be toggled with a full results details view by clicking the third icon from left in the view window menu bar.



Doing so will display *all* the results data associated with the selected item. Clicking the icon (which changes appearance in this mode to indicate that additional data is being shown) again will return to the flows-only view. The full results ExRES view may be refined by setting a global filter on desired attributes. For example, setting a global filter on ***EQ_CombalM***, ***VAR_Fin***, and ***VAR_FOut*** can aid in fuel chain debugging by allowing you to see only commodity production, consumption, and prices.

The ExRES view allows you to move up and downstream in the model RES to trace the sources and/or uses of commodities, making it a key tool for model debugging. Clicking on any process or commodity name in the table view will launch an ExRES window for that item. This may be pursued as far up or down the RES network as needed.

For example, clicking on *FTE-RSDBIO* in the table shown in Figure 23 reveals that this process merely changes the name of *BIO* to *RSDBIO* for purposes of sector fuel accounting (Figure 24). Clicking on *RSDBIO* then allows us to see that the primary use of increased residential biomass in the carbon tax scenario is a new space heating technology (where the tooltip of the process description has been shown in Figure 25 by hovering over the process name *RSHNBIO1*).

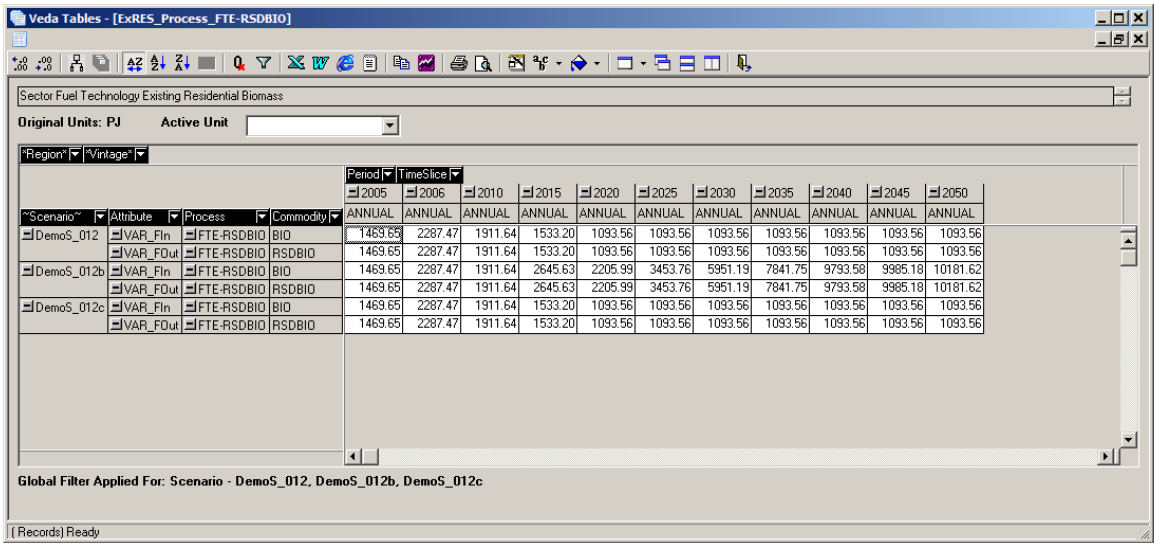


Figure 24: The ExRES for *FTE-RSDBIO*

To make it readily apparent when you are using an ExRES rather than table view, it is recommended to use the **view window** formatting tool, available from the **paint bucket icon**  in the view window toolbar to make the formatting of ExRES views distinct from table views. Figure 25 shows an example, where the table body (**Back Color** menu option) and items lists (**Headings Back Color** menu option) background color defaults have been changed. Once set in a single ExRES view, these formats will be active for all future ExRES views launched in the database.

The ExRES can also be invoked within the table view mode, by right clicking on any process, commodity, process set, or commodity set name in the table and choosing **ExRES** from the menu that appears (see Figure 26). This is an excellent way to further investigate the results you see in any table.

From the main window there is an additional ExRES function available by right clicking on any commodity, process, or user constraint name: the **Related ExRES**. This menu item allows you to generate an ExRES as follows:

- If selected for a commodity, all *Processes* consuming or producing that commodity, or all *User constraints* involving that commodity;
- If selected for a process, all *Commodities* being consumed or produced by that process, or all *User constraints* involving that process; and
- If selected for a user constraint, all *Commodities* or *Processes* involved in that user constraint.

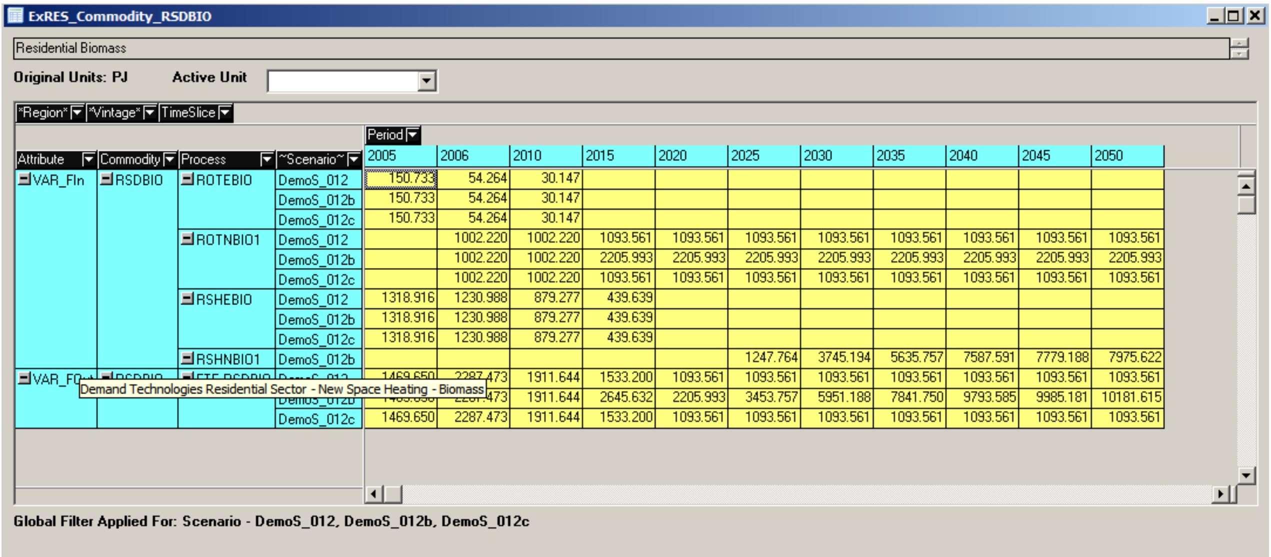


Figure 25: The ExRES for RSDBIO, with background coloring options set

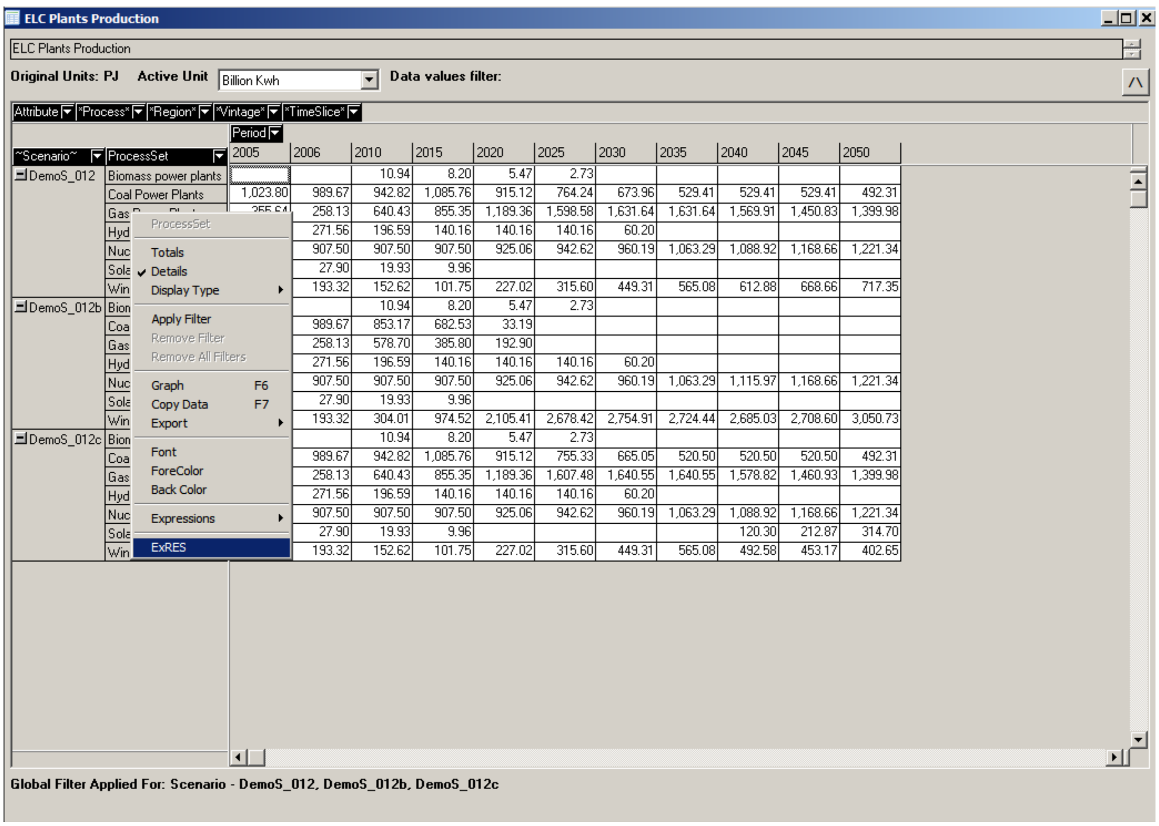
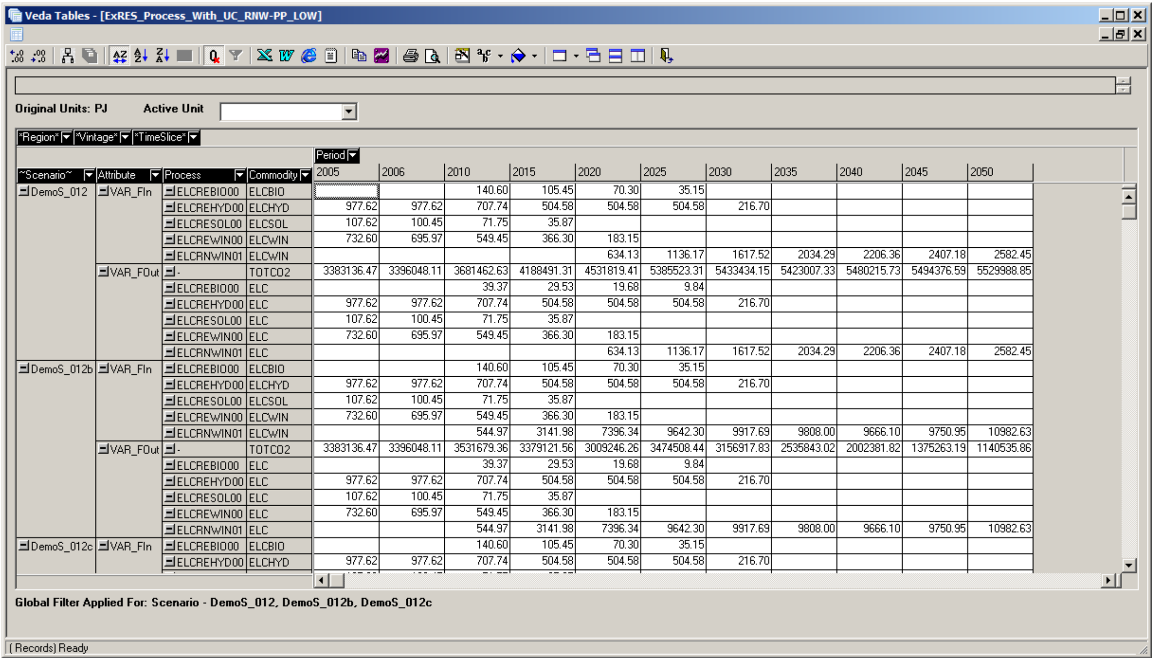


Figure 26: Invoking the ExRES within a table view

For example, right clicking on *UC_RNW-PP_LOW* on the *UserConstraint* tab and choosing **Related ExRES of → Process** generates the ExRES shown in Figure 27, displaying all inputs and outputs of the processes involved in that user constraint.



Veda Tables - [ExRES_Process_With_UC_RNW-PP_LOW]

Original Units: PJ   Active Unit

Region *Vintage* *TimeSlice*

Period

~Scenario~	Attribute	Process	Commodity	2005	2006	2010	2015	2020	2025	2030	2035	2040	2045	2050
DemoS_012	VAR_FIn	ELCREBIO00	ELCBIO			140.60	105.45	70.30	35.15					
		ELCREHYD00	ELCHYD	977.62	977.62	707.74	504.58	504.58	504.58	216.70				
		ELCRESOL00	ELCSOL	107.62	100.45	71.75	35.87							
		ELCREWIN00	ELCWIN	732.60	695.97	549.45	366.30	183.15						
		ELCRNWIN01	ELCWIN					634.13	1136.17	1617.52	2034.29	2206.36	2407.18	2582.45
	VAR_FOut	-	TOTCO2	3383136.47	3396048.11	3681462.63	4188491.31	4531819.41	5385523.31	5433434.15	5423007.33	5480215.73	5494376.59	5529988.85
		ELCREBIO00	ELC			39.37	29.53	19.68	9.84					
		ELCREHYD00	ELC	977.62	977.62	707.74	504.58	504.58	504.58	216.70				
		ELCRESOL00	ELC	107.62	100.45	71.75	35.87							
		ELCREWIN00	ELC	732.60	695.97	549.45	366.30	183.15						
		ELCRNWIN01	ELC					634.13	1136.17	1617.52	2034.29	2206.36	2407.18	2582.45
DemoS_012b	VAR_FIn	ELCREBIO00	ELCBIO			140.60	105.45	70.30	35.15					
		ELCREHYD00	ELCHYD	977.62	977.62	707.74	504.58	504.58	504.58	216.70				
		ELCRESOL00	ELCSOL	107.62	100.45	71.75	35.87							
		ELCREWIN00	ELCWIN	732.60	695.97	549.45	366.30	183.15						
		ELCRNWIN01	ELCWIN			544.97	3141.98	7396.34	9642.30	9917.69	9808.00	9666.10	9750.95	10982.63
	VAR_FOut	-	TOTCO2	3383136.47	3396048.11	3531679.36	3379121.56	3009246.26	3474508.44	3156917.83	2535843.02	2002381.82	1375263.19	1140535.86
		ELCREBIO00	ELC			39.37	29.53	19.68	9.84					
		ELCREHYD00	ELC	977.62	977.62	707.74	504.58	504.58	504.58	216.70				
		ELCRESOL00	ELC	107.62	100.45	71.75	35.87							
		ELCREWIN00	ELC	732.60	695.97	549.45	366.30	183.15						
		ELCRNWIN01	ELC			544.97	3141.98	7396.34	9642.30	9917.69	9808.00	9666.10	9750.95	10982.63
DemoS_012c	VAR_FIn	ELCREBIO00	ELCBIO			140.60	105.45	70.30	35.15					
		ELCREHYD00	ELCHYD	977.62	977.62	707.74	504.58	504.58	504.58	216.70				

Global Filter Applied For: Scenario - DemoS_012, DemoS_012b, DemoS_012c

(Records) Ready

Figure 27: The Related ExRES for *UC_RNW-PP_LOW*

3 Sets: A powerful tool

A key building block in the construction of VEDA-BE tables is the ability to create user-defined sets of model processes and commodities. While simple tables, such as the *_SysCost* table viewed in Section 2.3, do not require them, most VEDA-BE tables make use of process and/or commodity sets. A set is a group of like items grouped *by rule* in order to view their results together (and in VEDA-FE, to apply common input data.) Upon creation, a VEDA-BE database contains only those sets passed from the model by the GDX2VEDA utility, such as process sets *ELE* (electric power plants), *DMD* (demand devices), and *PRE* (energy processes), and commodity sets *DEM* (demands), *ENV* (emissions), and *NRG* (energy carriers).

Because a TIMES model may contain hundreds, or even thousands, of processes, the ability to group items into sets using logical rules becomes an essential component in the analysis of large models.

Remark: Note that the importance of sets has implications for model *design*. Structured naming conventions for items must be developed *and scrupulously adhered to* in order to make use of the ability to create sets using rules based on item names. Item descriptions may also be used in set creation, and so should not be overlooked for holding, in carefully structured form, information needed – or information that may be useful in the future – to create sets[5].

3.1 The set definition window

Set definitions can be viewed, edited, and created in the **set definition** window, which can be accessed for an existing set by selecting **Edit Set** from the **Sets** menu or right clicking on a set name on the **Process** tab and selecting **Edit/View Set** from the menu that appears, or, for a new set, by choosing **New Set(s)** from the **Sets** menu. A combo box in the upper lefthand corner of the allow the user to switch between working with commodity and process sets.

The set definition window allows sets to be defined through rules that include and/or exclude items based on:

- membership in other sets,
- characters in the item name and/or description, and
- for process sets, input and output commodities.

In specifying rules for item names and descriptions, the wildcards “?”, “_”, and “*” may be used to specify a single character wildcard (question mark and underscore[6]) and a general wildcard (asterisk) for any number of characters, respectively.

The rules that define any existing set, and its resulting members, can most easily be examined by right clicking on its listing on the **Commodity** or **Process** tab and choosing **Edit/View Set** from the menu that appears. Note that to manipulate the set definition form in any way, including

---

[5] See Wright and Kanudia (2015), “Highly Detailed TIMES Modeling to Analyze Interactions between Air Quality and Climate Regulations in the United States,” in Giannakidis, G., Labriet, M., Ó Gallachóir, B., Tosato, G. (Eds.), *Informing Energy and Climate Policies using Energy Systems Models: Insights from Scenario Analysis Increasing the Evidence Base*, Springer, for one example of how item descriptions have been packed with information in the FACETS model. Available at: http://facets-model.com/mats/.

[6] Note that to filter for a underscore in an item name or description, it is necessary to enclose it in square brackets (“[ ]”) so that it will be interpreted as a literal character, rather than a wildcard.

to scroll through the *Exist in Sets* lists, you must press the **Edit Set** button in the lower righthand corner of the window. Within the window, you may move to view another set by selecting its name from the **Set name** drop-down menu in the upper lefthand corner of the window.

Figures 28-30 show several examples of how these rules may be combined to create sets. Figure 28 shows how this window is used to specify a very simple set definition: all commodities that contain the string "GAS" in their names.

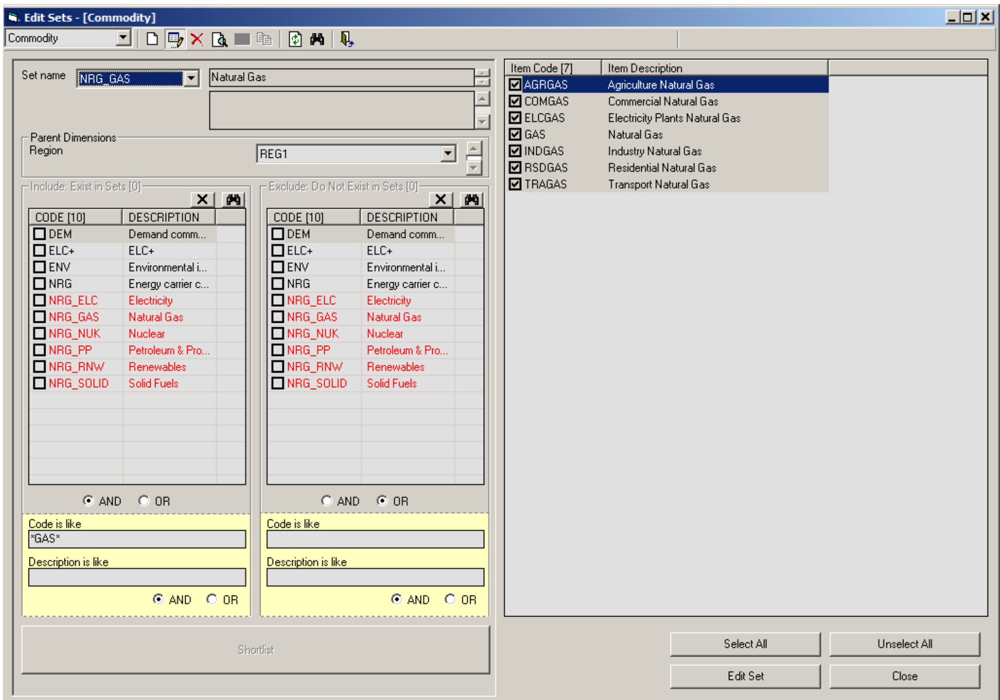


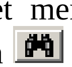
Figure 28: The *NRG_GAS* set definition

Figure 29 shows an example of a definition making use of the Exist in and Input Commodity criteria: all items in the set *ELE* that take input commodities having any of four specified strings in their names.

Figure 30 shows a combination of inclusion and exclusion rules to create a very useful type of set: all items that exist in the set *NRG* but not in any of six user-defined commodity sets. This set is looking for energy carriers that have been missed in defining the *NRG_** user-defined sets, and should be empty when all these sets have been properly defined.

3.2 Creating and editing sets

Sets may be created either by modifying an existing set from the **Edit/View Set** function described in Section 3.1 or by selecting **New Set(s)** from the **Sets** menu. Both methods use the set definition window described in the previous section to enter the rules to define the set.

Rules for *Inclusion* and *Exclusion* of items can be defined separately in the left and right halves of the left panel. The top portion of each half provides checkboxes for inclusion/exclusion based upon set membership (***Exist/Do Not Exist in Sets*** boxes). Above these boxes, the binoculars icon  opens a search window that may be used to search for sets by name and/or description. The “X” icon to the left of the binoculars clears all selections made. Multiple set choices made here are combined with an OR logic. That is, if multiple sets are selected for

Inclusion or *Exclusion*, an item must be a member of only one such set to be included or excluded.

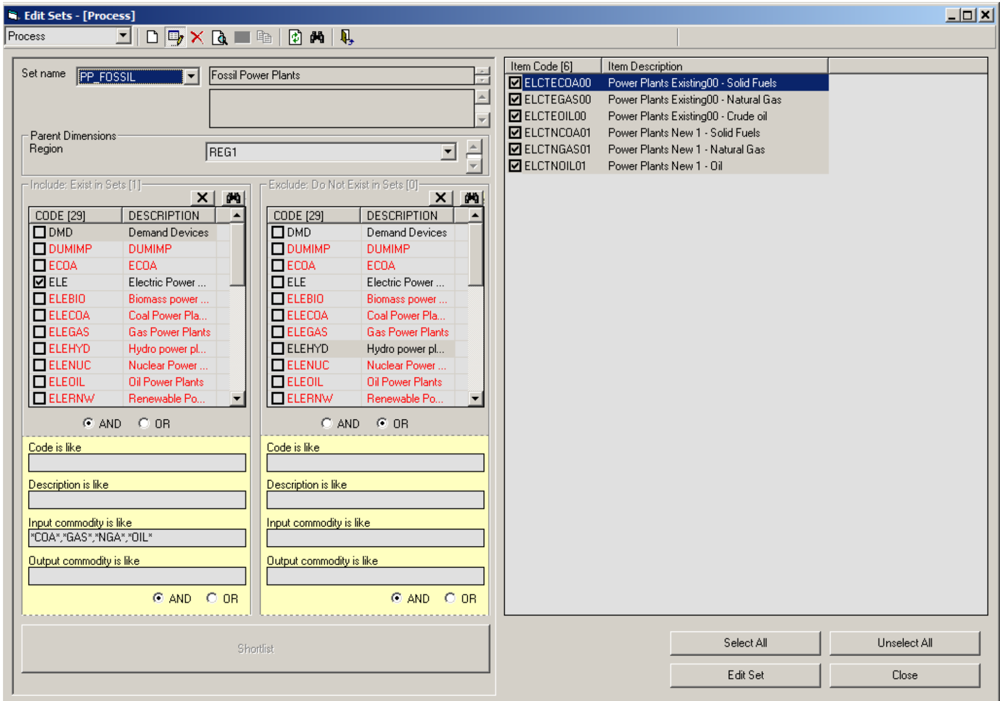


Figure 29: The *PP_FOSSIL* set definition

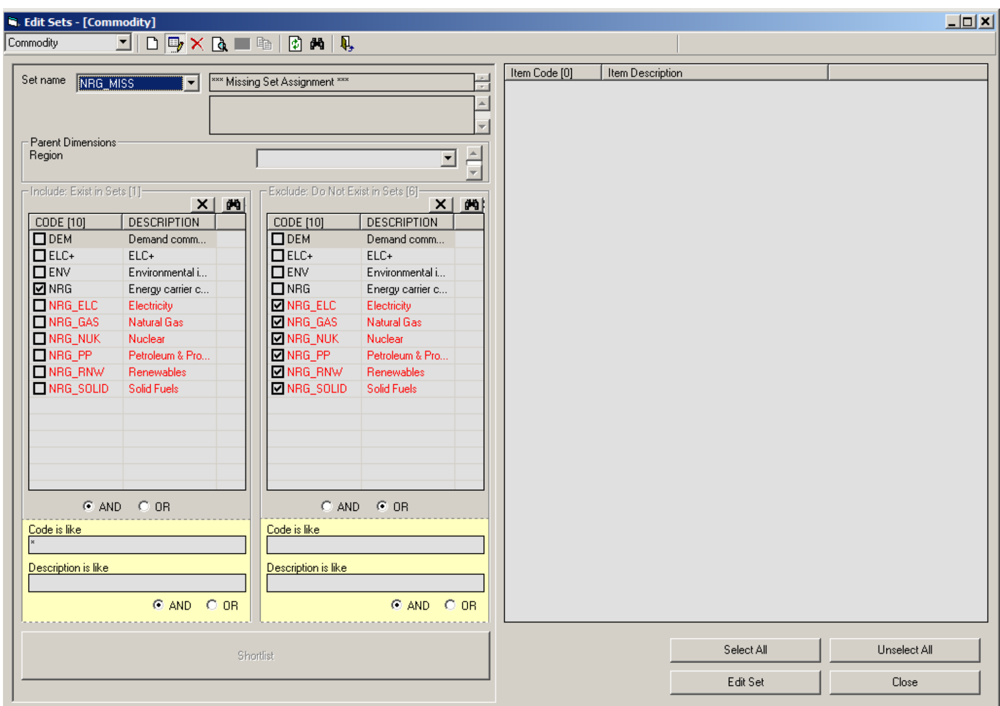


Figure 30: The *NRG_MISS* set definition

Below the set checkboxes are rows in which inclusion/exclusion criteria may be entered based upon item name, description, and, for process sets, input and output commodity. Multiple criteria may be specified on the name/description/input/output lines, separated by commas (and without spaces. These commas *within a single line* are always interpreted as logical OR. That is, to belong in the set specified in Figure 29, the process input commodity must fit only one of the four criteria entered into the ***Input commodity is like*** line on the *Include* side of the form.

Two sets of ***AND/OR*** radio buttons are found on each of the *Include/Exclude* sides. The upper one allows the user to control how the set membership and remaining conditions are combined. ***AND*** here means that *both* the set membership and the other conditions must hold, while ***OR*** means that only one is required. For example, again in Figure 29, the choice of ***AND*** in upper left ***AND/OR*** selection means that an item must be in the set *ELE* and meet one of the four stated input commodity conditions in order to belong.

The bottom set of ***AND/OR*** radio buttons on each side controls how conditions on the remaining four (two for commodity sets) rows are combined. ***AND*** means that conditions stated on different rows must all hold, while ***OR*** means that only one such condition must hold.

All of the inclusion and exclusion conditions are logically combined separately, and the final set consists of all those items that meet the inclusion conditions and do not meet the exclusion conditions. For example, the set shown in Figure 30 includes all items that exist in the set *NRG* **and** do not exist in all six specified commodity sets.

Note that sets that are based upon including/excluding members of user-defined sets may not be themselves used as *Exist/Do Not Exist in Sets* criteria, because such nesting of set memberships would cause ambiguity in set update processing.

There are often many possible ways to define a particular set of items. When choosing rules for set definition, consider which definition will be easier to maintain as your model evolves over time. In general, when choosing between rules based upon name/description or commodity input/output, it is recommended to choose the topology-based option wherever it is equally parsimonious, because it is more likely to be robust to future model development.

Once rules have been specified, press the **Shortlist** button to view the qualifying items. The right panel will list qualifying items in the region selected from the **Parent Dimensions Region** menu above the Include/Exclude panels on the lefthand side (by default, the alphabetically first region). If items may vary by region, you may wish to check the lists of qualifying items in other regions by selecting them from the menu and pressing **Shortlist** again. If your criteria do not yield any items, the following message will appear. Once it fades, you may create an empty set or modify your criteria.

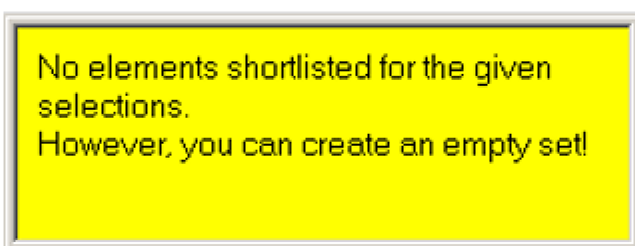


Figure 31: No elements shortlisted message

Although it is possible at this point to remove some of the shortlisted items from the set definition by manually clearing the checkboxes next to their names, this is not recommended. If needed, you may enter exclude conditions for particular items based upon their names/descriptions.

Once the set criteria have been satisfactorily defined, press the **Create Set** button (or, if you have been modifying a previous set's definition, the **Update Set** button), and a window opens to let you name the set (up to 10 characters) and provide a description. If you leave the description blank, it will be automatically set identical to the name. A comment for the set (such as its intended use) may also be entered at this time, and will appear in the set definition window under the description.

When editing a previously existing set, entering a new name will create a new set with that name, leaving the old one unchanged, whereas clicking **OK** without changing the set name will result in overwriting the old definition with the new.

You may define and/or edit multiple sets in the window before choosing **Close** when you are ready to return to the main window.

Set memberships are refreshed from their definitions whenever new results or sets are imported and when tables using them are viewed.

3.3 Managing sets

Individual sets may be deleted by right clicking on the set's name listing on the **Commodity** or **Process** tab and selecting ***Delete Set*** from the menu that appears. You will be prompted by a window similar to that shown in Figure 32, letting you know which tables the set is used in and asking you to confirm or cancel the delete.

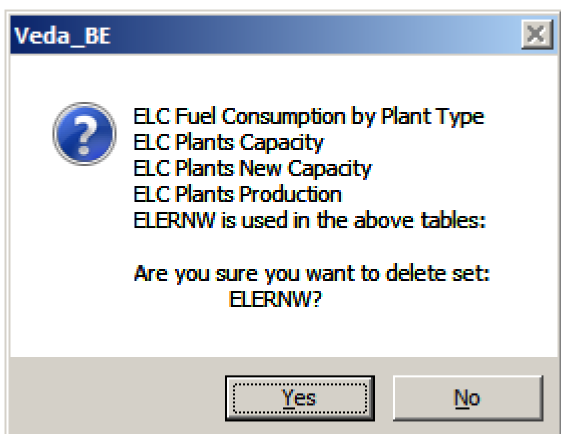


Figure 32: Confirm set deletion window

Multiple sets can be deleted by choosing **Delete Set(s)** from the **Sets** menu. This brings up a window in which sets may be searched for by name/description, and their membership and table usage viewed before choosing to delete them.

The following actions can be performed on sets by choosing the relevant item from the menu that appears upon right clicking on a set listing on the **Commodity** or **Process** tab:

- ***Copy set***: prompts you to enter a name for the new set. You may then modify the new set as needed;
- ***Rename set***;
- ***Edit description***;
- ***Copy set elements to clipboard***: puts the names of all set members in the clipboard for pasting wherever needed;
- ***Used in tables***: brings up a window showing which tables the set is used in;
- ***Comment for set***: allows you to enter a comment which will be visible when the set is viewed/edited in the set definition window.

3.4 Sharing sets

To share set definitions to another database/user/computer, there are two options. *All* sets in the database may be shared by saving the SnT file found in the database folder of the source computer to a known location on the second computer. Open the desired destination database and choose **Import Set(s)** from the **Sets** menu, and then select **Access** from the dropdown menu that appears. This will bring up a **Browse** window. Locate and select the folder containing the SnT file. (Note that the SnT file itself will not become visible in this browse, only the enclosing folder.) Next choose *Commodity* or *Process* sets from the **Select Dimension** window that appears. This will bring up the **Select** window shown in Figure 33.

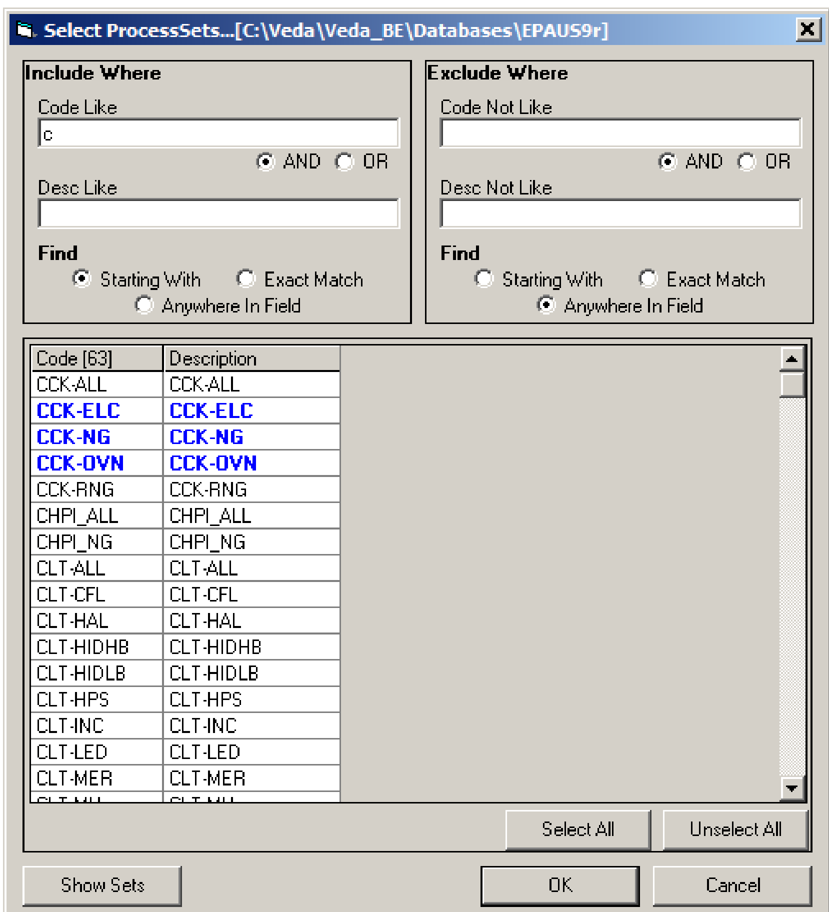


Figure 33: Select sets window for importing sets

You may enter desired criteria, if any, based on code and/or description. Select/deselect individual sets by clicking on their names (which will turn them **bold blue**), and/or Select All/Unselect All with the buttons in the bottom righthand corner. All selected (**bold blue**) sets will be imported into the destination database when you click **OK**.

If a set or sets are selected for import with names matching ones already in the database, the following window will appear. You can choose to import and overwrite, or skip these sets.

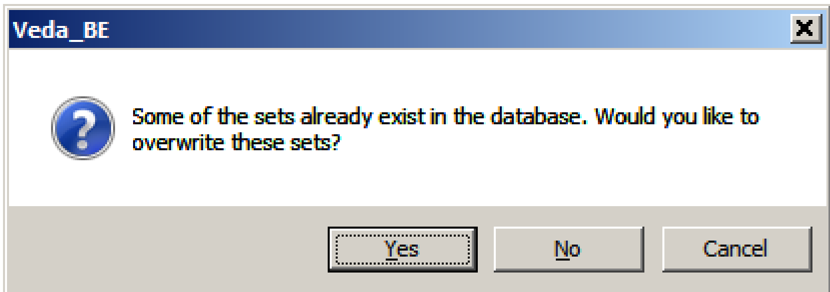


Figure 34: Import set overwrite prompt

The SnT file may also be used to share sets created in VEDA-BE with VEDA-FE for use in specifying inputs – for example, to identify qualifying renewable electricity generation processes for a portfolio standard – by pointing VEDA-FE to the VEDA-BE database folder, using the **File Locations** tab of the VEDA-FE **User Options** window.

To share only some of the sets from the source database, choose **Export Set(s)** from the **Sets** menu, and **Excel** or **Text** from the dropdown menu that appears. Next select *Commodity* or *Process* sets, and provide a name and location for the export file in the window that appears. This will bring up a search window similar that shown in Figure 33, where you may search for and select sets for export. Sets may then be imported into the destination database by choosing **Import Set(s)** and selecting **Excel** or **Text**, as relevant, from the dropdown menu, and then following the search and select procedure described above.

4 Working with tables

This section describes how to define and modify tables, how to set default table layout and formatting options, how to use the ***aggregation rules*** function to create some advanced table types, and some tools for managing and sharing tables.

4.1 Defining a new table

As introduced in Section 2.3, tables are defined in VEDA-BE by specifying which elements from each available dimension are to be included. You may think of table creation as a *filtering* process. The VEDA-BE database contains *all* the results information from the currently imported scenarios. Each table selects from and *limits* that information into a particular view, which can then be further shaped in the **View Window**.

Table definition takes place in the VEDA-BE main window. You may start the process by:

- Choosing **<New Table>** from the top of the **Table Selection** drop-down menu in the upper lefthand corner of the **Table Definition** form (see Figure 10);
- Choosing **New Table** from the **Table** menu; or
- Pressing **Crtl+N**.

All three options will give you a blank table definition form, which you may begin filling with specifications. Elements may be added to the definition form manually or via the *search engine*, as described below. On some occasions, you may find it more convenient to begin creating a new table by selecting and modifying an existing, similar table. When you request the table view, you will be given the opportunity to save the modified table with a new name, optionally retaining the old table, or to overwrite the previous table definition.

Manual filtering: Select the desired dimension tab in the right panel and locate the desired element in the appropriate list. (Having clicked on any element on a dimension tab, you may quickly jump to another element by typing the first letters of its name.) *Double-click* on the selected element (or press the *space bar*, or press the *Insert key*) to add it to the left-side table definition form. The chosen element is now displayed under the dimension heading in the window on the left, while in the right panel its entry turns **bold** to indicate that it is part of the current table definition. This operation can be reversed by highlighting the selected element in the left panel and *double clicking* on it (or pressing the *space bar*, or pressing the *Delete* key).

Filtering via the Search Engine: in the case of dimensions containing many elements, manual filtering may quickly become very awkward and time-consuming, due to the sheer number of elements. The search tool, launched by clicking the binoculars icon next to each dimension listing, allows filtering via a combination of inclusions and exclusions based on the names (short or long) of the elements in the dimension being considered. The list of elements in the bottom part of the window will be filtered as you type, showing the results of the search. Individual items may now be selected from this list by clicking on their names, turning them **bold blue** to show that they are selected, or the entire list may be selected/deselected using the **Select All/Unselect All** buttons. The **OK** button transfers the selected items to the table definition form. (Note that in doing so, they *overwrite* any items of the same dimension that are already on the table definition form.)

For example, Figure 35 shows one way in which the search engine can be used to select all the sector fuel process sets in the DemoS_VBE database. These may then be added to the table definition form (replacing any process sets already loaded onto it) by clicking **Select All** and then **OK**.

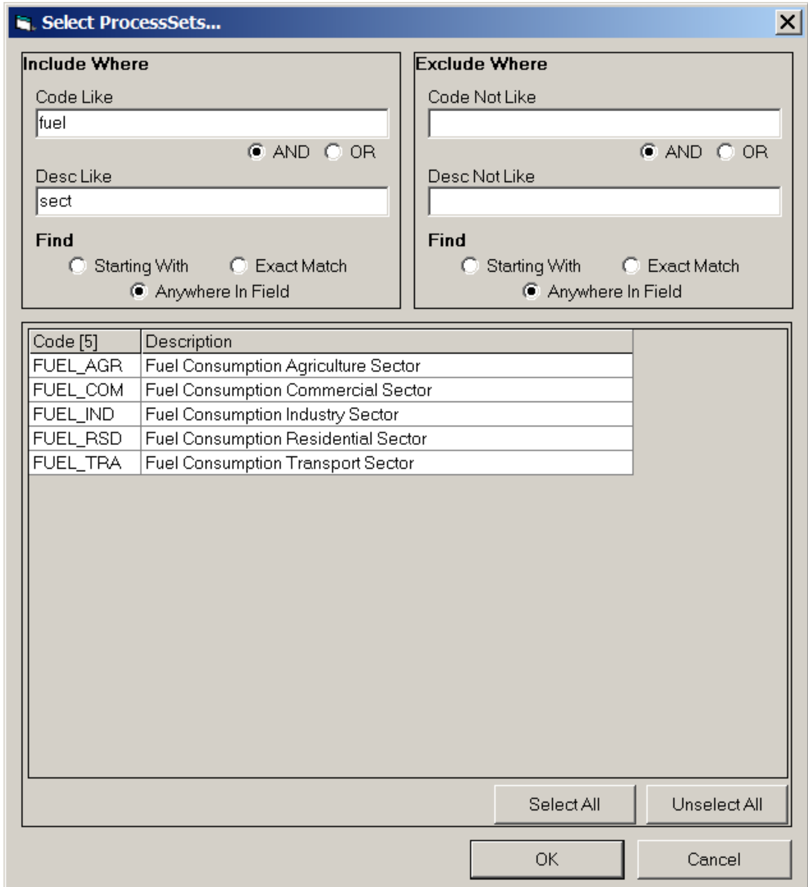


Figure 35: Using the search feature to add elements to the table definition

Dimensions for which no elements are specified for a given table are processed as follows:

- Dimensions that are not relevant to a given table are left off the table. (For example, the **_SysCost** table shown in Figure 14 lacks the process and commodity dimensions because these are not relevant dimensions of the **Reg_obj** attribute.)
- For relevant dimensions for which no specifications have been provided, *all* available elements are presented in the table, subject to any *global filter* currently in place (see Section 2.3.5).
- The exception to the previous rule is that processes and commodity sets are not added to a table when not specified. This is because any given process or commodity can belong to any number of process/commodity sets, and so no unambiguous procedure can be used to select sets to be added to such a table.

If you wish to make use of the units conversion feature for this table, select the appropriate native model units from the drop-down menu in the lower lefthand corner of the table definition form. (These units must have been previously defined, as described in Section 5.)

The **Include Null** checkbox in the lower lefthand corner of the screen allows combining attributes that have different relevant dimensions. For example, Figure 36 shows a table definition form that puts electricity generation and capacity, by fuel type, into the same table. Because ***VAR_FOut*** takes the commodity index (in this case, we want output of *ELC* only) and ***VAP_Cap*** does not, if **Include Null** was not checked, ***VAR_Cap*** would be absent from the created table. **Include Null** allows ***VAR_Cap*** to be included although its commodity is null.

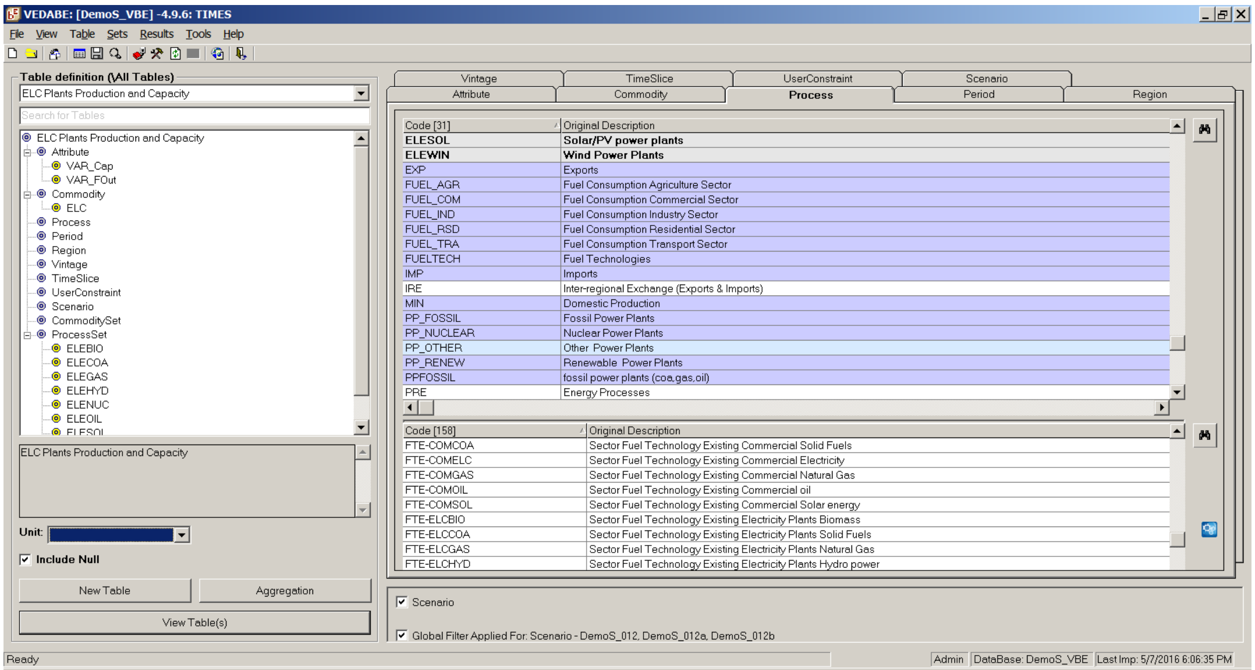


Figure 36: Using the Include Null checkbox

When you have completed specifying your table definition, click the **View Table(s)** button. You will be prompted to save the table by providing a name and description. (If you leave the description field blank, it will be set equal to the name.)

Note that if you have modified the definition of an existing table, rather than starting from a blank table, you will be presented with the table's original name here. If you click **Save** without providing a new name, the table's previous definition will be overwritten with the new one (and you will see a message telling you that the old definition has been removed). If you provide a new name, you will be presented with an option to **Retain Old Table**.

Clicking **Save** prompts VEDA-BE to construct the table by applying the filtering criteria specified for the table, according to the default layout specified using the user options described in the following section. You may then further modify the layout by pivoting, rearranging, and deselecting or collapsing items, adding or removing subtotals, and changing display types, as described in Section 2.3.

During the table construction process, a number of quality control feature checks take place. If a specified element is now missing from the database, you will seeing a message asking if you wish to continue. To remedy this situation for subsequent requests, simply remove the missing element from the table definition.

If the table includes process and/or commodity sets, and any element appears in more than one of these sets, an error message similar to that shown in Figure 37 will appear. The **Show**

Duplicates option will open a text file listing all elements that appear in multiple sets and halt the table creation process. **Continue** continues with creating the table, and **Cancel** halts the operation. The **Don't Show Duplicates For This Table** checkbox prevents this error message from being shown in the future. When this message first appears for a given table, it is recommended that you choose the **Show Duplicates** option and inspect the text file to confirm that there are no errors in the requested set specifications before proceeding.

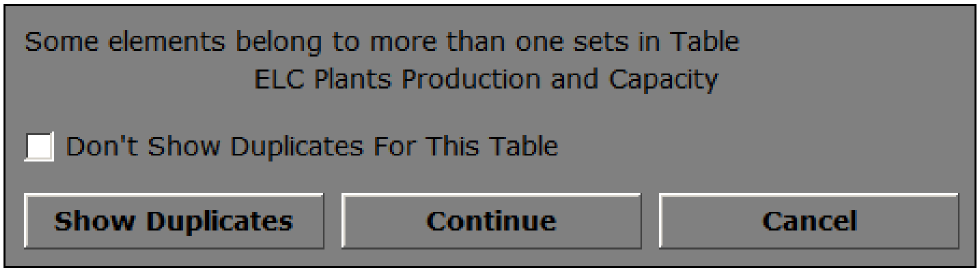


Figure 37: Duplicate elements error message

Figure 38 shows an example of the *Dupes* file listing duplicate elements found.

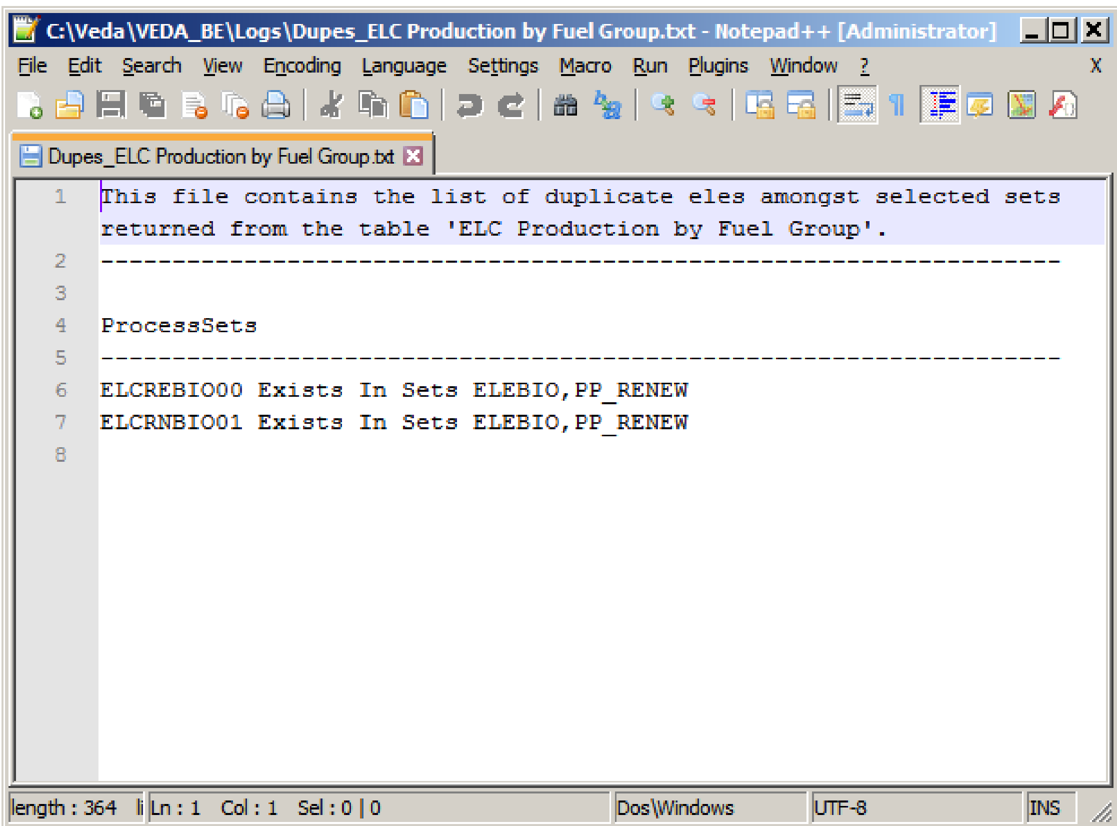


Figure 38: Example of Dupes text file listing duplicate elements

4.2 Setting table defaults

The **Options** function available from the **Tools menu** offers a number of table layout and formatting defaults that may be set, as follows:

- On the **General** tab, the default number of decimal places shown may be set;
- The **Dimensions** tab allows the default display type, location (row, column, or page), and use of subtotals to be set for each dimension;
- The **Auto Format** tab allows the default format of each portion of the table to be adjusted; and
- The **My Sort** tab allows the sort order for each dimension to be altered from its alphanumeric default.

Changes to the default table settings will be active for new tables, but will not affect existing tables.

4.3 Aggregation rules

The aggregation window, available by clicking the **Aggregation** button at the bottom of the table definition form, allows some simple operations to be requested on tables. Two of the most useful are the **enumerate** and **collapse** functions.

Enumerate directs that separate tables be produced for each element in the requested dimension. In the example shown in shown in Figure 39, separate tables will be created for each available scenario (limited by any scenarios specified in the table definition or by an applied global filter.) Up to three dimensions may be enumerated, with the order of table creation set by the suffix ***x*** on *ENUMx* when more than one dimension is enumerated.

If Export to Excel is requested when enumerated tables are open, each table will be placed on a different sheet within the same workbook. This can be very useful, for example, to create the same report table for each region in a large multi-region model.

Collapse directs that the selected dimension be summed over all available elements when the table is created. The effect is similar to that when the dimension is placed on the page of a table, rather than in a row or a column, except that when Collapse is used, the collapsed element is no longer available to be dragged back into a row or column and the details revealed. Collapse can thus greatly reduce table size, and in large models can make some tables much faster to create. (For example, power plant activity and output are tracked by timeslice, so tables containing these results may grow quite large. Collapsing by timeslice may make them far more manageable, if the timeslice detail is not required.)

The other options available within the aggregation window should be considered experimental. In general, they have been superseded by the ability to update Excel workbooks, which may contain any desired calculations and operations, with new results data (see Section 6.3).

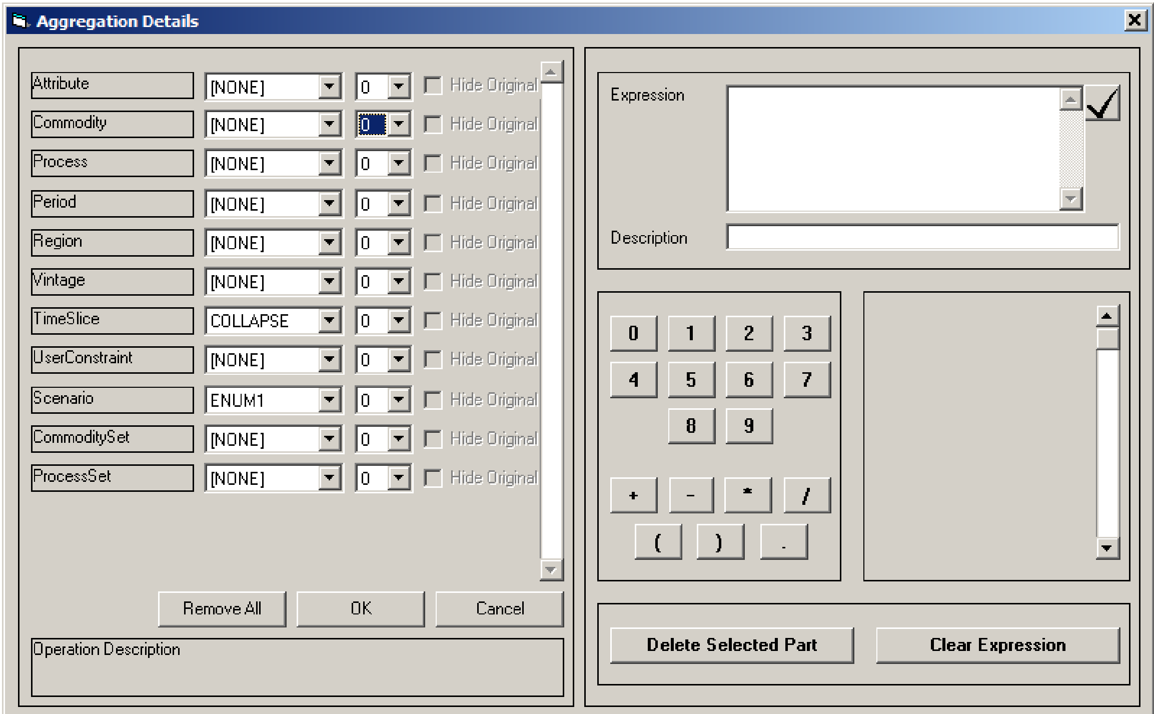


Figure 39: The aggregation window

A third type of aggregation – the difference in result values between scenarios – is easier to create from within a table view. Within the desired table, right click on the scenario column and select **Expression**, then **Diff**, and then the scenario from which you want differences calculated (see Figure 40). You will be asked if you want to hide the original table. **Yes** puts only the scenario differences in the new table, while **No** keeps the absolute scenario data in the table along with the new scenario differences. You will then be prompted to save the original table, or exit without saving any layout or definition changes you have made since opening it. Figure 41 shows the resulting table, when the original table was hidden.

You may save the scenario difference table for later use in the same way as for any other table: click the **SaveTable** icon at the top of the table window, or choose **Save** when closing the table. (Note that if you do not provide a new name and select **Retain Oil Table**, your original table will be overwritten.) The **Diff** expression can be used for other dimensions than scenario, but care must be taken to choose a difference that is meaningful.

When an aggregation rule has been set for a table, the **Aggregation** button on its table definition form will turn bold and receive asterisks (*) to show that a rule is in place, as shown:

.

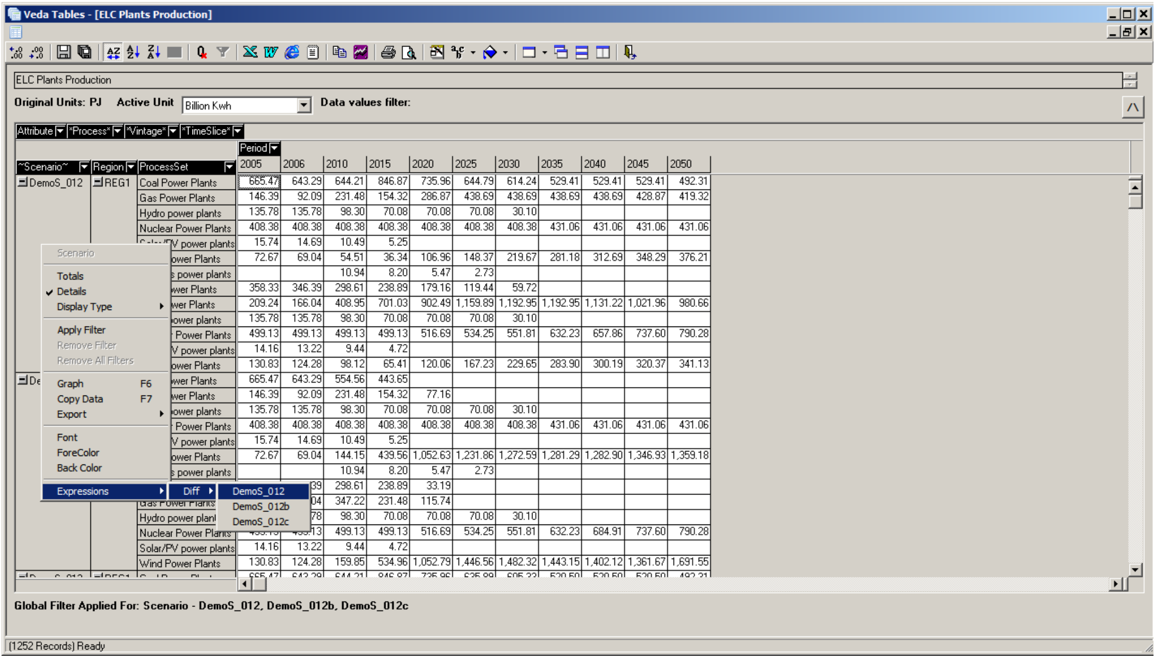


Figure 40: Requesting a difference across scenarios

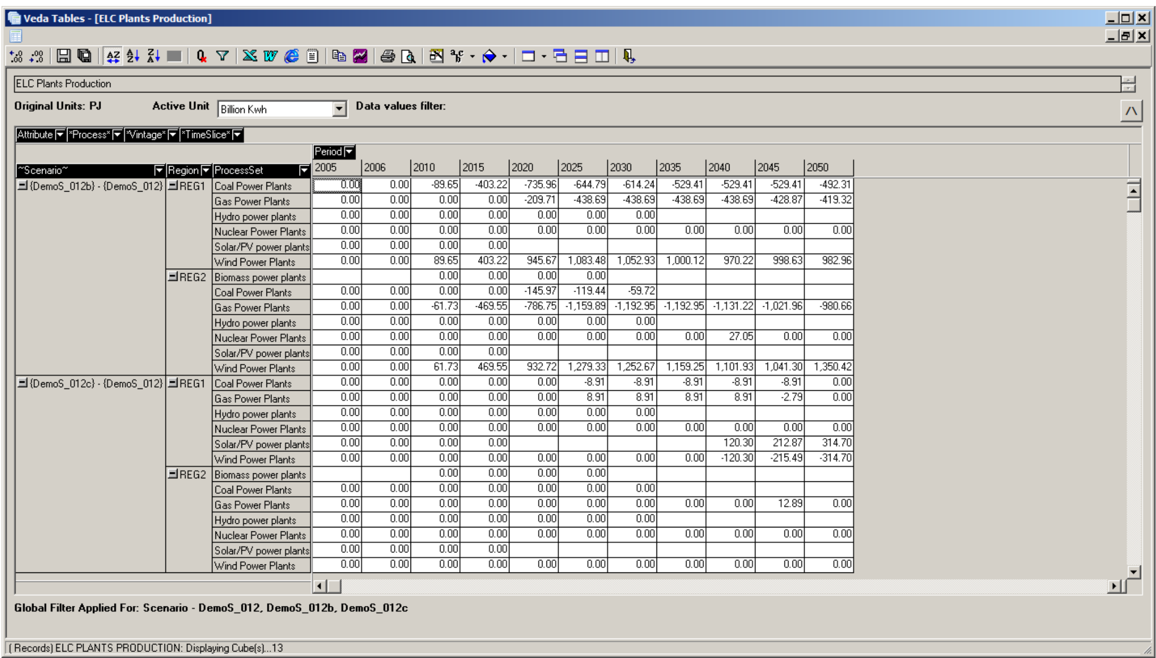


Figure 41: The resulting scenario difference table

4.4 Managing and sharing tables

From the **Table** menu of the main VEDA-BE window, one may access the following table management functions:

- Create a **New Table**;

- **Save** a table definition without launching the table view;
- Open a table **View;**
- **Delete**, **Rename**, and **Copy** tables. Note that **Delete Table** deletes only the currently selected table. To delete multiple tables at once, enter **Batch Mode** (described in Section 6.2) from the **View** menu, by selecting **Execution Mode** and then **Batch Mode**. This will provide checkboxes next to each table, which can be used to multi-select tables. Choosing **Delete Table** from the **Table** menu will then allow all the selected tables to be deleted.
- **Import** and **Export** tables using Excel or text format or the Access SnT file, in a manner precisely analogous to the import and export of sets described in Section 3.4;
- **Sort Tables** alphabetically in ascending or descending order, or by modification date;
- **Delete** saved **Cube File(s)**, which forces recreation of tables from their definitions, if they have been saved as described in Section 2.3.7. This option may be requested if you suspect a cube is not refreshing appropriately; and
- Open the **Table Master** window, described below.

The **Table Master** provides a functionality for organizing tables into folders (Figure 42). Upon opening the Table Master window, you will be shown a folder structure with the uppermost level called ***All Tables***, which always holds, as the name implies, all tables. You may add and delete folders and subfolders at will, creating a tree structure. The folders may be used in the main window to restrict the list of tables shown in the table definition window, by using the **Open Folder(s)** option from the **View** menu[7]. They may also be used to process (open for viewing) multiple or all tables within a folder from the Table Master window.

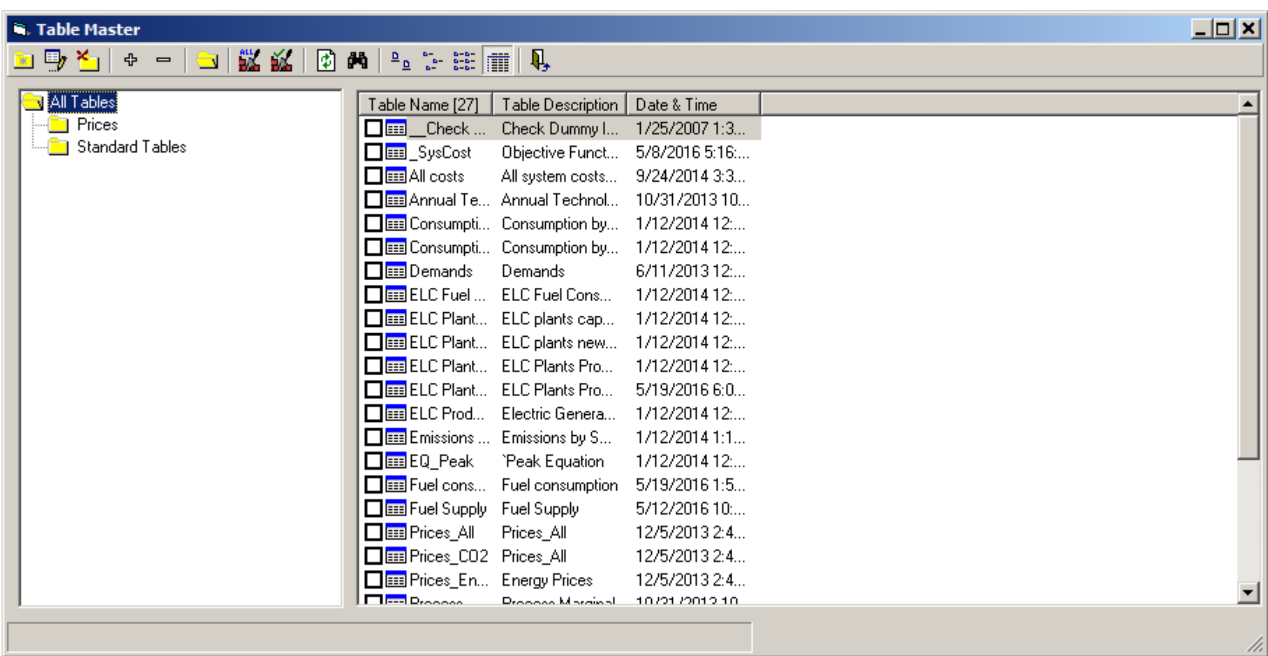


Figure 42: The Table Master window

---

[7] Note that the table list in the table definition form turns yellow when the open folder is set to anything other than ***All Tables*** to remind you that a filter has been set.

The toolbar at the top of the Table Master window offers the following tools:

- New folder
- Rename folder
- Delete folder
- Add tables to folder (opens a search window to find and select desired tables)
- Remove selected tables from folder
- Remove all tables from folder
- Process (open for view) all tables in the selected folder
- Process (open for view) selected tables in the selected folder
- Refresh tables (if you have made changes to tables while the form was open)
- Search for folder to open (useful if you have many folders)
- Display options for tables (Large Icons, Small Icons, List, and Details)
- Exit Table Master Window

For example, to group all the price-related tables into a folder, use the following steps:

- Click the **New folder** icon
- Click the **Rename** folder icon, and enter “Prices”
- Click the Plus-sign icon, which brings up a search window. Enter “pri” into the **Include Where Code Like** box, then click **Select All**, followed by **OK.**

One may now open all of the Prices tables by choosing the appropriate icon from the toolbar.

5 VEDA-BE units handling

VEDA-BE offers a simple yet powerful means for handling *units*. The user must identify the units to be used, provide each alternate unit *conversion factor*, and specify the native model units for each table. Once this is done, the resulting tables may be presented in any desired alternate units instead of the native model units. For example, we saw in Section 2.3.6 (Figure 19) that *ELC Plants Production* table is displayed in *billion kilowatt-hours* (*Billion KWh*), although the native energy carrier units for the DemoS_012 model are *petajoules* (*PJ*).

To create, edit, and delete units and to specify unit conversion factors between them, select **Units** and then **Create/Edit/Delete** from the **Tools** menu to bring up the **Units** window (Figure 43). The window offers the following tools:

- To specify a new unit, click on the **New Unit** icon . This clears the **Unit Name** and **Unit Description** textboxes on the right panel. Enter a name and, if desired, a description for the new unit, and click **Update Unit Info**.
- To edit a unit's name and/or description, select the unit from the list in the left panel. Make the desired changes to the name and/or description, and click **Update Unit Info**.

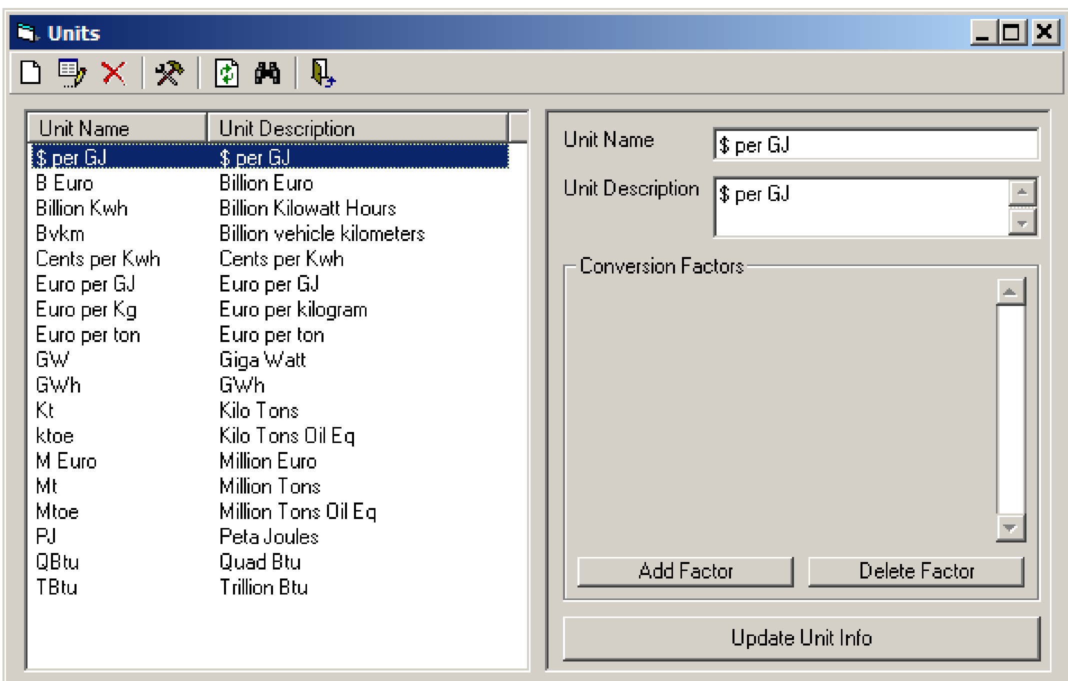


Figure 43: The Units window in the DemoS_VBE database

- To delete a unit, select it from the list in the left panel, and click on the **Delete Unit** icon . Click the **Yes** button when a message appears asking you to confirm the deletion of the unit.
- To add a new *unit conversion factor* between two units, both must first be defined as described above. Then, select one of the two units from the list in the left panel. Click on **Add Factor**. This provides a textbox into which one may type what the *currently active* unit should be multiplied by to get the resulting unit. Enter the desired conversion factor

and then choose the resulting unit from the drop-down menu immediately to the right. Finally, click **Update Unit Info**.

Conversions are automatically defined for the *reciprocal* relationship for each conversion factor created. However, additional conversions implied by this relationship with additional units are not automatically created and must be manually defined.

For example, to define a new unit for megawatts (MW) and specify that it is 1/1000 of a GW, use the following steps:

- Click on the **New Unit** icon.
- Enter “MW” in the **Unit Name** textbox and “Megawatt” in the **Unit Description** textbox. Click **Update Unit Info**.
- Click **Add Factor**. Type “.001” into the text box that appears.
- Select “GW” from the drop-down menu to the right.
- Click **Update Unit Info**. The **Units** window should now appear as shown in Figure 44.

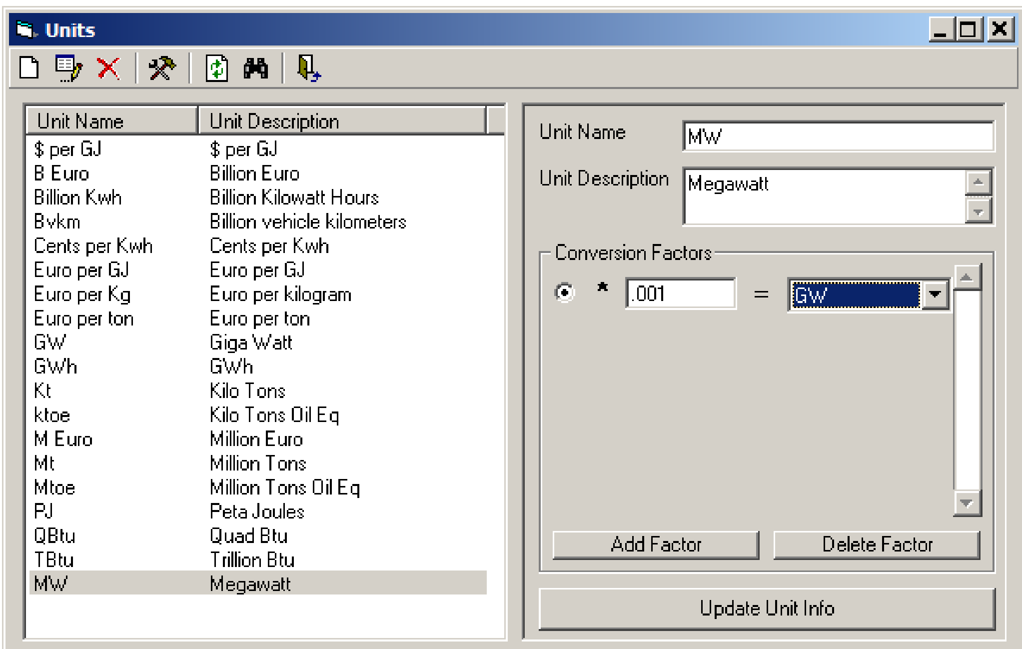


Figure 44: The Units window after adding MW

Note that selecting “GW” in the left panel should now reveal that the conversion

*** 1000 = MW**

has been added to it.

Units may be moved between VEDA-BE databases by using the units import/export function. To export units, select **Export Unit(s)** from the **Tools** menu. This will produce a prompt to name and provide a location in which to save a text (.UII) file. After you have clicked **Save**, a search form will come up allowing you to select the units to export in a very similar manner to the export of sets described in Section 3.4.

Choosing **Import Unit(s)** from the **Tools** menu will prompt you to locate and open a .UII file, and then allow you to search for and select the units to import.

The user should carefully observe the following notes on using the units handling facility:

- VEDA-BE does not know anything about native model units as they have been defined in VEDA-FE or ANSWER. The user must specify the native units in which results come to VEDA-BE from model using the table definition form, for each table where unit conversion is desired. These original units will be saved as part of the table definition. The units that appear when the table is viewed (the Active Units) may then be changed (and saved) within the view window.
- It is not necessary to provide native units as part of the table definition if no unit conversion is desired. Thus this step in the table definition may be postponed until such time that conversion is desired.
- Care needs to be taken when different attributes are mixed in a single table as they may be in different units. No native units should be provided for such a table, and the unit conversion facility should not be used, as the resulting values will be meaningless for the incompatible attribute(s).

6 Working with Excel and preparing reports

This section addresses two key capabilities for conducting a TIMES analysis: moving results data out of VEDA-BE into Microsoft Excel for further processing, and preparing analysis reports.

Results analysis often involves two distinct modes. One is exploratory in nature, where the analyst examines key results in the ExRES view and/or with a limited number of tables, drills into model details with the ExRES, exports selected pieces of data to Excel for further calculations, and designs new result tables to uncover new features of the results. There is a second mode, which may be called “production mode.” Here the table definitions and layouts have been stabilized, and it is desired to produce a set of tables and graphs without further user interference. After discussing two options for moving data into Excel, this section describes tools that may be used in production mode to produce analysis reports.

6.1 Moving results data into Excel

Two methods are available for moving data from the current table view into Excel.

First method – Copy/Paste: In any open table view, right click in the numerical area of the table, and choose **Copy Data** from the menu that appears. (Optionally, select any portion of the data by dragging through the desired cells first.) Switch to Excel and use **Edit->Paste** or **Ctrl-V** to paste the data in the desired location. The VEDA-BE table and appropriate row/column headers will be pasted along with the data. If no data was selected before choosing **Copy Data**, the entire table will be copied. This method allows you to rapidly select desired data and place it exactly where you want it.

Second method – Export: To move data from multiple tables to Excel automatically, use the **Export** method. First set user options for the handling of exported tables by choosing **Options** from the **Tools** menu of the main VEDA-BE window and going to the **Export Options** tab. Here you may set the path, file naming convention (add date/time stamp), and handling of multiple tables. Note that Excel is the recommended file type, and that **Show Exported Data** may not result in the exported files automatically opening after export, due to differences in how Excel may be configured on different machines. (On this tab you may also set behavior for **Batch** mode operations, as discussed in Section 6.2.)

To export data, in any open table view, click the **Export to Excel** icon in the toolbar, or right click in the numerical area of the table, and choose **Export** from the menu that appears. All open tables will be placed in one or multiple Excel files in the designated destination folder, according to the **Export Location** option selected.

6.2 The batch mode

The **batch mode** provides an option for processing multiple tables automatically without user intervention. This choice is available under the **View** menu, **Execution Mode** command. All we have seen up until now about VEDA-BE has been in the **Interactive** mode of operation.

If you choose the batch mode, the table definition window of the main screen is replaced by a list of existing tables, each preceded by a check box The user may then check any number of boxes, and VEDA-BE will process them as a batch when you click **View Table(s)**. Figure 45 shows an example of using the batch mode with four tables selected.

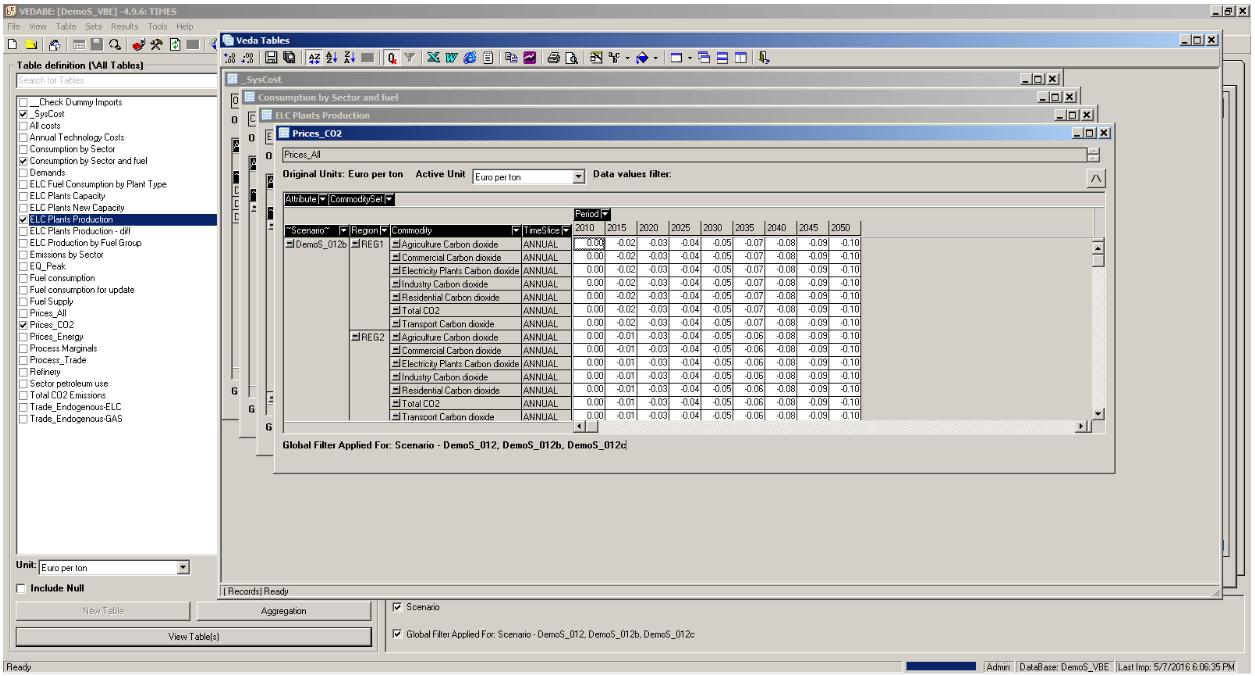


Figure 45: The batch mode

The operation(s) VEDA-BE performs on the requested tables depends on the **Batch Mode Options** set on the **Export Options** tab of the user options form, available by choosing **Options** from the **Tools** menu of the main VEDA-BE window. The user may request either or both of these options:

- **View Cube**: will open each requested table into view mode.
- **Export**: will export the requested tables according to the export options set as described in Section 6.1.

With the batch mode and its options, the user can easily reconstruct standard sets of tables as runs are refined and new runs performed.

The batch mode may also be used for database cleanup operations, to delete multiple tables at once. Choosing **Delete Table** from the **Table** menu while in batch mode will produce a prompt to confirm deletion of all tables with their checkboxes selected.

An alternative way to open multiple tables simultaneously for viewing is from the **Table Master** window (see Section 4.4). Here the user can request that all or selected tables from a Table Master folder be processed. The open tables can then be exported as described in Section 6.1.

6.3 Updating VEDA-BE results in Excel workbooks

Any Excel .xls format workbook containing tables that have been copy/pasted and/or exported using the methods described in Section 6.1 can be *updated* using the **Update Excel file** option from the **Tools** menu. When this tool is invoked, VEDA-BE will search the workbook for VEDA-BE tables corresponding to those in the open database, and then examine each data row/column, updating the data entries to the values currently in the associated tables. VEDA-BE will highlight (in yellow) all processed values, and indicate any problems encountered (in red). A comment will be placed in the table name cell, indicating any global filters used in the processing. A log file is also created (by default in the \VEDA_BE\Logs folder) that can be viewed at the end of the update request. It will list tables successfully updated along with table names found in the workbook that could not be matched (usually because of some difference in configuration) to those in the database.

This tool allows any calculations, graphs, etc. that are based upon results in a VEDA-BE table to be automatically updated with new results. The only restriction is that blank rows and columns should always be left on all four sides of tables that are to be updated by VEDA-BE, and blank rows/columns must not be inserted within the table. However, you can insert calculations such as growth rates/subtotals/shares within tables so long such row/column labels start with an asterisk (*).

Figure 46 shows a simple version of such a workbook, where a line graph has been made which will be updated each time the results are updated.

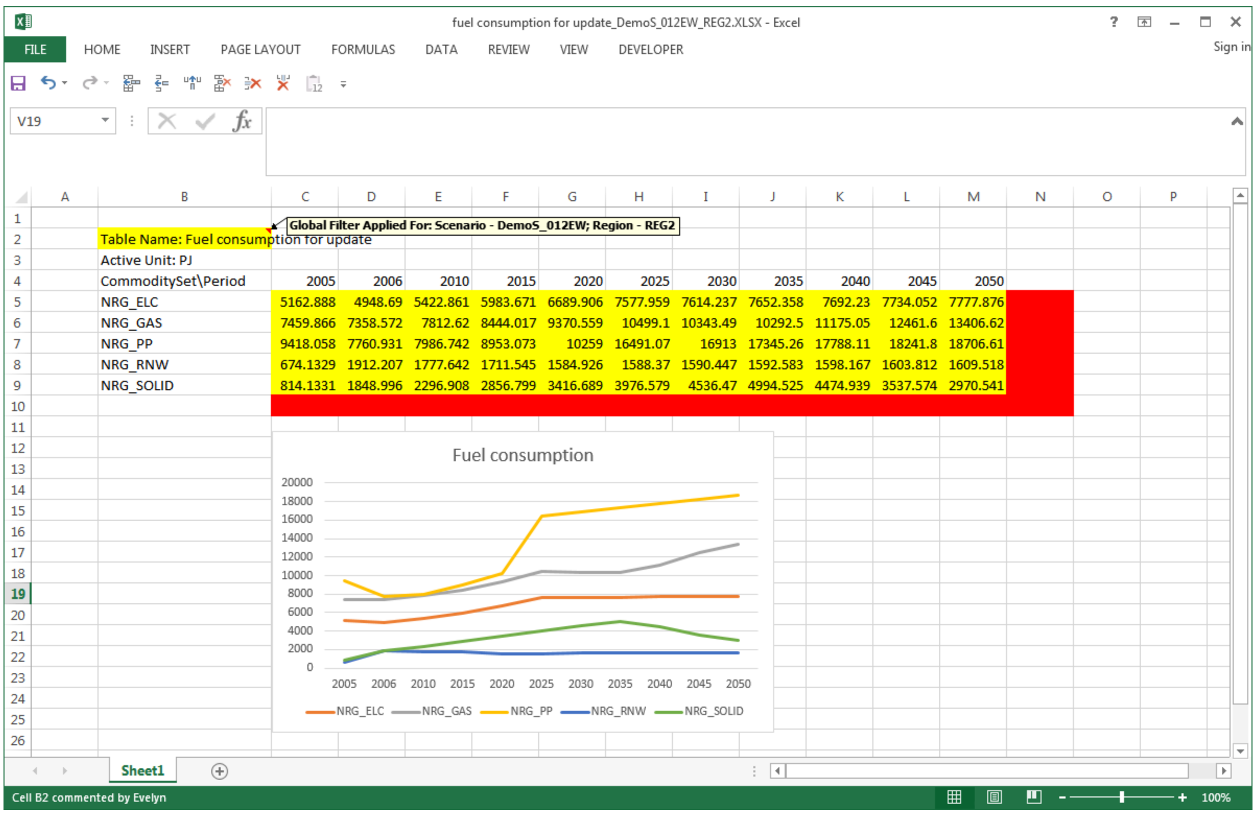


Figure 46: An Excel workbook updated with VEDA-BE results

This is a powerful tool, but one that must be used with care and attention. VEDA-BE will update those data entries for which it finds a match in the table/row/column specifications. Therefore, the user must ensure that tables currently available in VEDA-BE match exactly in format those in the target workbook. That is, table definitions and formats must not have changed, and VEDA-BE must be able to find the intended data upon processing the table, taking into account any global filters that may be currently set. VEDA-BE helps the user take care not to change the definition of tables that have been used in updates by highlighting the table name in red in the table definition form. It will also prompt you to reconsider if you attempt to save a change to the table definition.

In addition, the user should bear in mind that *only* the rows/columns listed in the target workbook table will be updated. VEDA-BE will not *add* additional rows/columns that may appear in currently available tables that were not pasted/exported into the target workbook. This can pose a problem if items (e.g., processes, fuels) that were not active in the scenarios used to create the Excel tables become active in later scenarios. If VEDA-BE does encounter extra records in a database table that are not listed in an Excel table updated, an *ExtraRecords* text file will be produced in the log folder listing these values (see Figure 47).

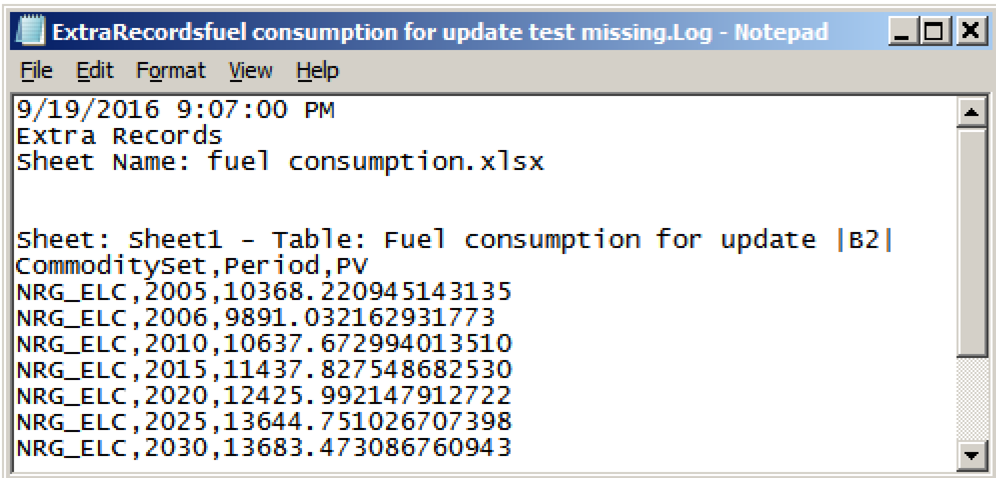


Figure 47: Sample ExtraRecords log file

In order to ensure intended results, the following best practices are suggested:

- Use sets rather than individual commodities/processes in tables wherever possible, as they are less likely to be omitted from some scenarios but appear in others.
- Prepare tables for copying/exporting to Excel by summing over the widest possible range of scenarios. (Run the full range of anticipated scenarios, place the scenario dimension on the page of all tables, and do not include scenarios in the table definition.) This will help ensure that surprise items do not show up later in the analysis.
- If elements are clearly missing, manually expand the target table to include them.
- Check the log file (and the *ExtraRecords* file, if any) in the VEDA_BE\Logs folder after updates for any possible issues.

Figure 48 shows how the *Fuel consumption* table was modified to prepare it for use in a workbook update. Process, commodity, scenario, and region have all been moved to the page. The desired scenario for any update will be selected using a global filter or through using the batch mode (described below). Individual regions may also be selected using a global filter or batch mode, or the table may be used to report the sum of results across regions. Finally, the table was saved with a different name ("*for update*") so that the original table may continue to be used within VEDA-BE.

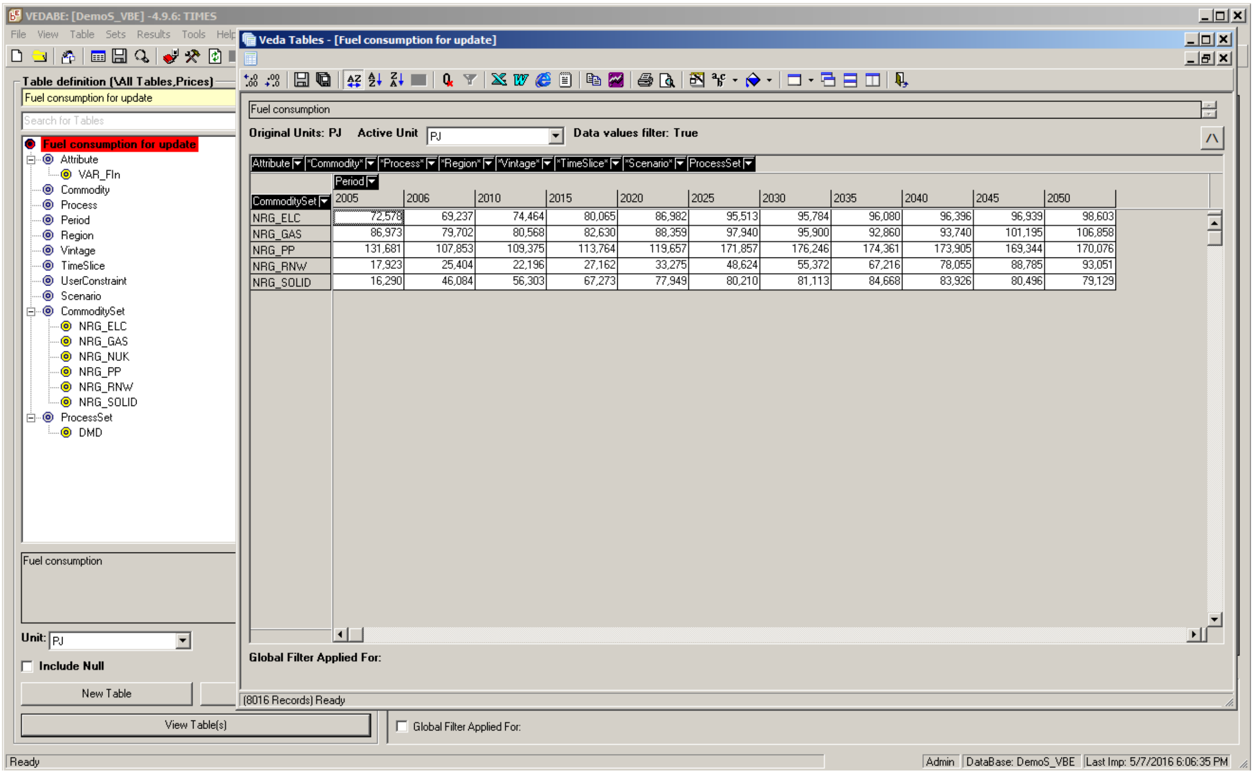


Figure 48: The Fuel consumption table as modified for use in update

The **Update Excel File** command offers three options:

- **All Tables** will prompt you to locate the desired workbook and update all properly formatted tables found in the workbook.
- **Selected Tables** will provide a checklist of all properly formatted tables found in the selected workbook and prompt you to choose the desired ones.
- **Batch Mode** will prompt you to select desired scenarios and regions, and then iterate the update process, producing separate workbooks for each scenario/region combination requested. If no regions are selected, the process will sum over all regions, subject to any global filter on regions in place at the time of update. (There is no option to sum over scenarios.)

This function can be combined with Excel functions and macros to generate quite complex reports and graphics.

7 Best practices for using VEDA-BE

This section provides a recap of advanced features and best practices discussed elsewhere in this manual that both new and experienced users are recommended to include in their repertoire.

- Use the **ExRES** for model debugging and quick result viewing without building new tables. Set a default coloring scheme for the ExRES view, to help readily distinguish it from table views (Section 2.4). The ExRES view can also be an excellent way to introduce new users to model results, as it presents a deep dive into the currently active portion of the model RES.
- When building and viewing tables, make use of **global filters.** Table definitions are meant to change only when you want to change the underlying logic of the table, not when you are just focused on a particular part of the results. Keep items that can better be applied to tables using global filters out of table definitions. Most commonly, this advice applies to scenarios and regions. Also, make use of the option to save and reuse global filter definitions (Section 2.3.5).
- When **defining sets**, consider set maintenance. How well will your definitions hold up over time? Use model topology, or rigorously adhered to naming conventions, as the basis for set definitions. And consider how you can use intersections between processes, commodities and process and commodity sets to build tables. You may need fewer sets than you think. For example, to view final energy by end use and fuel in a sector, combine process sets for devices satisfying each end use with the attribute ***Var_FIn*** and commodity sets for types of fuels (e.g., *all coal*, *all petroleum products*, etc). Separate sets by sector/end use and fuel are not needed.
- Use the **units** facility (Section 5) to readily convert tables from native model units to desired units for reporting.
- Select the user option to **save cube files** (Section 2.3.7). This speeds up the presentation of repeatedly selected analysis tables (when no information has changed in the table since the last request).
- Explore using **Batch mode** and **Excel export** and **update** operations to facilitate results reporting (Section 6).
- Join the **Veda Support Forum** (link available from the **Help** menu).

Appendix A
VEDA-BE TIMES Attributes

Table A-1 provides a list of VEDA-BE TIMES attributes produced by the **gdx2veda** GAMS utility from a TIMES run **GDX** file, according to the **times2veda.vdd** directives (see Appendix B) and the reporting options (see Part III, Section 3.10) invoked with the run. Not all attributes listed will appear in every VEDA-BE database. Many attributes will not appear if the driving input attributes were not used in the model input. See Part II, Section 3.3.1 for more details on the TIMES reporting parameters.

Table A-1 VEDA-BE attributes

VEDA-BE attribute	**Dimensions involved***	**Description**
Cap_New	p	New capacity and lumpsum investment costs. (UC tags INSTCAP and LUMPINV, respectively.)
Cost_Act	p	Annual variable activity costs of processes. Undiscounted.
Cost_Com	c	Annual commodity costs. Undiscounted.
Cost_Comx	c	Annual commodity taxes/subsidies. Undiscounted.
Cost_Dam	c	Annual undiscounted commodity related damage costs, generated by DAM_COST.
Cost_Dec	p	Annualized decommissioning costs for a process. Undiscounted.
Cost_Els	c	Annual elastic demand costs (losses) due to elastic demand changes. Undiscounted.
Cost_Fixx	p	Annual fixed taxes/subsidies associated with process installed capacity. Undiscounted.
Cost_Flo	p,c	Annual flow costs (including exogenous import/export prices). Undiscounted.
Cost_Flox	p,c	Annual undiscounted flow-related tax/subsidy costs (caused by FLO_TAX, FLO_SUB) in period (t) associated with a commodity (c) flow in/out of a process (p) with vintage period (v) as well as capacity related commodity flows.
Cost_Fom	p	Annual fixed operating and maintenance costs. Undiscounted.
Cost_Inv	p	Annualized investment costs. Undiscounted.
Cost_Invx	p	Annual undiscounted investment taxes/subsidies, spread over the economic process lifetime.
Cost_ire	p	Annual implied costs of endogenous trade, valued according to the marginal(s) of the trade equation of process p. Undiscounted
Cost_NPV	p,c	Total discounted costs by component. See Table A-2 below for components, and Part III, Section 3.10 for reporting options.
COST_Salv	p	Salvage value of investment cost, taxes and subsidies of process (p) with vintage period (v), for which the technical lifetime exceeds the end of the model horizon, value at year EOH+1.

Dual_Clic	c	Climate module results for the duals of constraint related to climate variable (c) in period (t).
EQ_Combal	c	Commodity Slack/Levels: commodity production minus consumption.
EQ_CombalM	c	Commodity shadow price
EQ_Cumflo	p,c	Level of cumulative constraint for flow of commodity (c) of process (p) between the year range (v–t).
EQ_CumfloM	p,c	Shadow price of cumulative constraint for flow of commodity (c) of process (p) between the year range (v–t). Not undiscounted.
EQ_IreM	p,c	Inter-regional trade equation marginal. The undiscounted shadow price can be interpreted as the import/export price of the traded commodity.
EQ_Peak	c	Peaking constraint slack
EQ_PeakM	c	Peaking Constraint shadow price (price premium for consumption during peak timeslice paid by the consumer in addition to COMBAL price).
ObjZ	none	Total discounted present value of system cost
PAR_CapLO	p	Capacity lower limit
PAR_CapUP	p	Capacity upper limit
Reg_ACost	r	Regional total annualized costs by period and cost category. (See Table A-2 below for categories.)
Reg_irec	r	Regional total discounted implied trade cost, derived by multiplying the shadow prices of the trade equations by the trade volumes. The sum of REG_IREC over regions is zero.
Reg_obj	r	Regional total discounted system cost
Reg_wobj	r	Regional total discounted system cost by cost type (uc_n). (See Table A-2 below for cost types.)
Time_NPV		Present value of the time in each model period (t) by region (r), with s='ANNUAL' and uc_n='COST'/'LEVCOST' depending on whether the $SET ANNCOST LEV reporting option has been used.
User_Con		Level of user constraint (or its slack). Only reported when the VAR_UC variables are used.
User_ConFXM		User constraint shadow price. Undiscounted only if the constraint is defined by region and period.
User_DynbM		Undiscounted shadow price of dynamic process-wise bound constraint, identified with name uc_n, for variable c (CAP / NCAP / ACT), in period t and timeslice s.
Val_Flo	p,c	Annual commodity flow values: Flows of process (p) multiplied by the commodity balance marginals of those commodities (c), which can be interpreted as the market values of the process inputs and outputs.
VAR_Act	p	Process activity level
VAR_ActM	p	Process activity marginal. Annual undiscounted reduced cost of process activity variable.
VAR_Cap	p	Process capacity. The vintage tags 0, -, and ¤ are used to indicate residual capacity, new capacity, and retired

		capacity, respectively.
VAR_CapM	p	Process capacity marginal. Undiscounted reduced cost of process capacity variable, when generated.
VAR_Climate		Climate module results for the levels of climate variable (c) in period (t).
VAR_Comnet	c	Commodity net quantity (consumption minus production); only generated when bound is specified by the user (COM_BNDNET).
VAR_ComnetM	c	Dual variable of bound put on the net production of a commodity.
VAR_Comprd	c	Commodity total production; only generated when bound is specified by the user (COM_BNDPRD).
VAR_ComprdM	c	Dual variable of constraint related to the bound on the production of a commodity.
VAR_CumCst		Cumulative costs by type (if constrained).
VAR_Eout		Electricity output of electricity supply processes by energy source. (Opted out by default – set RPT_OPT('FLO','5')=1 to activate; see Part III, Section 3.10).
VAR_Fin	p,c	Commodity consumption by process
VAR_Fout	p,c	Commodity production by process
VAR_Ncap	p	Technology investment
VAR_NcapM	p	Technology investment marginal. Undiscounted reduced cost of process investment variable.
VAR_NcapR	p	Technology Investment – BenCost + ObjRange (see Part II, Section 3.3.3 and Part III, Section 3.10 for more details): Cost-benefit and ranging indicators for process (p) in period (t), where uc_n is the name of the indicator: • COST - the total unit costs of VAR_NCAP (in terms of an equivalent investment cost) • CGAP - competitiveness gap (in terms of investment costs), obtained directly from the VAR_NCAP marginals (and optional ranging information) • GGAP - competitiveness gap (in terms of investment costs), obtained by checking also the VAR_ACT, VAR_FLO and VAR_CAP marginals, in case VAR_NCAP is basic at zero • RATIO - benefit / cost ratio, based on CGAP • GRATIO - benefit / cost ratio, based on GGAP • RNGLO - ranging information (LO) for VAR_NCAP (if ranging is activated; in terms of investment costs) • RNGUP - ranging information (UP) for VAR_NCAP (if ranging is activated; in terms of investment costs)

* p = *process*, c= *commodity*, r = *region*, t = *period*, s = *timeslice*, and v = *vintage*
In addition, the r,t,s,v dimensions are involved for attributes involving processes, and the r,t,s dimensions for attributes involving commodities only.

Table A-2: Acronyms used in the cost reporting parameters.

Cost parameter	**Component acronyms**	
COST_NPV	Total discounted costs by commodity/process:	
	COM	Commodity-related costs, taxes and subsidies
	ELS	Losses in elastic demands
	DAM	Damage costs
	INV	Investment costs, taxes and subsidies, excluding portions attributable to hurdle rates in excess of the general discount rate
	INV+	Investment costs, taxes and subsidies, portions attributable to hurdle rates in excess of the general discount rate
	FIX	Fixed costs, taxes and subsidies
	ACT	Activity costs
	FLO	Flows costs taxes and subsidies (including exogenous IRE prices)
	IRE	Implied trade costs minus revenues
REG_ACOST	Regional total annualized costs by period:	
	INV	Annualized investment costs
	INVX	Annualized investment taxes and subsidies
	FIX	Annual fixed costs
	FIXX	Annual fixed taxes and subsidies
	VAR	Annual variable costs
	VARX	Annual variable taxes and subsidies
	IRE	Annual implied trade costs minus revenues
	ELS	Annual losses in elastic demands
	DAM	Annual damage costs
REG_WOBJ	Regional total discounted system cost by component:	
	INV	Investment costs
	INVX	Investment taxes and subsidies
	FIX	Fixed costs
	FIXX	Fixed taxes and subsidies
	VAR	Variable costs
	VARX	Variable taxes and subsidies
	ELS	Losses in elastic demands
	DAM	Damage costs

Appendix B
GDX2VEDA and the VEDA data definition file

To get results from the GAMS solution (found in the GDX file) to VEDA-BE, the GDX2VEDA utility processes the GAMS GDX file according to the directives contained in the TIMES2VEDA.VDD (VEDA data definition) file, and produces the results files to be imported into VEDA-BE. The VDD file specifies which sets and model results are to be included and how to prepare this information for VEDA-BE. Thus the GDX2VEDA GAMS utility allows anyone familiar with a particular model to specify what information should be extracted from the GAMS model and passed to VEDA-BE. The call to the GDX2VEDA routine is embedded in the VTRUN / ANSRUN.CMD command routines.

For those working with TIMES, an appropriate initial VDD file is distributed in the source code folder (one for conventional TIMES (and MACRO/MCA), and another for Stochastics), which requests the dumping of the information summarized in Appendix A. However, with a proper understanding of the VDD file, after saving the standard VDD, a knowledgeable user may carefully augment the VDD to request additional information be moved from GAMS to VEDA-BE.

> The user is encouraged to first examine (via the **GAMSIDE**, see Section 2.3.3 of Part III) what information is found in the TIMES GDX file, including its structure (indexes), to determine what and how to introduce additional information to be dumped to the VDD. Furthermore, the process of changing the VDD should be done in consultation with someone fully familiar with the GAMS GDX file for TIMES and the basics of the GDX2VEDA utility.

In this Appendix we first simply reproduce the Command Prompt `GDX2VEDA --help` echoed help file, annotated with explanatory footnotes. The help file explains the way to call GDX2VEDA, as well as the various options available to the user in the VEDA Data Definition (VDD) file that control how and what gets dumped. The next section provides the initial TIMES2VEDA.VDD file for the core TIMES model distributed with the TIMES source code. The final section provides examples of the VD* results files.

B.1 The GDX2VEDA help file

```
>gdx2veda gdx vdd [run]

   gdx  GAMS GDX file
   vdd  VEDA Data Definition file
   run  VEDA Run identifier (optional)

The VEDA data file name and run identifier are either taken from the gdx file
name or specified with the run name. Use "token with blanks" if needed.

>gdx2veda mygdx    //  will dump the gdx symbols
>gdx2veda          //  prints this message
>gdx2veda --help   //  prints more detailed help message[8]

Add .csv to the run name to write in csv format.
```

[8] This message!

```
VDD File Summary
----------------
[DataBaseName]9
 myveda

[Dimensions]10 cube dimensions
 long_name tuple_element1 tuple_element2 ...

[DataEntries]11 data for the cube
 long_name gams_name tuple_element1 tuple_element2 ...

[DimensionText]12 for generating .vde file (only for data in [DataEntries])
 gams_set tuple_element1 tuple_element2 ...

[DimensionTextAll]13 for generating .vde file (also for data not in
[DataEntries])
 gams_set tuple_element1 tuple_element2 ...

[SubSets]14 for generating .vds file
 sub_name gams_name tuple_element1 tuple_element2 ...

[ParentDimension] defines parent-child structure
 parent_tab  child_tab1 child_tab2 ...

[ParentDimensionTextAll] .vde file definitions with parent-child structure
2d_gams_set  parent_tab child_tab
2d_gams_set  child_tab parent_tab

[ParentSubSets] .vds file definitions with parent-child structure
sub_name 2d_gams_set parent_tab child_tab
sub_name 2d_gams_set child_tab parent_tab

[Options]
TupleSeparator "string"  use a different separator symbol between tuple
elements
ShowAllSeparators        don't squeeze unnecessary separators
RelaxDimensionAll        relax strict dimensionality checks in
DimensionText(All) sections.
ValueDim n               if n=2 write PV/DV value pairs for VEDA
SetsAllowed dim1 dim2 .. write SetsAllowed specification line to VEDA .vd
file
Scenario scenarioSet     specify the scenario set; a record with expl text
goes to .vde
Format veda/csv          specify the format of the data files
Not-0 attribute ...      don't write records with zero values for these
attributes

[SpecialValues]
EPS "string"             value to be used for EPS
```

[9] Name of the application.

[10] The various tabs to appear on the VEDA-BE screen.

[11] The individual data structures (Attributes) to be managed by VEDA-BE.

[12] Optional list of individual elements of the tab_names with their descriptions. Otherwise, all instances found in the data are enumerated and their associated description passed to VEDA-BE. If one [DimensionText] is provided the auto-generated tab list is not used.

[13] Optional list of additional elements of the tab_names not found in the set/index in the model data but whose description is desired.

[14] For each tab for which set management is to be performed by VEDA-BE, the system sets and subsets need to be provided.

```
INF "string"             value to be used for +INF
MINF "string"            value to be used for -INF
NA "string"              value to be used for NA
UNDEF "string"           value to be used for UNDEF
```

```
<myveda> is usually the application name which will be displayed
         on the top of the VEDA splash screen. When a new VEDA database is
         created, a new folder with this name will appear:
         ...veda\database\mayveda_date_time.
         where date and time are the creation time stamp.
<tab_name> corresponds to the tabs of your VEDA screen
```

Lines starting with * and empty lines are ignored.
Blanks, commas and tabs are delimiters, blanks before and after delimiters are ignored. Quotes around data items are optional.
The input data is NOT case sensitive.

Example of a VEDA Data Definition file

```
* Transport model

[DataBaseName]
myveda

[Dimensions]
* tab-name  indices
Plants      i
Warehouses  j
Links       ii jj

[DataEntries]
* veda_attribute gams_name tab1 tab2 ... for gams index 1, 2, ...
"x(i,j) duals" x.m      i  Warehouses
Shipments      x.l      i  j
SupplyPrice    supply.m i
DemandPrice    demand.m j
TransportCost  c        i  j
Distance       d        ii jj
Supply         a        i
Demand         b        j
TotalCost      z.l
SupplyNodes    i        i
DeamndNodes    j        j
Rate           f

[DimensionText]
* gams_set tab
i  i

[DimensionTextAll]
j  j

[SubSets]
* sub_name gams_name tab
i1 ic  Plants
i1 id  i
```

Notes:
The long name from the [Dimensions] section can be used as a macro that expands to the tuples it defines. E.g. “Links” is identical to “ii jj”.
In the [DataEntries] section a literal tuple element can be defined as /element/.

When ValueDim=2, the [DataEntries] section can contain X.LM entries, indicating both .L and .M needs to written as a pair.

B.2 The TIMES2VEDA.VDD file for the core TIMES model

```
*
* TIMES GDX2VEDA Set Directives
*

[DataBaseName]
 TIMES

[Dimensions]
 Attribute       attr
 Commodity       c
 Process         p
 Period          t
 Region          r
 Vintage         v
 TimeSlice       s
 UserConstraint  uc_n

[ParentDimension]
 Region Commodity Process UserConstraint

[Options]
SetsAllowed Commodity Process UserConstraint
*Scenario SCENCASE
*ValueDim 2
not-0 var_fin var_fout var_act var_actm var_cap var_capm cost_flo cost_act
eq_cumflo eq_combal eq_combalm

[DataEntries]
* VEDA Attr     GAMS              - indexes -
*** Variables & Parameters
  VAR_Act       par_actl          r v t p s
  VAR_ActM      par_actm          r v t p s
  VAR_Cap       par_capl          r t p
  VAR_Cap       par_pasti         r t p v
  VAR_CapM      par_capm          r t p
  VAR_Ncap      par_ncapl         r t p
  VAR_NcapM     par_ncapm         r t p
  VAR_NcapR     par_ncapr         r t p uc_n
  VAR_FIn       f_in              r v t p c s
  VAR_FOut      f_out             r v t p c s
  VAR_FOut      agg_out           r t c s
  VAR_Comprd    par_comprdl       r t c s
  VAR_ComprdM   par_comprdm       r t c s
  VAR_Comnet    par_comnetl       r t c s
  VAR_ComnetM   par_comnetm       r t c s
  VAR_Eout      par_eout          r v t p c
  VAR_CumCst    par_cumcst        r v t uc_n c
*** Equations
  EQ_Combal     eqg_combal.l      r t c s
  EQ_CombalM    par_combalem      r t c s
  EQ_Combal     eqe_combal.l      r t c s
  EQ_CombalM    par_combalgm      r t c s
  EQ_Peak       eq_peak.l         r t c s
  EQ_PeakM      par_peakm         r t c s
  EQ_IreM       par_ipric         r t p c s uc_n
  EQ_Cumflo     par_cumflol       r p c v t
  EQ_CumfloM    par_cumflom       r p c v t
*** Parameters
  PAR_Top       par_top           r t p c uc_n
  PAR_CapLO     par_caplo         r t p
```

```
  PAR_CapUP     par_capup       r t p
  Cap_New       Cap_New         r v p t uc_n
*** Costs
  Cost_Inv      cst_invc        r v t p uc_n
  Cost_Invx     cst_invx        r v t p uc_n
  Cost_Salv     cst_salv        r v p
  Cost_Dec      cst_decc        r v t p
  Cost_Fom      cst_fixc        r v t p
  Cost_Fixx     cst_fixx        r v t p
  Cost_Act      cst_actc        r v t p uc_n
  Cost_Flo      cst_floc        r v t p c
  Cost_Flox     cst_flox        r v t p c
  Cost_Com      cst_comc        r t c
  Cost_Comx     cst_comx        r t c
  Cost_Els      cst_come        r t c
  Cost_Dam      cst_dam         r t c
  Cost_ire      cst_irec        r v t p c
  Cost_NPV      cst_pvp         uc_n r p
  Cost_NPV      cst_pvc         uc_n r c
  Time_NPV      cst_time        r t s uc_n
  Val_Flo       val_flo         r v t p c
  ObjZ          ObjZ.l
  Reg_wobj      reg_wobj        r uc_n c
  Reg_obj       reg_obj         r
  Reg_irec      reg_irec        r
  Reg_ACost     reg_acost       r t uc_n
  User_con      par_ucsl        uc_n r t s
  User_conFXM   par_ucsm        uc_n r t s
  User_conFXM   par_ucmrk       r t uc_n c s
  User_DynbM    par_ucrtp       uc_n r t p c
  User_MaxBet   par_ucmax       uc_n r p c
*** Climate and MACRO
  VAR_Climate   CM_RESULT       c t
  Dual_clic     CM_MAXC_M       c t
  VAR_Macro     TM_RESULT       c r t

[DimensionTextAll]
* Gams_set_name  Veda_Tab
adesc    attr
uc_n     uc_n
sysuc    uc_n

[ParentDimensionTextAll]
* Gams_set_name  Veda_Tab
prc_desc   r p
com_desc   r c

[ParentSubSets]
*  subset  GAMS    VEDA Tab
* processes
  DMD DMD      r p
  PRE PRE      r p
  PRW PRW      r p
  PRV PRV      r p
  REF REF      r p
  ELE ELE      r p
  CHP CHP      r p
  HPL HPL      r p
  STG RP_STG   r p
  DISTR DISTR r p
  IRE RP_IRE   r p
* commodities
  NRG  NRG     r c
  DEM  DEM     r c
```

```
ENV   ENV     r c
MAT   MAT     r c
RES   RES     r c
COMM COMM    r c
TRN   TRN     r c
AGR   AGR     r c
NE    NE      r c
IND   IND     r c
ELC+ NRGELC r c
HET+ NRGHET r c
UC_Const uc_r_each r uc_n
UC_Const uc_const  r uc_n
UC_DynBD uc_dynbd  r uc_n
```

B.3 The VEDA-BE results files

There are three files produced for VEDA-BE by the GDX2VEDA utility: the <scenarioname>.VD data dump with the attributes, <scenarioname>.VDE (set elements), and <scenarioname>.VDS (sets definition). In addition, VEDA-FE and ANSWER produce a <scenarioname>.VDT (topology) file with the RES connectivity information. These files are dumped in comma delimited format. They never require user intervention, though they may be processed by other software if desired.

Snippets of each file are shown below, after a brief description of the layout of each.

B.3.1 <scenarioname>.VD

The <scenarioname>.VD file contains the application VEDA-BE header directives (controlling the appearance of the main VEDA-BE table specification form) followed by the actual model data.

Layout, after the header: Attribute, Commodity, Process, Period, Region, Vintage, Timeslice, UserConstraint, Value;

Excerpt:

```
*  GDX2VEDAversion- 2005-10-07
*         ImportID- Scenario:DemoS_012b
*       VEDAFlavor- TIMES
*       Dimensions-
Attribute;Commodity;Process;Period;Region;Vintage;TimeSlice;UserConstraint
;PV
* ParentDimensions- Commodity: Region; Process: Region
*      SetsAllowed- Commodity;Process
*        FieldSize-
Attribute:31;Commodity:31;Process:31;Period:31;Region:31;Vintage:31;TimeSl
ice:31;UserConstraint:31;PV:20
*       NotIndexed- PV
*         ValueDim- PV
*  DefaultValueDim- PV
*   FieldSeparator- ,
*        TextDelim- "

"VAR_Act","-","AOTETOT","2005","REG1","2005","ANNUAL","-",564.8409
"VAR_Act","-","CAPEELC","2005","REG1","2005","ANNUAL","-",1148.6992095
"VAR_Act","-","COTEBIO","2005","REG1","2005","ANNUAL","-",3.939
"VAR_Act","-","COTECOA","2005","REG1","2005","ANNUAL","-",37.3712625
```

```
"VAR_Act","-","COTEELC","2005","REG1","2005","ANNUAL","-",63.8166227499999
"VAR_Act","-","COTEGAS","2005","REG1","2005","ANNUAL","-",212.309676
"VAR_Act","-","COTEOIL","2005","REG1","2005","ANNUAL","-",129.503715
"VAR_Act","-","CSHEBIO","2005","REG1","2005","ANNUAL","-",35.451
"VAR_Act","-","CSHEELC","2005","REG1","2005","ANNUAL","-",63.81662275
"VAR_Act","-","CSHEGAS","2005","REG1","2005","ANNUAL","-",495.389244
"VAR_Act","-","CSHEOIL","2005","REG1","2005","ANNUAL","-",302.175335
"VAR_Act","-","CSHESOL","2005","REG1","2005","ANNUAL","-",7.575
"VAR_Act","-","ELCNENUC00","2005","REG1","2005","S","-",746.220483540681
"VAR_Act","-","ELCNENUC00","2005","REG1","2005","W","-",723.929516459321
"VAR_Act","-","ELCREHYD00","2005","REG1","2005","SD","-",244.125
"VAR_Act","-","ELCREHYD00","2005","REG1","2005","SN","-",0.837000000000359
"VAR_Act","-","ELCREHYD00","2005","REG1","2005","WD","-",243.846
"VAR_Act","-","ELCRESOL00","2005","REG1","2005","SD","-",16.9805936073059
"VAR_Act","-","ELCRESOL00","2005","REG1","2005","SN","-",10.4147640791476
"VAR_Act","-","ELCRESOL00","2005","REG1","2005","WD","-",16.9611872146119
"VAR_Act","-","ELCRESOL00","2005","REG1","2005","WN","-",12.2907153729072
"VAR_Act","-","ELCREWIN00","2005","REG1","2005","SD","-",71.5472816780823
"VAR_Act","-","ELCREWIN00","2005","REG1","2005","SN","-",43.3049336472603
"VAR_Act","-","ELCREWIN00","2005","REG1","2005","WD","-",75.2268561643836
"VAR_Act","-","ELCREWIN00","2005","REG1","2005","WN","-",71.5472816780823
"VAR_Act","-","ELCTECOA00","2005","REG1","2005","S","-",1351.31629846897
"VAR_Act","-","ELCTECOA00","2005","REG1","2005","W","-",1044.37445353102
"VAR_Act","-","ELCTEGAS00","2005","REG1","2005","SD","-",226.711979456911
"VAR_Act","-","ELCTEGAS00","2005","REG1","2005","SN","-",300.30842173892
"VAR_Act","-","EXPCOA1","2005","REG1","2005","ANNUAL","-",745.59485
"VAR_Act","-","EXPHFO1","2005","REG1","2005","ANNUAL","-",804.770973903333
"VAR_Act","-","EXPKER1","2005","REG1","2005","ANNUAL","-",295.3885
"VAR_Act","-","EXPLPG1","2005","REG1","2005","ANNUAL","-",32.1237949472747
"VAR_Act","-","EXPNAP1","2005","REG1","2005","ANNUAL","-",400.84
"VAR_Act","-","EXPOIL1","2005","REG1","2005","ANNUAL","-",1648.4855
"VAR_Act","-","EXPOPP1","2005","REG1","2005","ANNUAL","-",453.036
"VAR_Act","-","FTE-AGRBIO","2005","REG1","2005","ANNUAL","-
",8.9265138547333
"VAR_Act","-","FTE-AGRCOA","2005","REG1","2005","ANNUAL","-
",8.46903001967851
"VAR_Act","-","FTE-AGRELC","2005","REG1","2005","SD","-",72.2276809309521
"VAR_Act","-","FTE-AGRELC","2005","REG1","2005","SN","-",66.4494664564759
"VAR_Act","-","FTE-AGRELC","2005","REG1","2005","WD","-",72.1451350098882
"VAR_Act","-","FTE-AGRELC","2005","REG1","2005","WN","-",78.418625010748
"VAR_Act","-","FTE-AGRGAS","2005","REG1","2005","ANNUAL","-
",160.377867843615
"VAR_Act","-","FTE-AGROIL","2005","REG1","2005","ANNUAL","-
",97.8265808739081
"VAR_Act","-","FTE-COMBIO","2005","REG1","2005","ANNUAL","-",48.25275
"VAR_Act","-","FTE-COMCOA","2005","REG1","2005","ANNUAL","-",37.3712625
"VAR_Act","-","FTE-COMELC","2005","REG1","2005","SD","-",309.846497299342
"VAR_Act","-","FTE-COMELC","2005","REG1","2005","SN","-",309.846497299342
"VAR_Act","-","FTE-COMELC","2005","REG1","2005","WD","-",329.999115009868
"VAR_Act","-","FTE-COMELC","2005","REG1","2005","WN","-",329.999115009868
"VAR_Act","-","FTE-COMGAS","2005","REG1","2005","ANNUAL","-
",762.742169333334
"VAR_Act","-","FTE-COMOIL","2005","REG1","2005","ANNUAL","-",507.22288375
```

B.3.2 <scenarioname>.VDE

The <scenarioname>.VDE file contains the list of individual set member elements for each index managed by VEDA-BE along with their descriptions.

Layout: Dimension Name - Region - Element name - Element Description;

Excerpt:

```
"Attribute","-","VAR_act","Process Activity"
"Attribute","-","VAR_actM","Process Activity - Marginals"
"Attribute","-","VAR_cap","Technology Capacity"
"Attribute","-","VAR_capM","Technology Capacity - Marginals"
"Attribute","-","VAR_ncap","Technology Investment - New capacity"
"Attribute","-","VAR_ncapM","Technology Investment - Marginals"
"Attribute","-","VAR_ncapR","Technology Investment - BenCost + ObjRange"
"Attribute","-","VAR_fin","Commodity Consumption by Process"
"Attribute","-","VAR_fout","Commodity Production by Process"
"Attribute","-","VAR_comprd","Commodity Total Production"
"Attribute","-","VAR_comprdM","Commodity Total Production - Marginal"
"Attribute","-","VAR_comnet","Commodity Net"
"Attribute","-","VAR_comnetM","Commodity Net - Marginal"
"Attribute","-","VAR_eout","Electricity supply by technology and energy
source"
"Attribute","-","EQ_combal","Commodity Slack/Levels"
"Attribute","-","EQ_combalM","Commodity Slack/Levels - Marginals"
"Attribute","-","EQ_peak","Peaking Constraint Slack"
"Attribute","-","EQ_peakM","Peaking Constraint Slack - Marginals"
"Attribute","-","EQ_Cumflo","Cumulative flow constraint - Levels"
"Attribute","-","EQ_CumfloM","Cumulative flow constraint - Marginals"
"Attribute","-","PAR_capLO","Capacity Lower Limit"
"Attribute","-","PAR_capUP","Capacity Upper Limit"
"Attribute","-","PAR_Top","Process topology (Opted out - SET RPT_TOP YES
to activate)"
"Attribute","-","Cap_New","Newly installed capacity and lumpsum investment
by vintage and commissioning period"
"Attribute","-","COST_inv","Annual investment costs"
"Attribute","-","COST_dec","Annual decommissioning costs"
"Attribute","-","COST_salv","Salvage values of capacities at EOH+1"
"Attribute","-","COST_late","Annual late costs"
"Attribute","-","COST_fom","Annual fixed operating and maintenance costs"
"Attribute","-","COST_act","Annual activity costs"
"Attribute","-","COST_flo","Annual flow costs (including import/export
prices)"
"Attribute","-","COST_com","Annual commodity costs"
"Attribute","-","COST_els","Annual elastic demand cost term"
"Attribute","-","COST_dam","Annual damage cost term"
"Attribute","-","COST_invx","Annual investment taxes/subsidies"
"Attribute","-","COST_fixx","Annual fixed taxes/subsidies"
"Attribute","-","COST_flox","Annual flow taxes/subsidies"
"Attribute","-","COST_comx","Annual commodity taxes/subsidies"
"Attribute","-","COST_ire","Annual implied costs of endogenous trade"
"Attribute","-","COST_NPV","Total discounted costs by process/commodity
(optional)"
"Attribute","-","Time_NPV","Discounted value of time by period"
```

```
"Attribute","-","VAL_Flo","Annual commodity flow values"
"Attribute","-","ObjZ","Total discounted system cost"
"Attribute","-","Reg_wobj","Regional total expected discounted system
cost"
"Attribute","-","Reg_obj","Regional total discounted system cost"
"Attribute","-","Reg_irec","Regional total discounted implied trade cost"
"Attribute","-","Reg_ACost","Regional total annualized costs by period"
"Attribute","-","User_Con","Level of user constraint"
"Attribute","-","User_ConFXM","Marginal cost of fixed bound user
constraint"
"Attribute","-","User_ConLOM","Marginal cost of lower bound user
constraint"
"Attribute","-","User_ConUPM","Marginal cost of upper bound user
constraint"
"Attribute","-","User_DynbM","Marginal cost of dynamic process bound
constraint"
"Attribute","-","User_Maxbet","Level of MaxBet constraint"
"Attribute","-","VAR_climate","Climate result variables"
"Attribute","-","Dual_Clic","Shadow price of climate constraint"
"Attribute","-","VAR_Macro","MACRO result variables"
"Commodity","REG2","GAS","Natural Gas"
"Commodity","REG1","GAS","Natural Gas"
"Commodity","REG2","ELC","Electricity"
"Commodity","REG1","ELC","Electricity"
"Commodity","REG2","AGRBIO","Agriculture Biomass"
"Commodity","REG1","AGRBIO","Agriculture Biomass"
"Commodity","REG2","AGRCO2","Agriculture Carbon dioxide"
"Commodity","REG1","AGRCO2","Agriculture Carbon dioxide"
"Commodity","REG2","AGRCOA","Agriculture Solid Fuels"
"Commodity","REG1","AGRCOA","Agriculture Solid Fuels"
"Commodity","REG2","AGRELC","Agriculture Electricity"
"Commodity","REG1","AGRELC","Agriculture Electricity"
"Commodity","REG2","AGRGAS","Agriculture Natural Gas"
"Commodity","REG1","AGRGAS","Agriculture Natural Gas"
"Commodity","REG2","AGROIL","Agriculture oil"
"Commodity","REG1","AGROIL","Agriculture Oil"
"Commodity","REG2","BIO","Biomass"
"Commodity","REG1","BIO","Biomass"
"Commodity","REG2","COA","Solid Fuels"
"Commodity","REG1","COA","Solid Fuels"
"Commodity","REG2","COMBIO","Commercial Biomass"
"Commodity","REG1","COMBIO","Commercial Biomass"
"Commodity","REG2","COMCO2","Commercial Carbon dioxide"
"Commodity","REG1","COMCO2","Commercial Carbon dioxide"
"Commodity","REG2","COMCOA","Commercial Solid Fuels"
"Commodity","REG1","COMCOA","Commercial Solid Fuels"
"Commodity","REG2","COMELC","Commercial Electricity"
"Commodity","REG1","COMELC","Commercial Electricity"
"Commodity","REG2","COMGAS","Commercial Natural Gas"
"Commodity","REG1","COMGAS","Commercial Natural Gas"
"Commodity","REG2","COMOIL","Commercial oil"
"Commodity","REG1","COMOIL","Commercial Oil"
"Commodity","REG2","COMSOL","Commercial Solar energy"
"Commodity","REG1","COMSOL","Commercial Solar energy"
```

B.3.3 <scenarioname>.VDS

The <scenarioname>.VDS file provides the set membership information for the dimensions where sets are allowed. Note that these are different from the user-defined sets (rule-based) that are managed in VEDA BE. But these sets can be used as a part of those rules.

Layout: Type of set (tab), region, set name, item name;

Excerpt:

```
"Commodity","REG1","ELC+","ELC"
"Commodity","REG2","ELC+","ELC"
"Commodity","REG1","ELC+","AGRELC"
"Commodity","REG2","ELC+","AGRELC"
"Commodity","REG1","ELC+","COMELC"
"Commodity","REG2","ELC+","COMELC"
"Commodity","REG1","ELC+","RSDELC"
"Commodity","REG2","ELC+","RSDELC"
"Commodity","REG1","ELC+","TRAELC"
"Commodity","REG2","ELC+","TRAELC"
"Commodity","REG1","ENV","AGRCO2"
"Commodity","REG2","ENV","AGRCO2"
"Commodity","REG1","ENV","COMCO2"
"Commodity","REG2","ENV","COMCO2"
"Commodity","REG1","ENV","ELCCO2"
"Commodity","REG2","ENV","ELCCO2"
"Commodity","REG1","ENV","INDCO2"
"Commodity","REG2","ENV","INDCO2"
"Commodity","REG1","ENV","RSDCO2"
"Commodity","REG2","ENV","RSDCO2"
"Commodity","REG1","ENV","TOTCO2"
"Commodity","REG2","ENV","TOTCO2"
"Commodity","REG1","ENV","TRACO2"
"Commodity","REG2","ENV","TRACO2"
"Commodity","REG1","DEM","DAOT"
"Commodity","REG2","DEM","DAOT"
"Commodity","REG1","DEM","DCAP"
"Commodity","REG2","DEM","DCAP"
"Commodity","REG1","DEM","DCOT"
"Commodity","REG2","DEM","DCOT"
"Commodity","REG1","DEM","DCSH"
"Commodity","REG2","DEM","DCSH"
"Commodity","REG1","DEM","DIDM1"
"Commodity","REG2","DEM","DIDM1"
"Commodity","REG1","DEM","DRAP"
"Commodity","REG2","DEM","DRAP"
"Commodity","REG1","DEM","DROT"
"Commodity","REG2","DEM","DROT"
"Commodity","REG1","DEM","DRSH"
"Commodity","REG2","DEM","DRSH"
"Commodity","REG1","DEM","DTCAR"
"Commodity","REG2","DEM","DTCAR"
"Commodity","REG1","DEM","DTPUB"
"Commodity","REG2","DEM","DTPUB"
"Commodity","REG1","NRG","GAS"
```

B.3.4 <scenarioname>.VDT

The <scenarioname>.VDT file contains all the Reference Energy System (RES) topology information.

Layout: Region, Process, Commodity, IN/OUT topology indicator. VEDA BE also enables one to look at UCs that are related to a process or commodity. <UC Name>, Process, Commodity, “UC” entries are needed for that.

Excerpt:

```
*VFEPATH=C:\Veda\VEDA_Models\DemoS_012
*ScenDesc=Demo Step 012 CO2 Tax
*ScenEDesc=Demo Step 012 CO2 Tax
"AU_NUC_MaxCAP","ELCNENUC00","-","UC"
"AU_NUC_MaxCAP","ELCNNNUC01","-","UC"
"REG1","AOTETOT","AGRBIO","IN"
"REG1","AOTETOT","AGRCO2","OUT"
"REG1","AOTETOT","AGRCOA","IN"
"REG1","AOTETOT","AGRELC","IN"
"REG1","AOTETOT","AGRGAS","IN"
"REG1","AOTETOT","AGROIL","IN"
"REG1","AOTETOT","DAOT","OUT"
"REG1","AOTETOT","DEMO","OUT"
"REG1","AOTETOT","NRGI","IN"
"REG1","CAPEELC","COMELC","IN"
"REG1","CAPEELC","DCAP","OUT"
"REG1","CAPEELC","DEMO","OUT"
"REG1","CAPEELC","NRGI","IN"
"REG1","CAPNELC1","COMELC","IN"
"REG1","CAPNELC1","DCAP","OUT"
"REG1","CAPNELC1","DEMO","OUT"
"REG1","CAPNELC1","NRGI","IN"
"REG1","COTEBIO","COMBIO","IN"
"REG1","COTEBIO","DCOT","OUT"
"REG1","COTEBIO","DEMO","OUT"
"REG1","COTEBIO","NRGI","IN"
"REG1","COTECOA","COMCO2","OUT"
"REG1","COTECOA","COMCOA","IN"
"REG1","COTECOA","DCOT","OUT"
"REG1","COTECOA","DEMO","OUT"
"REG1","COTECOA","NRGI","IN"
"REG1","COTEELC","COMELC","IN"
"REG1","COTEELC","DCOT","OUT"
"REG1","COTEELC","DEMO","OUT"
"REG1","COTEELC","NRGI","IN"
"REG1","COTEGAS","COMCO2","OUT"
"REG1","COTEGAS","COMGAS","IN"
"REG1","COTEGAS","DCOT","OUT"
"REG1","COTEGAS","DEMO","OUT"
"REG1","COTEGAS","NRGI","IN"
"REG1","COTEOIL","COMCO2","OUT"
"REG1","COTEOIL","COMOIL","IN"
"REG1","COTEOIL","DCOT","OUT"
"REG1","COTEOIL","DEMO","OUT"
```