Energy Technology Systems Analysis Programme

https://iea-etsap.org/index.php/documentation

Documentation for the TIMES Model

PART II

July 2016
(Last update: February 2021)

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 VEDA2.0 user interface for model management and results analysis. It first offers an orientation to the basic features of VEDA2.0, including software layout, data files and tables, and model management features, both for handling the input and examining the results. 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 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;
Appendix C	The Endogenous Technological Learning capability;
Appendix D	TIMES Demand Functions.

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 should usually be before the beginning of the first period but past investments may also be specified on later years.
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.

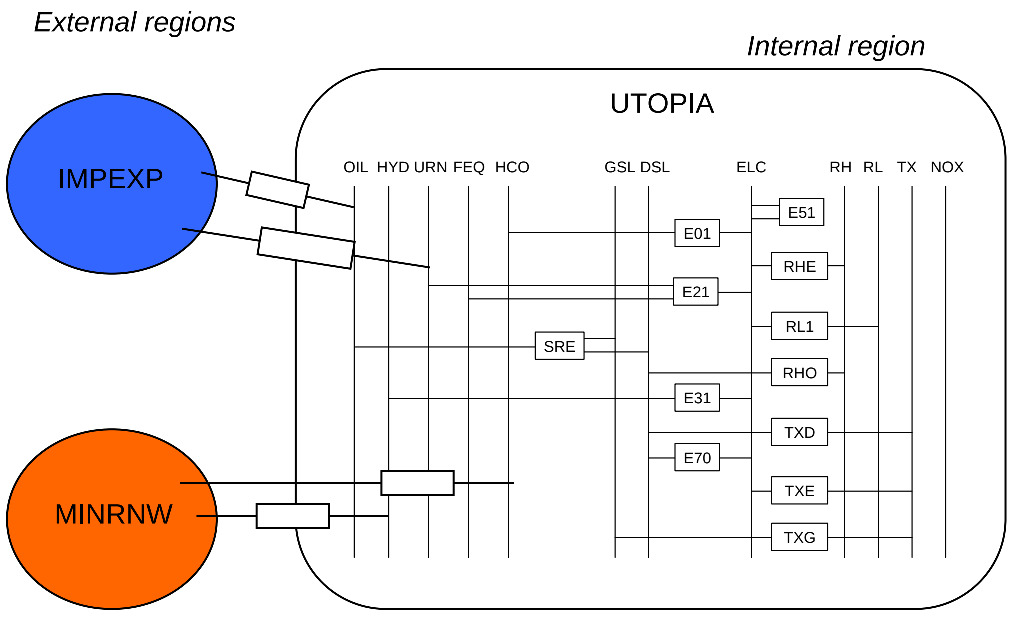


Figure 1: Example of internal and external regions in TIMES

2.2.1.1 <u>Processes</u>

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).

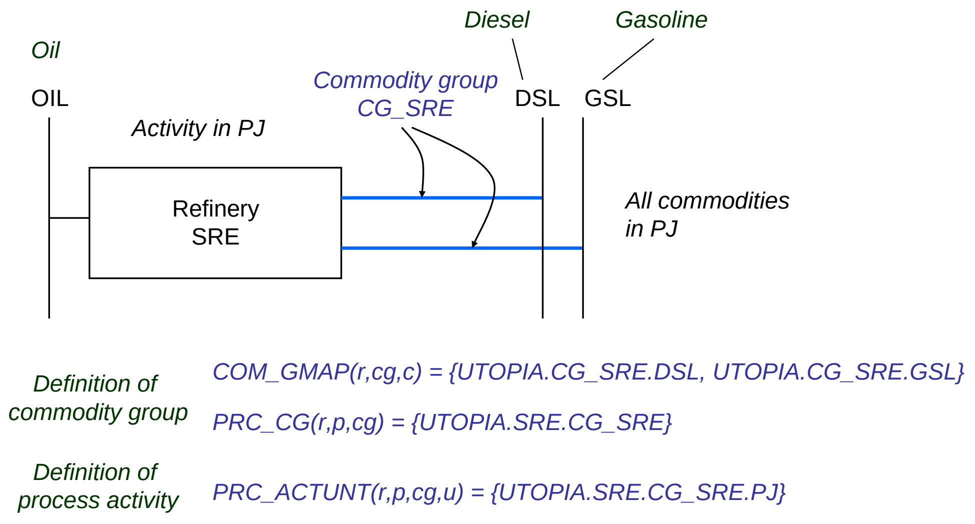

Diesel
Gasoline
Oil
OIL
Commodity group CG_SRE
DSL
GSL
Activity in PJ
Refinery SRE
All commodities in PJ
Definition of commodity group
COM_GMAP(r,cg,c) = {UTOPIA.CG_SRE.DSL, UTOPIA.CG_SRE.GSL}
PRC_CG(r,p,cg) = {UTOPIA.SRE.CG_SRE}
Definition of process activity
PRC_ACTUNT(r,p,cg,u) = {UTOPIA.SRE.CG_SRE.PJ}

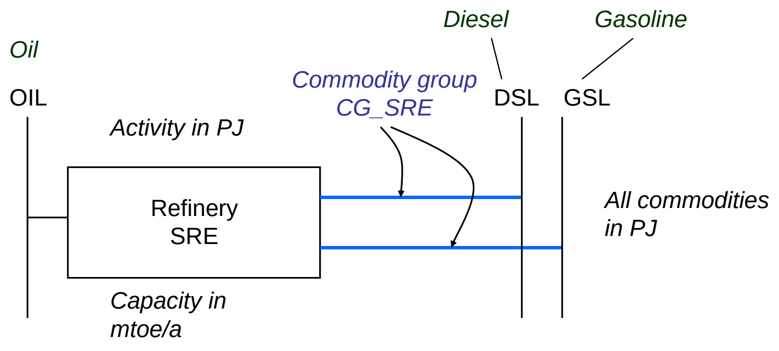
Figure 2: Example of the definition of a commodity group and the activity of a normal process.

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.

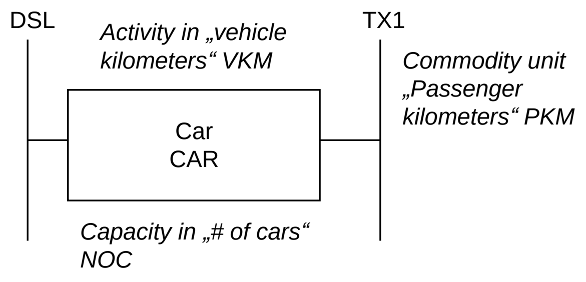


Definition of capacity unit *PRC_CAPUNT(r,p,cg,u) = {UTOPIA.SRE.CG_SRE.MTOE}*

Conversion factor from capacity to activity unit *$PRC\_CAPACT_{UTOPIA,SRE} = 41.868$*

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.



Definition of process activity *PRC_ACTUNT(r,p,cg,u) = {UTOPIA.CAR.TX1.PKM}*

Definition of capacity unit *PRC_CAPUNT(r,p,cg,u) = {UTOPIA.CAR.TX1.NOC}*

Conversion factor from capacity to activity unit *$PRC\_CAPACT_{UTOPIA, CAR} = 10000$*

Conversion factor from activity unit to commodity unit *$PRC\_ACTFLO_{UTOPIA, 2000,CAR,TX1} = 1.5$*

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.

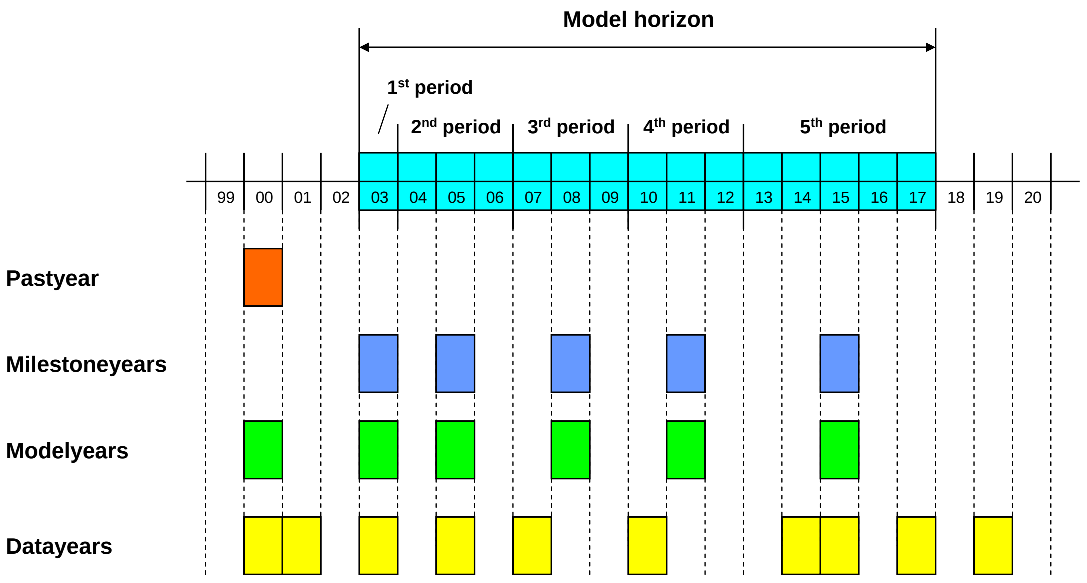


Figure 5: Definition of the time horizon and the different year types

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 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.

[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 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.

The original design of TIMES assumes that within each region, the definition of the timeslice tree applies to all model periods, such that one cannot employ different subsets of timeslices in different periods. In fact, allowing dynamically changing timeslice trees would tend to make both the model pre-processing and equation formulations substantially more complex, and therefore this design decision may be considered well justified. However, an experimental "light-weight" implementation has been made in view of supporting also dynamic timeslice trees (see Appendix E).

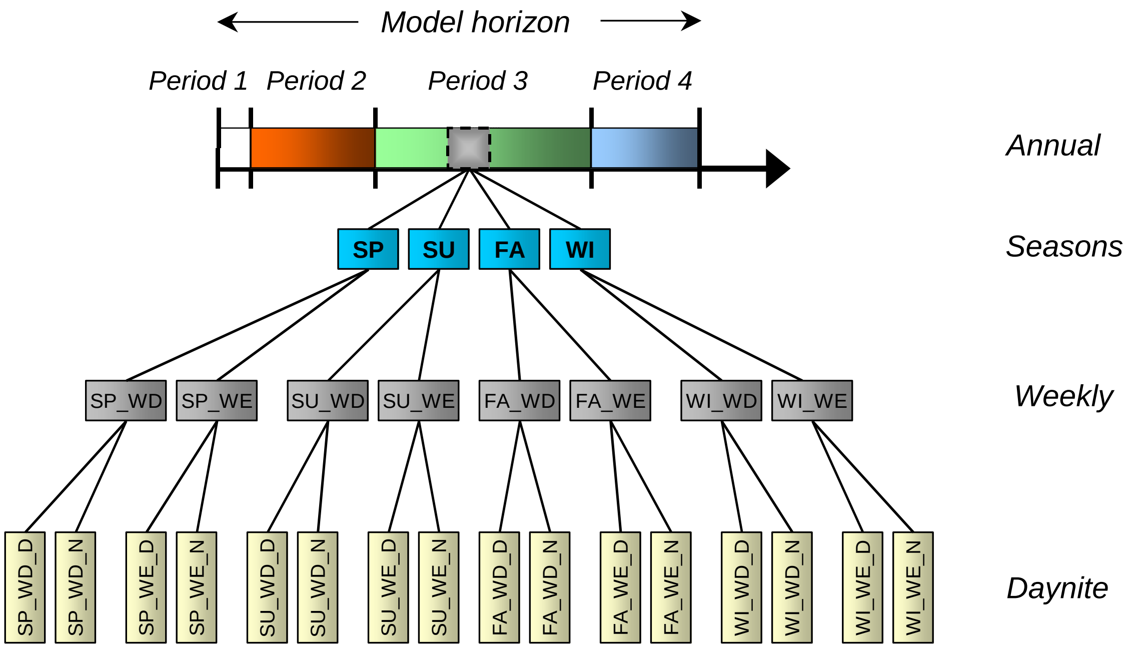


Figure 6: Example of a timeslice tree

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.

[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.

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.

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.

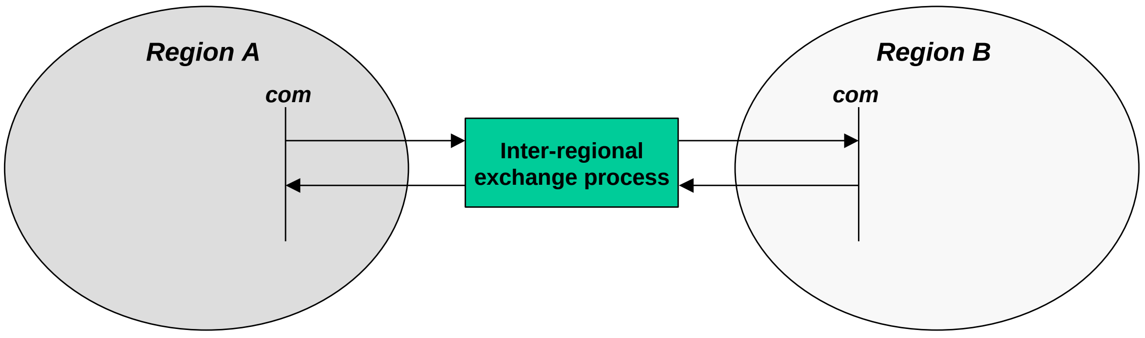


Figure 7: Bilateral trade in TIMES

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).

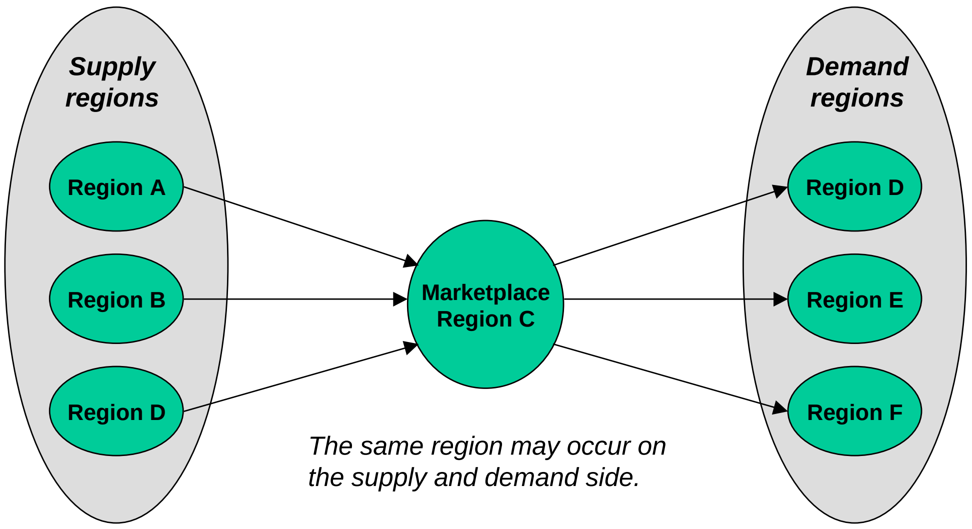


Figure 8: Example of multi-lateral trade in TIMES

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 usually 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).
ts_off (r,ts,y1,y2)		Set of quadruples {**r,ts,y1,y2**} such that the timeslice branch consisting of the timeslice **ts** and all the timeslices below it will not be taken into account in the model 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. The timeslice **ts** specified in **ts_off** must be directly below ANNUAL in the timeslice tree specified (usually at the SEASON level).
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.

Set ID/Indexes[15]	**Alias**[16]	**Description**
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**.
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.

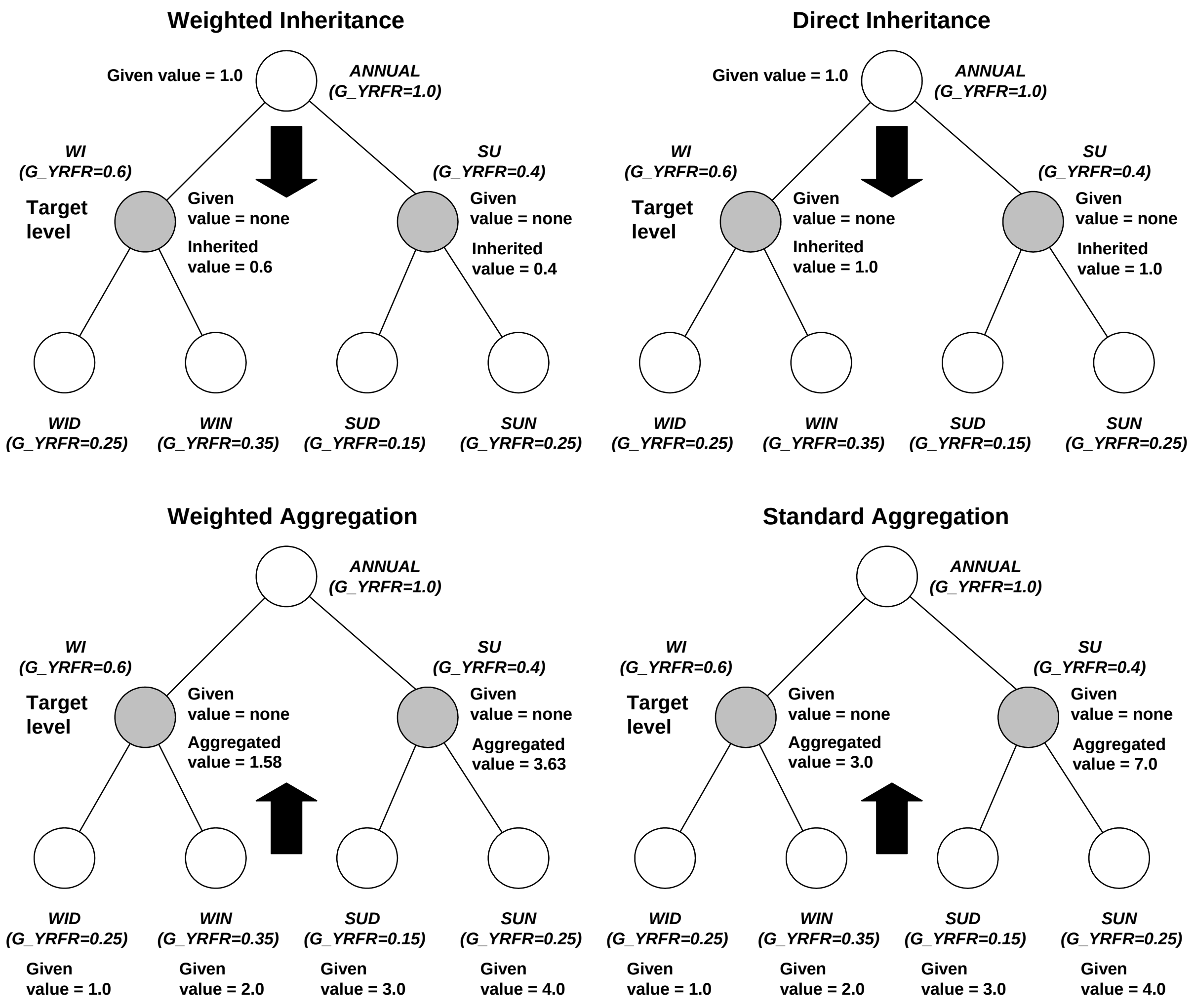


Figure 9: Inheritance and aggregation rules for timeslice specific parameters in TIMES

[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_CSTRMP (r,datayear,p,bd,cur)	ACT_UPS	Corrency unit per unit of capacity (change in load) [0,∞); default value: none Default i/e: STD	Can be used for standard processes in basic, advanced and discrete unit commitment extensions. Can also be used for load-shifting processes for defining the cost of shifting loads.	Defines ramp-up (L=UP) or ramp-down (L=LO) cost per unit of load change (in capacity units). For **load-shifting** processes defines the cost of shifting one unit of load by one hour, forward (UP) or backward (LO).	Activates generation of EQ_ACTRMPC. Generates an additional term in EQ_OBJVAR for the increase in operating cost.
ACT_CSTSD (r,datayear,p,upt,bd,cur)	ACT_CSTUP ACT_SDTIME ACT_MAXNON	Currency units per unit of started-up capacity [0,∞); Default value: none Default i/e: STD	Activates the advanced unit commitment option. In the case of the shut-down costs, only the tuple (upt, bd) = (HOT, LO) is a valid instance for this parameter. Requires the parameter ACT_MAXNON to be defined as well.	Defines start-up (bd=UP) and shutdown costs (bd=LO) per unit of started-up capacity, differentiated by start-up type (upt). The start-up type of a power plant depends on its non-operational time after shut-down,	Generates an additional term in EQ_OBJVAR for the increase in operating cost.

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]
				as defined by using ACT_MAXNON.	
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, 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.	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 EQE_ACTEFF.	Generates instances of the activity efficiency constraint (EQE_ACTEFF)
ACT_FLO (r,datayear,p,cg,s)		Flow unit per activity unit [0,∞);	Inherited/aggregated to the timeslice levels of the process activity.	Flow of commodities in cg in proportion to activity, in timeslice	Establishes a transfor-mation relationship (EQ_PTRANS) between

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	Direct inheritance. Weighted aggregation.	s.	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, which must be defined on the shadow side of the process. 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 con-sumption 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_LOSSD (r,datayear,p,upt,bd)	ACT_LOSPL ACT_MINLD ACT_SDTIME ACT_EFF	Dimensionless [0,∞); default value: none Default i/e: STD	Can only be used when the advanced unit commitment option is used for the process (therefore, defining both ACT_CSTSD and ACT_MAXNON is required) Requires also that ACT_EFF has been used for defining the process efficiency (on the shadow side of the process).	Used for modeling endogenous partial load efficiency losses during the start-up and shut-down phase. • With bd=UP defines increase in specific fuel consumption at the start up load level defined by the ratio ACT_MINLD / ACT_SDTIME(upt,'UP') for start-up type upt;	Activates generation of EQ_SUDPLL

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]
				• With bd=LO defines the increase in specific fuel consumption at the start up load level defined by the ratio ACT_MINLD / ACT_SDTIME('HOT', 'LO').	
ACT_MAXNON (r,datayear,p,upt)	ACT_CSTSD ACT_SDTIME	hours [0,∞); default value: none Default i/e: STD	Can only be used when the advanced unit commitment option is used for the process (thus defining ACT_CSTSD is required)	Max. non-operational time before transition to next stand-by condition, by start-up type, in hours • Defines the max. non-operational time before a subsequent start-up of type upt.	Activates generation of EQ_SUDUPT
ACT_MINLD (r,datayear,p)	ACT_UPS ACT_CSTUP ACT_CSTPL ACT_LOSPL	Decimal fraction [0,∞); default value: none Default i/e: STD	Can only be used for standard processes (not IRE or STG). Must be defined if ACT_CSTUP or ACT_TIME is specified.	Minimum stable operating level of a dispatchable process.	Generates instances of equations EQ_CAPLOAD and EQE_ACTUPS.
ACT_SDTIME (r,datayear,p,upt,bd)	ACT_CSTSD ACT_MAXNON	hours [0,∞); default value: none Default i/e: STD	Can only be used when ACT_CSTSD is specified for the process (advanced unit commitment option) When specifying the duration of the shut-down phase, only the tuple (upt,bd)=(HOT,LO) is valid	Defines the duration of start-up (bd=UP) and shut-down (bd=LO) phases, by start-up type, in hours.	Activates generation of EQ_SUDTIME, and used also in the equations EQ_ACTPL EQ_SDSLANT EQ_SDMINON EQ_SUDPLL
ACT_TIME (r,datayear,p,lim)	ACT_MINLD ACT_CSTUP ACT_UPS STG_SIFT	Hours [0,∞); default value: none Default i/e: STD	Can be used for standard processes when start-up costs have been modeled, using both ACT_MINLD and ACT_CSTUP at the DAYNITE/WEEKLY level.	1) Minimum online (UP) / offline (LO) hours of a process with start-up costs modeled (lim=LO/UP) 2) Maximum number	Generates instances of EQL_ACTUPC. For load-shifting storage processes, generates instances of EQ_SLSIFT.

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 lim type 'FX' is not supported for this use, and is ignored. Can also be used for load-shifting storage processes, for defining the maximum delay/advance of load shift, or the time-window for load balancing (cf. Sect. 4.3.9).	of start-up cycles within process time-slice cycles (lim=N). 3) Maximum delay or advance of load shift (lim=UP/LO/FX) or load balancing time (lim=N) for a load-shifting storage.	
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 Default i/e: N/A	See Appendix on Climate Module for details.	Various climate module constants, e.g. phi and sigma values between reservoirs.	EQ_CLITOT EQ_CLICONC EQ_CLITEMP EQ_CLIBEOH
CM_EXOFORC (year)		Forcing unit [open]; default value: none	Default values are provided. See Appendix on Climate Module for details.	Radiative forcing from exogenous sources	EQ_CLITOT

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: STD			
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
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	Aggregation of commodity NET/PRD production to the production side of the balance of another	Adds a term in EQ(l)_COMBAL and EQ(l)_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]
			is aggregated to c2.	commodity.	
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 (VAR_COMPRD) below it, according to the timeslice tree. Standard aggregation.	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).
COM_BPRICE	COM_ELAST,	Monetary unit per	The control parameter	Base price of a	Controls the inclusion 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]
(r,t,c,s,cur)	COM_STEP, COM_VOC	commodity unit [open]; default value: none Default i/e: none	$SET TIMESED 'YES' to activate elastic demands must be set.	demand commodity for the elastic demand formulation.	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 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	The years y1 and y2 may be any years of the set allyear; where y1 may also be 'BOH' for first year of	Bound on the cumulative net amount of a commodity between	Forces the net commodity variable (VAR_COMNET) to be included in the equality

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]
		possible	first period and y2 may be 'EOH' for last year of last period.	the years y1 and y2, within a region for a particular timeslice.	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,lim)	COM_BPRICE, COM_STEP, COM_VOC, COM_AGG	Dimensionless [open]; default value: none Default i/e: STD	The control parameter $SET TIMESED YES must be set to activate elastic demands. An elasticity is required for each direction the demand is permitted to move. The index lim = 'LO' corresponds to demand decrease, while lim = 'UP' denotes the direction for demand increase. A different value may be provided for each direction, thus curves may be asymmetric. Substitution elasticities can be defined with lim='N', among a group of demands aggregated by COM_AGG.	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. See also Appendix D for additional details on defining demand functions.	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]
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	Normally defined only for demand commodities (com_type = 'DEM'), but can be applied to any commodity for defining load profiles. 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) or commodity flow 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 to the bounds of elastic demand step variables (VAR_ELAST). Applied via RTFCS_FR in all equations to flows having a timeslice level coarser than target level.
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).

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_PKRSV (r,datayear,c)	com_peak, com_pkts, COM_PKFLX, FLO_PKCOI	Scalar [0,∞); default value: none Default i/e: STD		Peak reserve margin as fraction of peak demand, e.g. if COM_PKRSV = 0.2, the total installed capacity must exceed the peak load by 20%.	Applied to the total consumption of a 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' must be set to activate elastic demands. The number of steps is required for each direction the demand is permitted to move. The index bd=LO denotes the direction of demand decrease, bd=UP increase, and bd=FX is a shortcut for both. 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 linearized elastic demand formulations.	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 propor-tion 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)_FLOMRK) 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_CYCLE (tslvl)	TS_CYCLE	Number of cycles [1,∞); Default values: • 1 for ANNUAL • 1 for SEASON • 52 for WEEKLY • 365 for DAYNITE	Not recommended to be changed; use TS_CYCLE instead, whenever the timeslice cycles are different from the default, because changing G_CYCLE would change the meaning of storage availability factors.	Defines the total number of cycles on level tslvl, in a year. Provides default values for TS_CYCLE (see entry for that).	Affects interpretation of availability factors for the storage level, whenever capacity represents the maximum nominal output level (EQ(l)_CAPACT, EQL_CAPFLO).
G_DRATE (r,allyear,cur)	OBJ_DISC, OBJ_DCEOH, NCAP_DRATE, COR_SALVI,	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

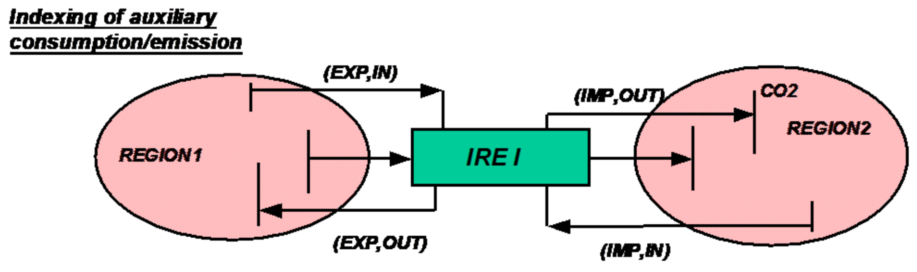
Input parameter (Indexes)[23]	**Related sets / parameters**[24]	**Units / Ranges & Default values & Default inter-/extrapolation**[25]	**Instances**[26] (Required / Omit / Special conditions)	**Description**	**Affected equations or variables**[27]
	COR_SALVD, COEF_PVT VDA_DISC				discounting multiplier (OBJ_DISC), which is applied in each components of the objective function (EQ_OBJVAR, EQ_OBJINV, EQ_OBJFIX, EQ_OBJSALV, 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	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	Prevents the investment costs associated with investment lead-times from energy the investment component of the objective function (EQ_OBJINV).

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]
			formulation is used.	costs.	
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 (datayear)	PRC_NOFF PRC_AOFF PRC_FOFF COM_OFF	Scalar [0,1] Default value: 0 Default i/e: 5	Setting G_OFFTHD=1 will make the *_OFF attributes effective only for periods fully included in the OFF range specified.	Threshold for considering an *_OFF attribute disabling a process/commodity variable in period.	Affects availability of 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	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

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 commodity (c) timeslice resolution, the bound is applied to the sum of the imports/exports according to the timeslice tree. Standard aggregation.		RHS.
IRE_CCVT (r1,c1,r2,c2))	IRE_TSCVT, top_ire	Scalar (0,∞) Default value: 1 if commodity names are the same in both regions I/e: N/A	Required for mapping commodities involved in inter-regional exchanges between two regions whenever commodities traded are in different units in the regions.	Conversion factor between commodity 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.	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_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.	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).

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]
			Weighted aggregation.		
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 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.	Auxiliary consumption (io = IN, owing to the commodity entering the process) or production/ emission (io = OUT, owing to the commodity leaving the process) of commodity c2 due to the IMPort / EXPort (index ie) of the commodity c1 in region r[30]	The multiplier is applied to the flow variable (VAR_IRE) associated with an inter-reginal exchange in the commodity balance constraint (EQ(l)_COMBAL). 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)

[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]
					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. Direct inheritance. Weighted aggregation.	IMPort/EXPort price (index ie) for to/from an internal region of a commodity (c) originating from/heading to an external region all_r.	The price of the exchange commodity is applied to the trade flow variable (VAR_IRE) in the variable costs 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.	Bound on the total IMPort (EXPort) (index ie) of commodity c in region all_r with all sources	The trade limit equation EQ(l)_XBND generated either sums lower flow variables (VAR_IRE) or splits (according to the

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]
			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).	(destinations).	timeslice tree) coarser variables.
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,	Decimal fraction [0,1];	Provided when ‘ANNUAL’ level process operation is	Annual availability factor relating the	The corresponding capacity-activity

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]
	NCAP_AFM, NCAP_AFX, COEF_AF	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. technical level of activity per unit of capacity).	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-dependent annual availability NCAP_AFS with s='ANNUAL' can be used.	annual activity of a process to the installed capacity.	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.
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		Decimal fraction	NCAP_AF, NCAP_AFA and	Availability factor	The corresponding

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,s,bd)		[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 to represent the maximum technical level of activity per unit of capacity).	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. Can be used also on the process timeslices, and will then override the levelized NCAP_AF availability factors.	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 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.	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, 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.	Bound on the permitted level on investment in new capacity	Imposes an indirect limit on the capacity transfer equation (EQ_CPT) by means 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]
			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.		a direct bound on the new investments capacity variable (VAR_NCAP).
NCAP_BPME (r,datayear,p)	NCAP_CDME	Decimal fraction [0,∞); default value: none Default i/e: STD	The parameter is only taken into account when the process is of type CHP, and NCAP_CDME has been also defined.	Back pressure mode efficiency (or total efficiency in full CHP mode).	Process transformation equation, either 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 process-es 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	FLO_SHAR	Decimal fraction	The parameter is only	Heat-to-power ratio	Activates the

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,lim)		[0,∞); default value: 1 (only when process type is CHP, for lim='UP') Default i/e: STD	taken into account when the process is defined to be of type CHP. The defaults can be disabled by defining any i/e value with lim='N', which will eliminate the output share equations.	of a CHP technology (fixed / minimum / maximum ratio). If no ratio equations should be generated, one can define any I/E value with lim='N'.	generation of output share equations, 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 during the years [K+CLAG, K+NCAP_TLIFE–1].	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 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	Provided when the consumption or production of a commodity is tied to the level of the installed	Emission (or land-use) of commodity c associated with the capacity of a process	Applied to the capacity variable (VAR_CAP) in the commodity balance (EQ_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]
		Default i/e: STD	capacity.	for each year said capacity exists.	
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_CPX (r,datayear,prc)	COEF_CPT	Integer number Default value: 0 (no shape curve applied) Default extrapolation: MIG	Provided when shaping based upon age is desired. The SHAPE index given by NCAP_CPX is applied to the internal capacity transfer parameter (COEF_CPT). Note: Shape index 1 is reserved for constant 1.	Defines a shape index for shaping the capacity transfer coefficients by the age of each process vintage. As a result, the capacity will have a survival rate as a function of age.	Impacts all calculations that are dependent upon the availability of capacity (VAR_NCAP), most directly the capacity transfer (EQ_CPT), and capacity availability equations (EQ(l)_CAPACT).
NCAP_DCOST (r,datayear,p,cur)	NCAP_DLAG, COR_SALVD, OBJ_DCOST, CST_DECC, CST_PVP	Monetary unit per capacity unit [0,∞); default value: none Default i/e: STD	Provided when there are decommissioning costs associated with a process. Decommissioning of a process and the payment of decommissioning costs may be delayed by a lag time (NCAP_DLAG).	Cost of dismantling a facility after the end of its lifetime.	Applied to the current capacity subject to decommissioning (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	rp_dscncap	Capacity unit	Used for lumpy	Size of capacity units	Applied to the lumpy

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,unit)		[0,∞); default value: none Default i/e: MIG	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.	that can be added.	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 (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).

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]
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 components (EQ_OBJINV, EQ_OBJSALV, EQ_OBJFIX).
NCAP_FDR (r,datayear,prc)	NCAP_COST	Decimal fraction (0,∞); default value:none Default i/e: STD	Provided when the effect of functional depreciation is considered significant to justify accelerated decrease in salvage value.	Defines an annual rate of additional depreciation in the salvage value.	Affects the salvage value coefficients in EQ_OBJSALV
NCAP_FOM (r,datayear,p,cur)	OBJ_FOM, CST_FIXC, CST_PVP	Monetary unit per capacity unit [0,∞); default value: none	Provided when there is a fixed cost associated with the installed capacity.	Fixed operating and maintenance cost per unit of capacity according to the	Fixed operating and maintenance costs associated with total installed capacity

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: STD		installation year.	(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 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 subsidy (NCAP_FSUB).	*{See Related Parameters}*
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}*

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]
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 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	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]
			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 costs accounting.	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, 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

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]
					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, although the dismantling may take place only for a few years of the 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.
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	NCAP_PASTY,	capacity unit	Past investment can also	Investment in new	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]
(r,pastyear,p)	OBJ_PASTI, PAR_PASTI, PRC_RESID	[0,∞); default value: none No i/e	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.	capacity made before the beginning of the model horizon (in the year specified by pastyear).	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,∞); default value: none Default i/e: MIG	Upper bound for the capacity must be defined by NCAP_BND; if not defined, assumed to be equal to the lower bound. Requires MIP.	Semi-continuous new capacity, lower bound. (See Section 5.9)	Applied to the semi-continuous investment 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	NCAP_ELIFE,	Years	Expected for all	Technical lifetime of a	Impacts all calculations

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)	COEF_CPT, COEF_RPTI, DUR_MAX	(0,∞); default value: G_TLIFE Default i/e: STD	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).	process.	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. The group (cg) can be the whole PCG or any individual commodity in the PCG, or 'ACT' (=PCG).	1) Conversion factor from units of activity to units of those flow variables that define the activity (primary commodity group), or, 2) Conversion multiplier representing the amount of flow(s) in the cg per 1 unit of activity.	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 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

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]
					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 Default i/e: 11	Combined limit on commodity production is derived as the sum of the process-specific productions multiplied by the inverse values of PRC_MARK. The constraint is applied to the annual production of commodity. Item can be a any desired label identifying the group.	Process group-wise market share, which defines a constraint for the combined market share of multiple processes in the total commodity production.	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]
PRC_REFIT (r,prc,p)	PRC_RCAP	Dimensionless [1,3]; default value: none Default i/e: n/a	Requires that early retirements are allowed in the model. The parameter value determines the type of the refurbishment option as follows: • Value=(±1 mod 2): Technology p will be a lifetime extension option (+1), or a retrofit option (−1), for the host prc • Value=2 for p=prc: refitted capacity in each period is forced to be equal to the retired capacity of the host prc	Defines a mapping of host process prc to a retrofit or lifetime extension option p in region r, where p is another process representing the refurbishment option. The value of the parameter determines the type of the refurbishment option (see column on the left).	Activates generation of the retrofit / lifetime extension equations (EQL_REFIT)
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	Affects cost coefficients in EQ_OBJ

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]
				objective function.	
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 (r,datayear,agg,cur,bd)	REG_CUMCST	Monetary unit [0,∞); default value: none Default i/e: MIG	The cost aggregations (agg) supported are listed in the set COSTAGG (see Table 1).	Bound on regional costs by type of cost aggregation.	EQ_BNDCST 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	*{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]
				identified by the index j.	
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 devices.	Efficiency of storage process.	Applied to the storage output flow (VAR_SOUT) in the commodity balance (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.
STG_MAXCYC (r,datayear,p)	NCAP_AF	Number of cycles [0,∞); default value: none Default i/e: STD	Can only be used for genuine storage processes. The limit can be exceeded by paying for additional replacement capacity, with a penalty cost equal to the investment annuity.	Defines the maximum number of storage cycles over the lifetime. Sets a limit for the total discharge divided by storage capacity.	Activates generation of the cycle limit/penalty equations (EQL_STGCCL).

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]
STG_SIFT (r,datayear,prc,com,ts)	ACT_TIME	Decimal fraction [0,∞); default value: none Default i/e: STD	Can only be used for a timeslice storage process. Levelized to the timeslice level of the process flow. Direct inheritance. By specifying com='ACT' one can define a limit in total shifting over a season, in proportion to demand.	Defines process prc as a load-shifting process, and limits the load shifting of demand com in timeslice ts to at most the fraction specified by the parameter value.	Activates generation of load shifting constraints (EQ(l)_SLSIFT).
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.
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	(Alias: CCAPM)	Capacity unit	Requires using ETL.	Maximum cumulative	Core ETL 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]
(r,teg)	CCOSTM	[open]; default value: none	For learning technologies teg when ETL is used.	capacity.	
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 [open]; default value: none	Requires using ETL. For learning technologies teg when ETL is used.	Initial specific investment costs.	Defines together with CCAP0 initial point of 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	Number of segments.	Influences the piecewise linear

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]
			teg when ETL is used. Currently limited to six segments by set kp.		approximation of the cumulative cost curve (EQ_COS, EQ_LA1, EQ_LA2).
TS_CYCLE (r,ts)	G_CYCLE	Number of days [1,∞); Default values: • 365 for ts=ANNUAL • 7 for any ts above the WEEKLY level • 1 for any ts above the DAYNITE level	Recommended to be used whenever timeslice cycles are different from the default, instead of changing G_CYCLE. Does not affect interpretation of availability factors for storage level, which thus remain to be according to G_CYCLE.	Defines the length of the timeslice cycles under timeslice ts, in days, and thereby also the number of timeslice cycles under each parent.	Affects the calculation of actual timeslice lengths and number of timeslice cycles in various equations, notably storage and dispatching equations.
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_COMNET	uc_n, uc_gmap_c	None	Used in user constraints.	Coefficient of the net	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]
(uc_n,side,r,datayear,c,s)		[open]; default value: none Default: i/e: STD	No inheritance/aggregation (might be changed in the future).	commodity production variable VAR_COMNET in a user constraint.	
UC_COMPRD (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 total commodity production variable VAR_COMPRD in a user constraint.	EQ(l)_UCXXX
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

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]
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 using the lim type lim=N.	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 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,	None [open];	Used in user constraints. No inheritance /	RHS constant with bound type of bd of a	RHS constant of user constraints, which are

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]
	uc_t_succ, uc_ts_each	default value: none Default i/e: MIG	aggregation, unless the target timeslice level is specified by UC_TSL. Direct inheritance, if the target timeslice level is specified by UC_TSL. 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.	user constraint.	generated for each specified region (uc_r_each), period (uc_t_each) and timeslice (uc_ts_each) (EQ(l)_UCRTS). 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	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)

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]
			types lim=UP/LO/FX. Non-binding (free) user constraints can be defined using the lim type lim=N.		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 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.
LEAD (t)	For each milestone year.	Time between milestone years **t**–1 and **t**, in years. For the first milestone year t1, LEAD(t1)=M(t1)–B(t1)+1.
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.

Internal parameter[31] (Indexes)	**Instances (Required / Omit / Special conditions)**	**Description**
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.
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).

Internal parameter[31] **(Indexes)**	**Instances (Required / Omit / Special conditions)**	**Description**
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).
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).

Internal parameter[31] (Indexes)	**Instances (Required / Omit / Special conditions)**	**Description**
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).
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.

Internal parameter[31] **(Indexes)**	**Instances (Required / Omit / Special conditions)**	**Description**
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).
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.

Internal parameter[31] (Indexes)	**Instances (Required / Omit / Special conditions)**	**Description**
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.
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.

Internal parameter[31] **(Indexes)**	**Instances (Required / Omit / Special conditions)**	**Description**
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, activate by setting RPT_OPT('OBJ','1')=1): 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, activate by setting RPT_OPT('OBJ','1')=1): 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. Do not include emission prices or by-product revenues in the calculation (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. <u>Include both emission prices and by-product revenues in the calculation</u> (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. <u>Include not only emission prices and by-product revenues, but also the revenues from the main product in the calculation</u> (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);

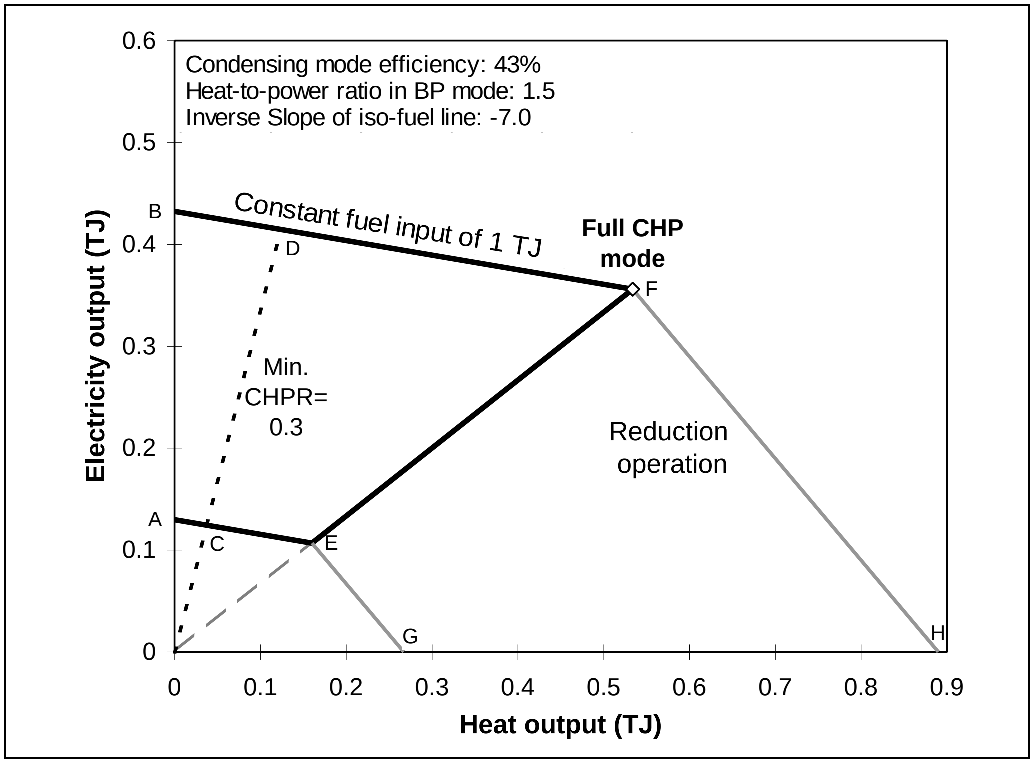


Figure 10: Illustration of basic CHP characteristics supported in TIMES.

- 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

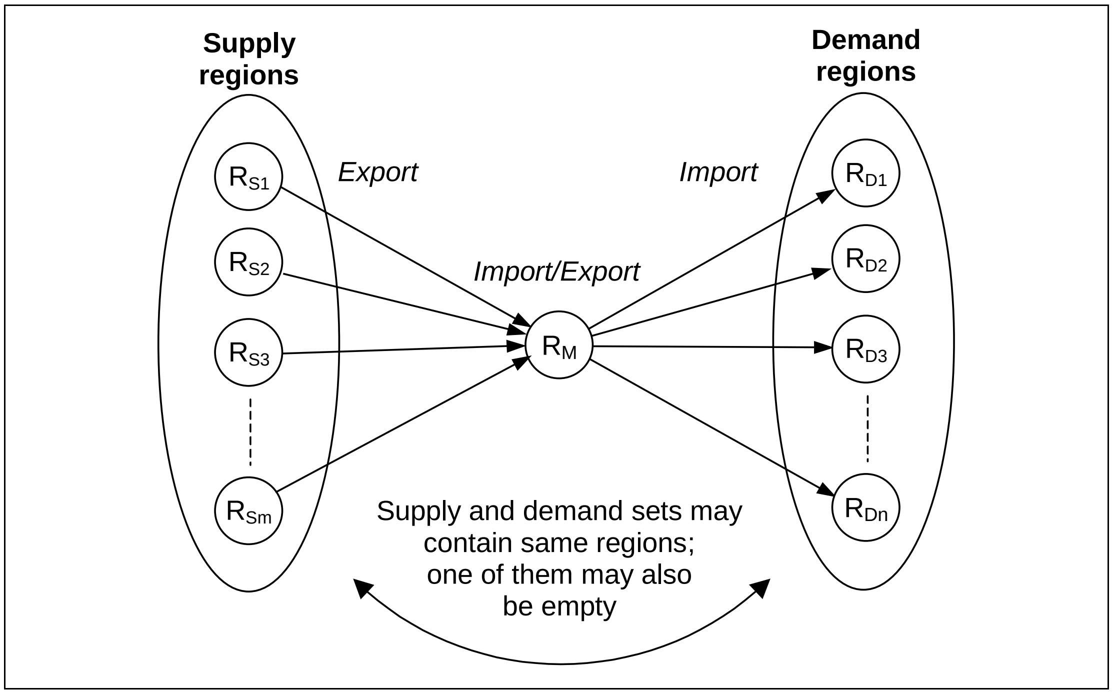


Figure 11. General structure of the pair-wise specification of the trading sub-network allowed in TIMES for a single exchange process.

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

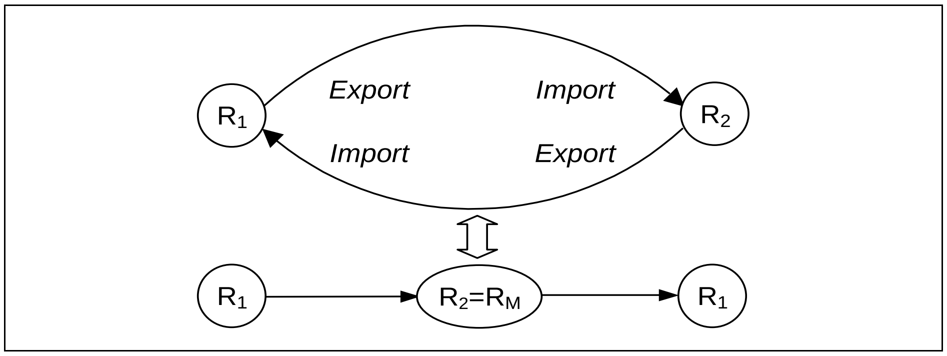


Figure 12. Case 1: Bi-lateral trade (both R_1 and R_2 qualify as R_M).

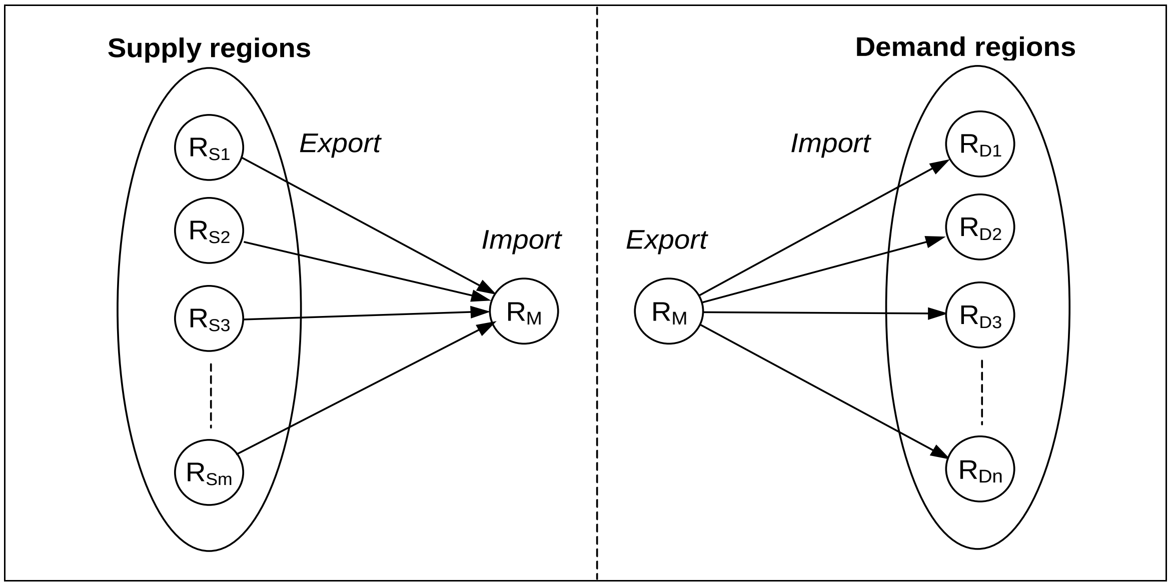


Figure 13. General structure of unidirectional trade into a single import region (Case 2, left) and multidirectional trade from a single export region (Case 3, right).

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. Nevertheless, 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

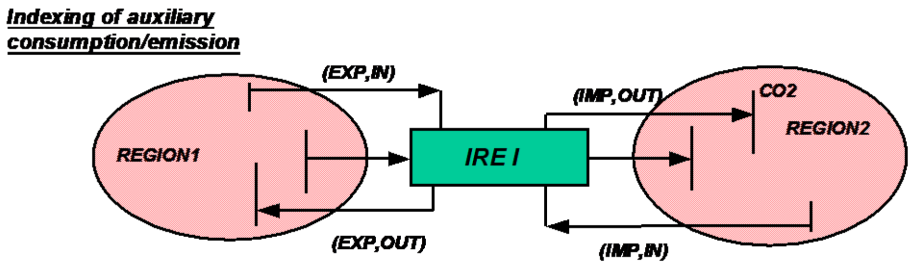
Input parameters specific to inter-regional exchange processes are listed in Table 19.

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).
STG_MAXCYC (r,y,p)	Defines a limit for the storage cycling within each period, by giving the maximum number of cycles over the full lifetime for process **p**, region **r**.
STG_SIFT (r,y,p,c,s)	Defines the storage process **p** as a special load-shifting storage process for commodity **c**, and defines the maximum fraction of shifted loads in proportion to the demand. See section 4.3.9 for additional information.
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(r,y,p,'ACT',tsl) 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 gross storage capacity equivalent to the amount produced by full power during one full year/week/day for SEASON/WEEKLY/DAYNITE level storage processes, respectively, assuming STG_EFF=1. 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 / STG_EFF, on the DAYNITE level.

<u>*Remarks:*</u>

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/NCAP_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.

4.3.9 Load-shifting storage processes

In TIMES, load-shifting for demands can be modelled by introducing load shifting processes, which are special storage processes where the input / output flows represent demand shifting upwards and downwards. For utilizing this built-in support for modelling demand shifting operation for a demand or final energy commodity *D*, the user would thus need to define a storage process *P*, such that the *D* is both an input and an output (or more generally, the input could also be another commodity upstream). In addition, the user only needs to define the proportional limits for the allowed demand shifting, by using the attribute *STG_SIFT(r,t,p,c,s)*, on the DAYNITE level, optionally also on next the level above. The dedicated load shifting constraints will then be generated for the process (see Equation *EQ_SLSIFT* for the constraint formulations).

This approach based on a storage process may be more convenient than manual constraints, because for the process the user will also be able to define investment costs, fixed O&M cost and variable costs for the demand shifting operation, and one would able to refer to the process activity and capacity variables easily in additional user constraints, if needed. Unlike for normal storage, the activity of the load-shifting processes is defined to be the output flow, and so the capcity represents the maximum level of the shifted loads. Preventing load shifting in either direction in any individual time-slices would also be easy by bounding the corresponding process flows to zero using *STGIN_BND* / *STGOUT_BND*.

The following constraints can be modelled for load-shifting processes:

- Maximum allowed fractions of loads shifted (required, defined by *STG_SIFT*);
- Seasonal balance equations (automatically generated);
- Standard capacity-activity equations (optional, when the capacity is modelled);
- User-defined balance constraints over sets of adjacent time-slices (optional, defined by *ACT_TIME*);
- Maximum allowed time to meet the shifted loads either in advance or with delay (optional, defined by *ACT_TIME*);
- Capacity bounds, activity bounds, flow bounds (optional).

The following types of costs can be modelled for load-shifting processes:

- Activity costs (cost on the discharge flow, using *ACT_COST*);
- Flow costs (cost on the charging and/or discharging flows, using *FLO_COST* and/or *FLO_DELIV*);
- Capacity cost (cost on the discharge load capacity, using *NCAP_COST*);
- Fixed O&M cost (cost on the discharge load capacity, using *NCAP_FOM*);
- Cost of shifting of one unit of demand load by one hour, forward (UP) and/or backward (LO) (using *ACT_CSTRMP*).

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
	VAR_UDP	Capacity unit dispatching load level variables
Commodity related		
	VAR_BLND	Blending variable (for oil refining)
	VAR_COMNET	Net amount of a commodity
	VAR_COMPRD	Gross production of a commodity (COM_IE applied)
	VAR_CUMCOM	Cumulative gross/net production of commodity
	VAR_ELAST	Variables used to linearize elastic demand curves
	VAR_GRIDELC	Transfer of power between grid nodes and demand nodes in the add-on grid formulation
	VAR_COMAUX	Phase angles in the DC power flow formulation
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

Category	Variable name	Brief description
		function
	VARTAXSUB	Parameter representing the variable taxes and subsidies of a regional component of the objective function
	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**.
Miscellaneous		
Load levels	VAR_RLD	Power load levels by user-defined supply category. Used in the residual load and ABS extensions.

<u>Notation for indexes</u>: 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.

[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.

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 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**, after applying the commodity efficiency COM_IE.

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 $\geq$ 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.

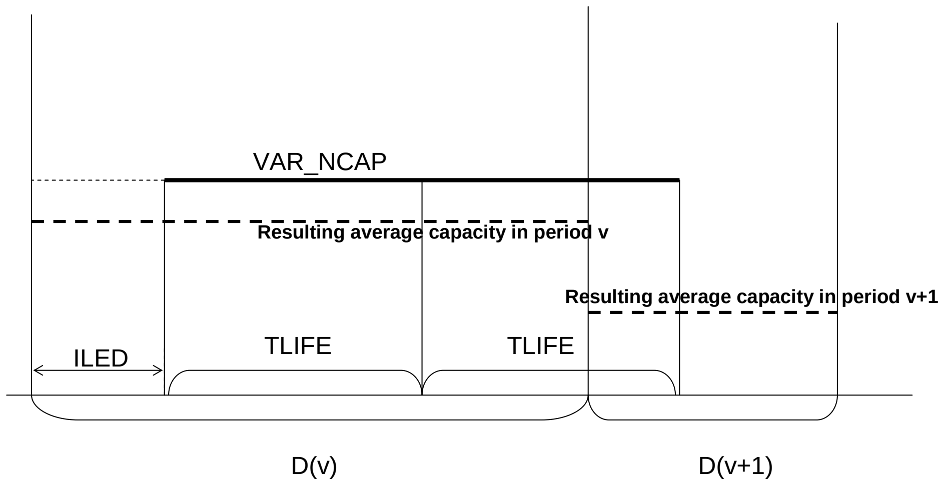

VAR_NCAP
Resulting average capacity in period v
Resulting average capacity in period v+1
ILED
TLIFE
TLIFE
D(v)
D(v+1)

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 VAR_UDP(r,v,t,p,s,l)

Definition: amount of on-line capacity by cycling/continuously (l='N'/'FX'), or load change due to ramping (l='UP'/'LO') in period **t** process **p** vintage index **v**.

Role: used for modeling capacity dispatching, start-up costs as well as ramping costs, but only when requested so by the user.

Bounds: Not available
User constraints: Not directly available

5.21 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> $\leq/=/\geq$ 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**.

5.21.1 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**).

5.21.2 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**.

5.21.3 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**.

5.21.4 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**.

5.21.5 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**.

5.21.6 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 (usually 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: $\boldsymbol{r(y)=1/(1+d(y))}$ (time dependent, although not shown in notation)
$\boldsymbol{d_s(t)}$:	technology specific discount rate (model year dependent)
$\boldsymbol{r_s(t)}$:	technology specific discount factor: $\boldsymbol{r_s(t)=1/(1+d_s(t))}$
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. $\boldsymbol{DISC(y,z) = \Pi_{u=z \text{ to } y-1}\, r(u)}$
$\boldsymbol{CRF_s(t)}$:	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 $\boldsymbol{CRF_s=\{1-r_s(t)\}/\{1-r_s(t)^{ELIFE}\}}$[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*. Note that below we in general assume *ILED*≥0, although *ILED* can also be negative (causing the lead-time be shifted ILED years backward).
$\boldsymbol{ILED_{Min}}$	***=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) $\ni z \in ALLYEARS$

$$\boxed{VAR\_OBJ(z) = \sum_{r \in REG} REG\_OBJ(z,r)} \qquad (A)$$

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) $\ni z \in ALLYEARS\ ,\ r \in REG$

$$\boxed{\begin{array}{l} REG\_OBJ(z,r) = \sum_{y \in (-\infty,+\infty)} DISC(y,z) \times \left\{ \begin{array}{l} INVCOST(y) + INVTAXSUB(y) + INVDECOM(y) + \\ FIXCOST(y) + FIXTAXSUB(y) + SURVCOST(y) + \\ VARCOST(y) + VARTAXSUB(y) + ELASTCOST(y) - \\ LATEREVENUES(y) \end{array} \right\} \\ \qquad -SALVAGE(z) \end{array}} \qquad (B)$$

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.

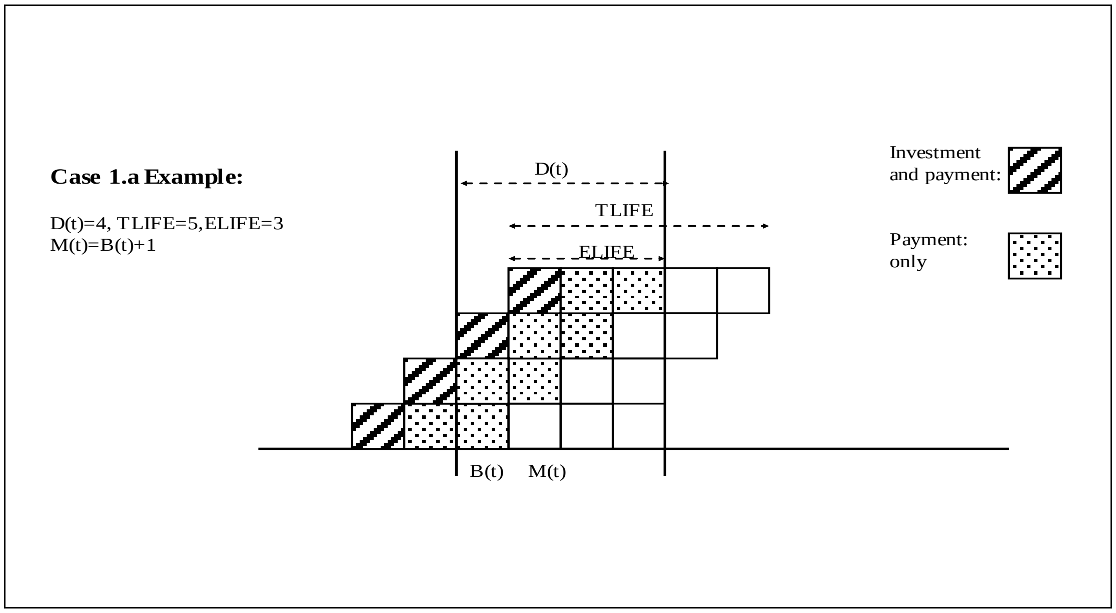
The four investment cases

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)

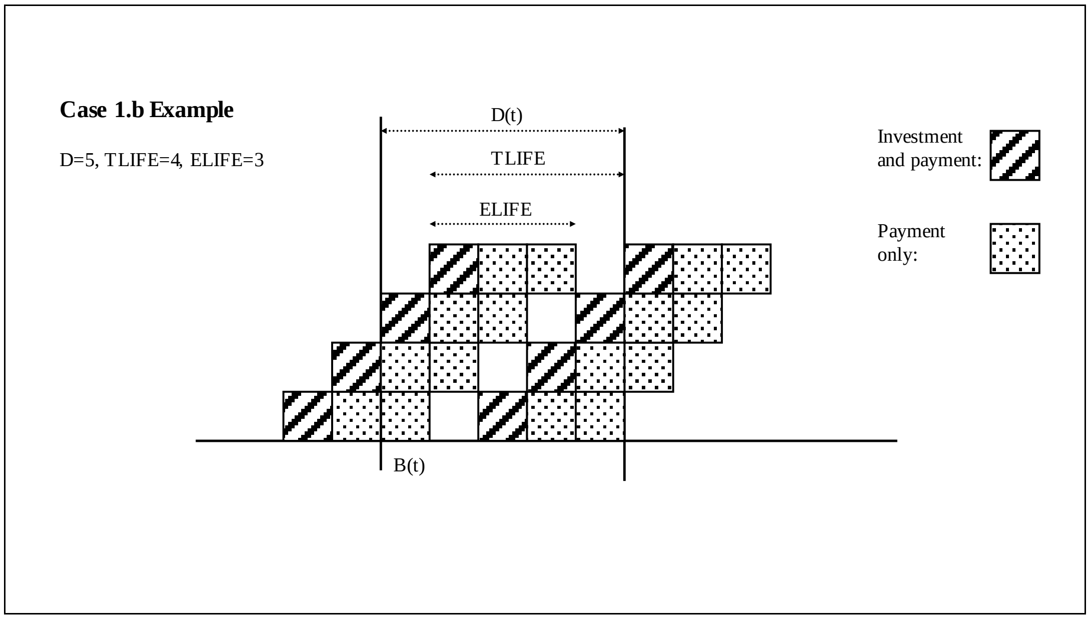
Comments: The summand represents the payment effected in year *y*, due to the investment increment that occurred in year *v* (recall that investment payments are spread over *ELIFE*). The summand consists of three factors: the first is the amount of investment in year *v*, the second is the capital recovery 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*:

$$\left\{\langle B(t) - TLIFE_t / 2\rangle , \langle B(t) - TLIFE_t / 2\rangle + C \times TLIFE_t + ELIFE_t - 2\right\}$$

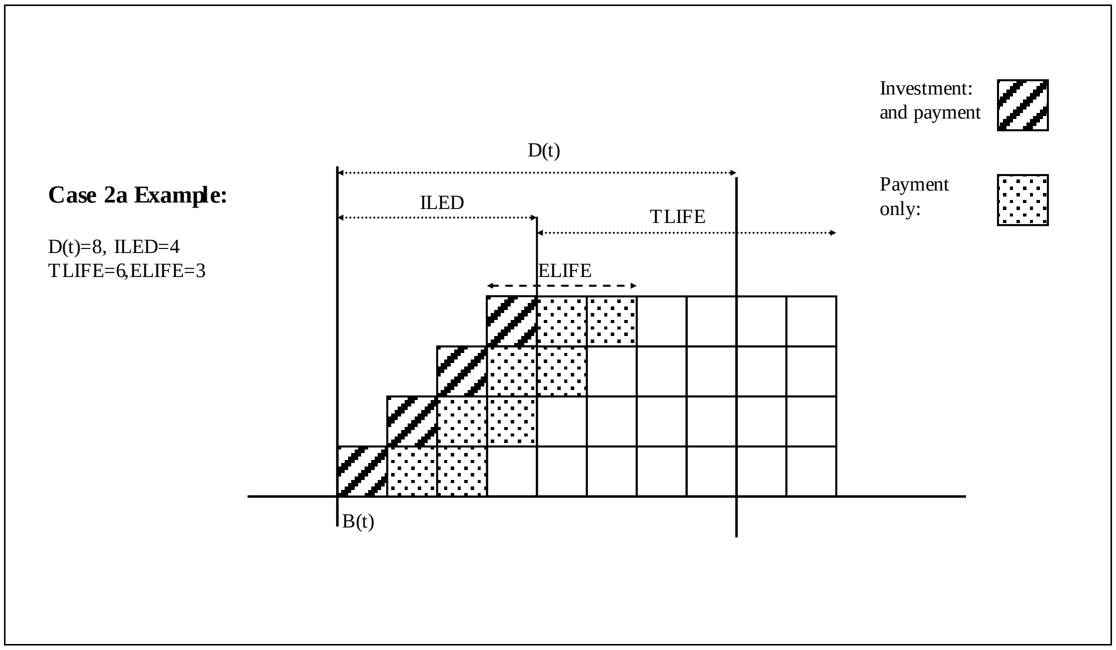
(I.1.b)

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

$$INVCOST(y) =$$

$$\sum_{\substack{t\in MILESTONEYEARS\\ t\leq T(y)}} INDIC(2.a) \times \sum_{k=Max\{B(t)+Min(0,ILED_t),y-ELIFE_t+1\}}^{Min(B(t)+Max(-1,Min(0,ILED_t),ILED_t-1),y)} \left(\frac{VAR\_NCAP_t}{ILED_t}\right) \times CRF_s \times NCAP\_COST_{B(t)+Max(0,ILED_t)} +$$

$$\sum_{v\in PASTYEARS} INDIC(2.a) \times \sum_{k=Max\{B(v)+Min(0,ILED_v),y-ELIFE_v+1\}}^{Min(B(v)+Max(-1,Min(0,ILED_v),ILED_v-1),y)} \left(\frac{NCAP\_PASTI_v}{ILED_v}\right) \times CRF_s \times NCAP\_COST_{B(v)+Max(0,ILED_v)}$$

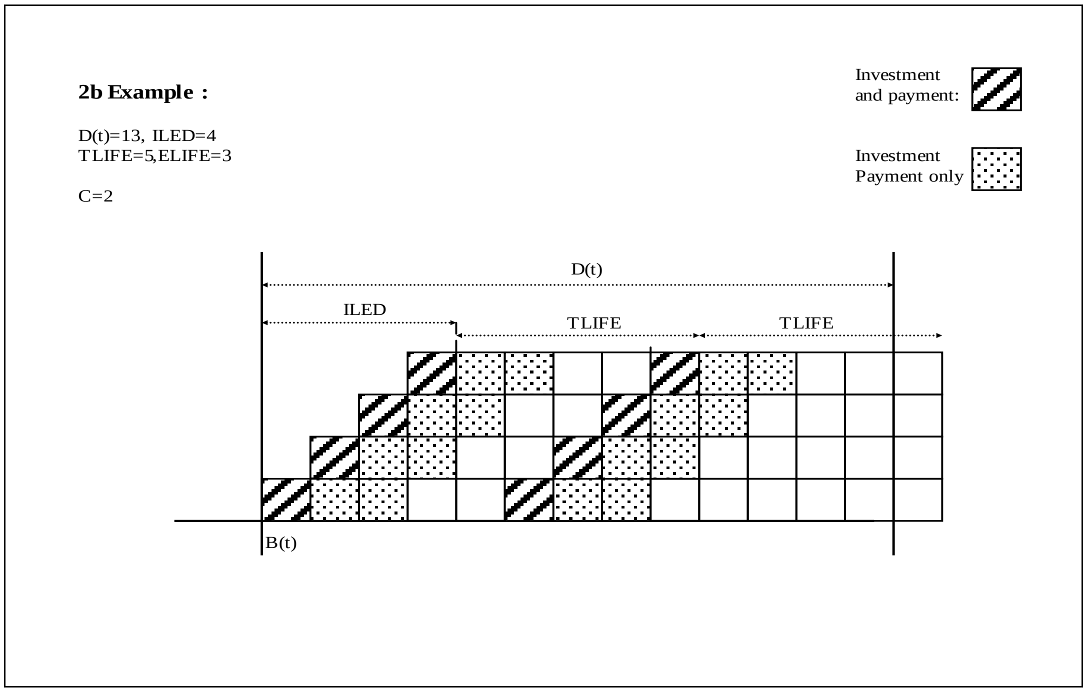
$Useful\ Range\ for\ y:$

$\{B(t),\ B(t)+ILED_t+ELIFE_t-2\}$ **(I.2.a)**

Comments: 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. Equation I.2.a also shows the impact of negative *ILED*s, which is simply a shift of the lead-time *ILED* years backwards.

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:

```
For $I = 1$ to $C$
    For $u = B(t) + (I-1) \cdot TLIFE_t$ to $B(t) + (I-1) \cdot TLIFE_t + ILED_t - 1$
        $NI_t(u) := NI_t(u) + \frac{NCAP\_COST_{B(t)+(I-1)\times TLIFE_t + ILED_t}}{ILED_t}$
    Next $u$
Next $I$
```

Step 3: Compute payments incurred in year *y*, and resulting from variable *VAR_NCAP*$_t$
For each *y* in range:

$$B(t) \le y \le B(t) + (C-1) \cdot TLIFE_t + ILED_t + ELIFE_t - 2$$

(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 $ILED_t \le ILED_{Min,t}$ and $TLIFE_t + ILED_t \ge D(t)$

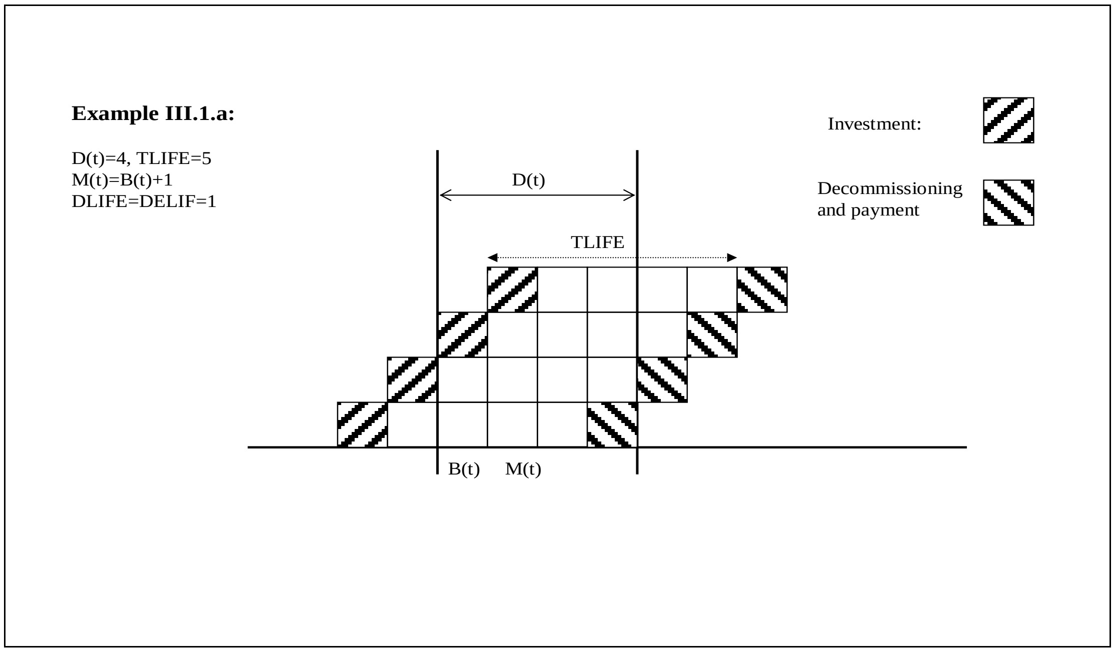
(Small divisible projects, non-repetitive, progressive investment in period)

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).*

$$INVDECOM(y) =$$
$$\sum_{\substack{t \in MILESTONES \cup PASTYEARS \\ t \le T(y)}} INDIC(1.a) \times \left(\frac{VAR\_NCAP_t}{D(t)} + NCAP\_PASTI_t \right) \times NCAP\_DCOST_{y-TLIFE_t}$$
$$\times \begin{cases} 1 & if \quad M(t) - D(t) + 1 + TLIFE_t + DLAG_t \le y \le M(t) + TLIFE_t + DLAG_t \\ 0 & otherwise \end{cases}$$

(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 $ILED_t \leq ILED_{Min,t}$ and $TLIFE_t + ILED < D(t)$

(Small projects, repeated investments in period)

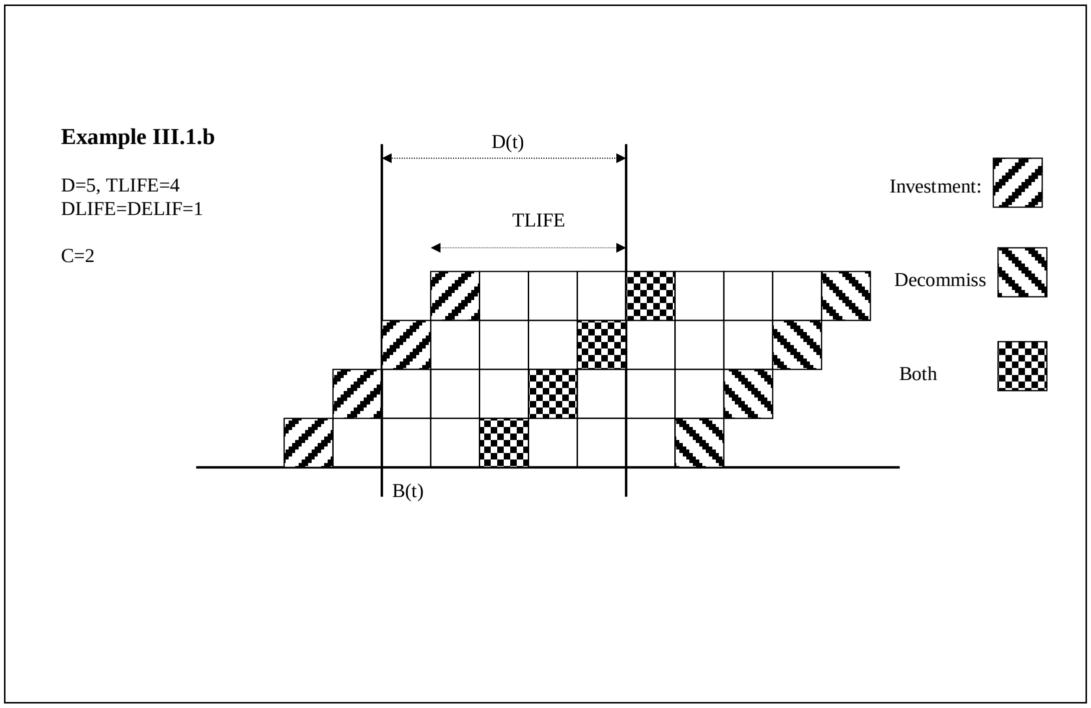
This cost expression is similar to I.1.b, but with payments shifted to the right by *TLIFE* (see example). The inner summation disappears because of the assumption that *DELIF=1*. Note also that past investments have no effect in this case, because this case does not arise *when D(t)=1*, which is always the case for past periods.

$$INVDECOM(y) = \sum_{\substack{t \in MILESTONES \\ t \leq T(y)}} INDIC(1.b) \times \left(\frac{VAR\_NCAP_t}{TLIFE_t} \right) \times NCAP\_DCOST_{y-TLIFE_t}$$

$$\times \begin{cases} 1 \; if \;\; B(t) + \left[\dfrac{TLIFE_t}{2} \right] \leq y \leq B(t) + \left[\dfrac{TLIFE_t}{2} \right] + C \cdot TLIFE_t - 1 \\ 0 \quad otherwise \end{cases}$$

$$where \; C = \left\langle \frac{D(t)}{TLIFE_t} \right\rangle$$

(III.1.b)



Case 2.a: $ILED_t > ILED_{Min,t}$ and $ILED_t + TLIFE_t \ge D(t)$
(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.

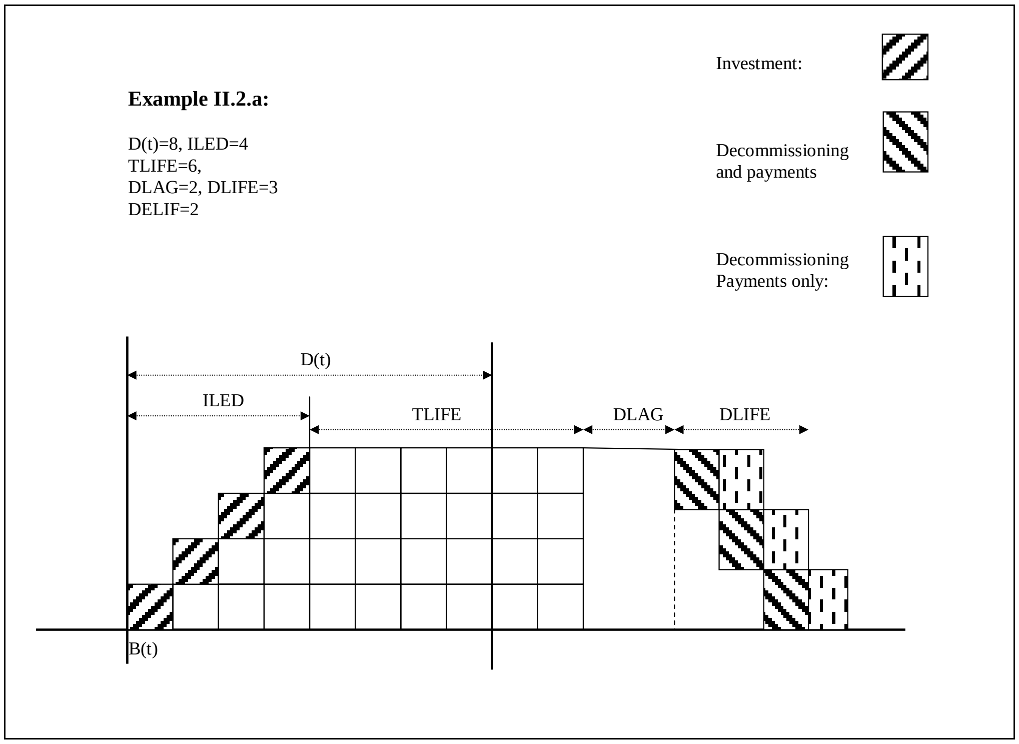
$$INVDECOM(y) =$$

$$\sum_{\substack{t\in MILESTONES\\ t\le T(y)}} INDIC(2.a)\times \sum_{k=Max\{B(t)+ILED_t+TLIFE_t+DLAG_t,\ y-DELIF_t+1\}}^{Min\{y,B(t)+ILED_t+TLIFE_t+DLAG_t+DLIFE_t-1\}} \left(\frac{VAR\_NCAP_t}{DLIFE_t}\right)\times CRF_s\times NCAP\_DCOST_{B(t)+ILED_t}$$

$$+\sum_{t\in PASTYEARS} INDIC(2.a)\ \times \sum_{k=Max\{t+TLIFE_t+DLAG_t,\ y-DELIF_t+1\}}^{Min\{y,t+TLIFE_t+DLAG_t+DLIFE_t-1\}} \left(\frac{NCAP\_PASTI_t}{DLIFE_t}\right)\times CRF_s\times NCAP\_DCOST_t$$

(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≤T(y)*)

Step 0: Initialization

$$P_t(y) := 0 \quad \forall \; B(t) + ILED_t + TLIFE_t + DLAG_t \le y \le same + (C-1) \times TLIFE_t + DLIFE_t + DELIF_t - 2$$

Where:

$$C = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

Step 1: Compute payment vector

```
$For\ \ I = 1\ \ to\ \ C$
        $For\ \ J = 1\ \ to\ \ DLIFE_t$
                $For\ \ L = 1\ to\ DELIF_t$
                $P_t\left(B(t) + ILED_t + I \times TLIFE_t + DLAG_t + J + L - 2\right) :=$
                $same + \dfrac{NCAP\_DCOST_{B(t)+ILED_t+(I-1)\times TLIFE_t}}{DLIFE_t}$
                $Next\ \ L$
        $Next\ \ J$
$Next\ \ I$
```

END ALGORITHM

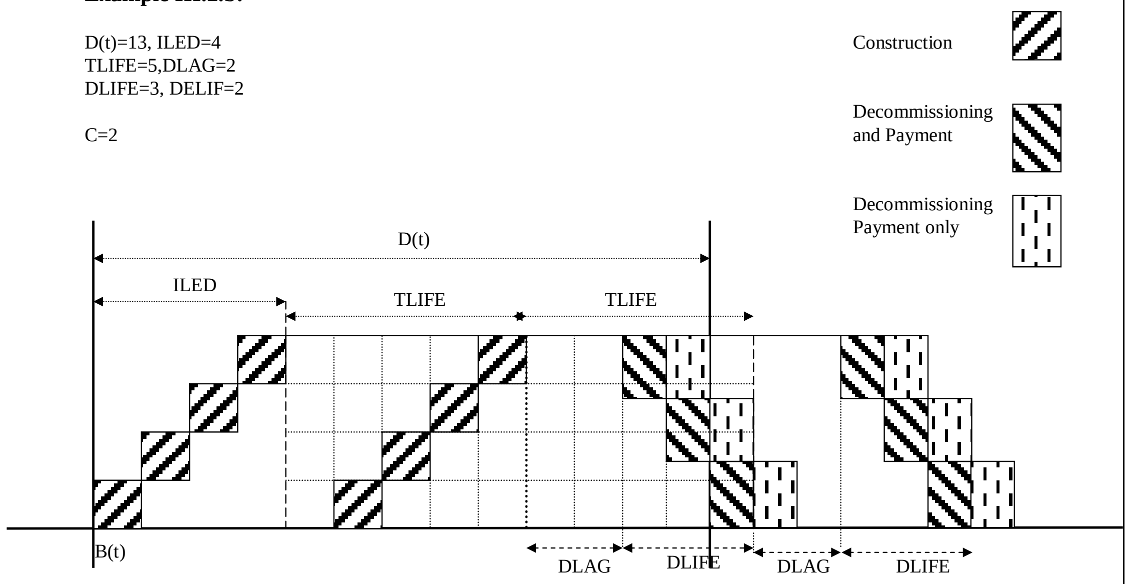
$$INVDECOM(y) = \sum_{t \in MILESTONES,\ \ t \le T(y)} INDIC(III.2.b) \times P_t(y) \times VAR\_NCAP_t \times CRF$$

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:

$$FIXCOST^{\circ}(r,y) = \sum_{(r,v,p)\in\mathbf{rtp}} \left(\begin{array}{l} VAR\_NCAP_{r,v,p} \quad (\ni \mathbf{t}_v) \\ +NCAP\_PASTI_{r,v,p} \\ -\sum\limits_{\substack{\mathbf{prc\_rcap}_{\mathbf{r,p}} \\ \mathbf{periodyr}_{\mathbf{t,y}}}} VAR\_SCAP_{r,v,t,p} \end{array} \right) \times CF_{r,v,p,y}$$

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)

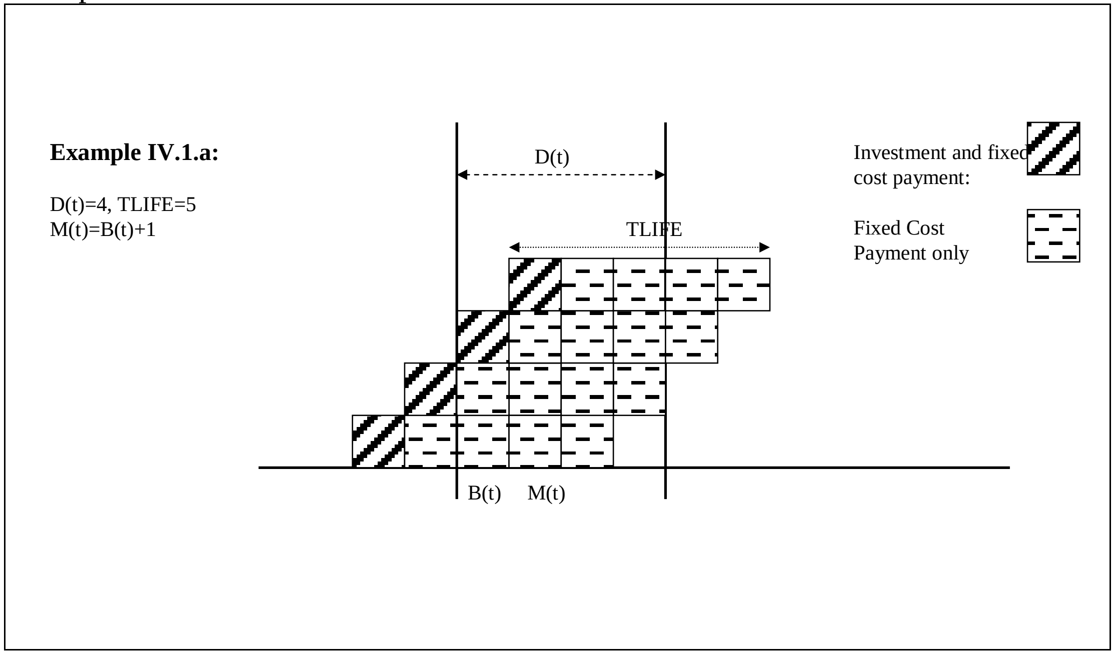
$EQ\_FIXCOST(y)$, $y \le EOH$

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).

$$FIXCOST(y) =$$
$$\sum_{t \in MILESTONYR \cup PASTYEARS} INDIC(1.a) \times \sum_{v=Max\{M(t)-D(t)+1,\, y-TLIFE_t+1\}}^{Min(M(t),y)} \left(\frac{VAR\_NCAP_t}{D(t)} + NCAP\_PASTI_t \right)$$
$$\times NCAP\_FOM_v \times SHAPE(v, y-v) \times MULTI(y)$$

$$\begin{aligned} &\textit{The useful range for y is:} \\ &\{M(t)-D(t)+1, M(t)+TLIFE_t-1\} \\ &\textit{and} \\ &y \le EOH \end{aligned} \qquad \textbf{(IV.1.a)}$$

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

$$C = \left\langle \frac{D(t)}{TLIFE_t} \right\rangle$$

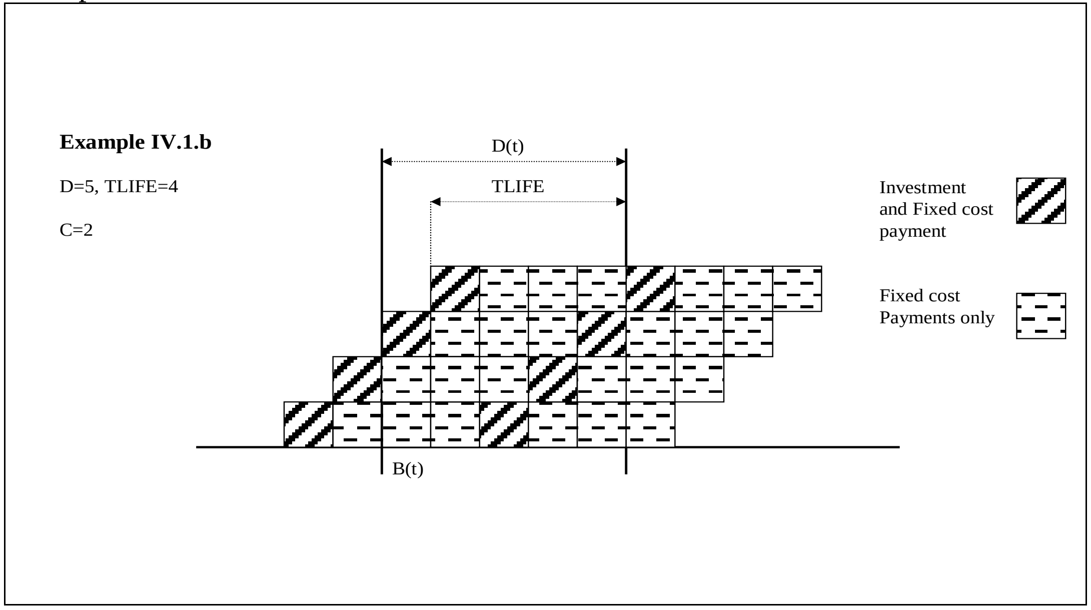
$Useful\ Range\ for\ y:$

$$\left\{ \left\langle B(t) - \frac{TLIFE_t}{2} \right\rangle, \left\langle B(t) - \frac{TLIFE_t}{2} \right\rangle + (C+1) \times TLIFE_t \right\}$$

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*.

$$FIXCOST(y) = \sum_{t \in MILESTONYR,\ t \leq T(y)} INDIC(2.a) \times \left(VAR\_NCAP_t\right) \times NCAP\_FOM_{B(t)+ILED_t}$$
$$\times \begin{Bmatrix} 1 \ if \ B(t) + ILED_t \leq y \leq B(t) + ILED_t + TLIFE_t - 1 \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad otherwise \end{Bmatrix} \times SHAPE(t, y - B(t) + ILED_t) \times MULTI(y)$$

$$+ \sum_{t \in PASTYEARS} INDIC(2.a) \times \left(NCAP\_PASTI_t\right) \times NCAP\_FOM_t$$

$$\times \begin{cases} 1 \ \ if \qquad t \leq y \leq t + TLIFE_t - 1 \\ 0 \qquad\qquad\qquad\qquad otherwise \end{cases} \times \ SHAPE(t \ \ , y - t) \times MULTI(y)$$

(IV.2.a)

Useful Range for y:

$$\{B(t) + ILED_t,\ B(t) + ILED_t + TLIFE_t - 1\}$$
$$and$$
$$y \leq EOH$$

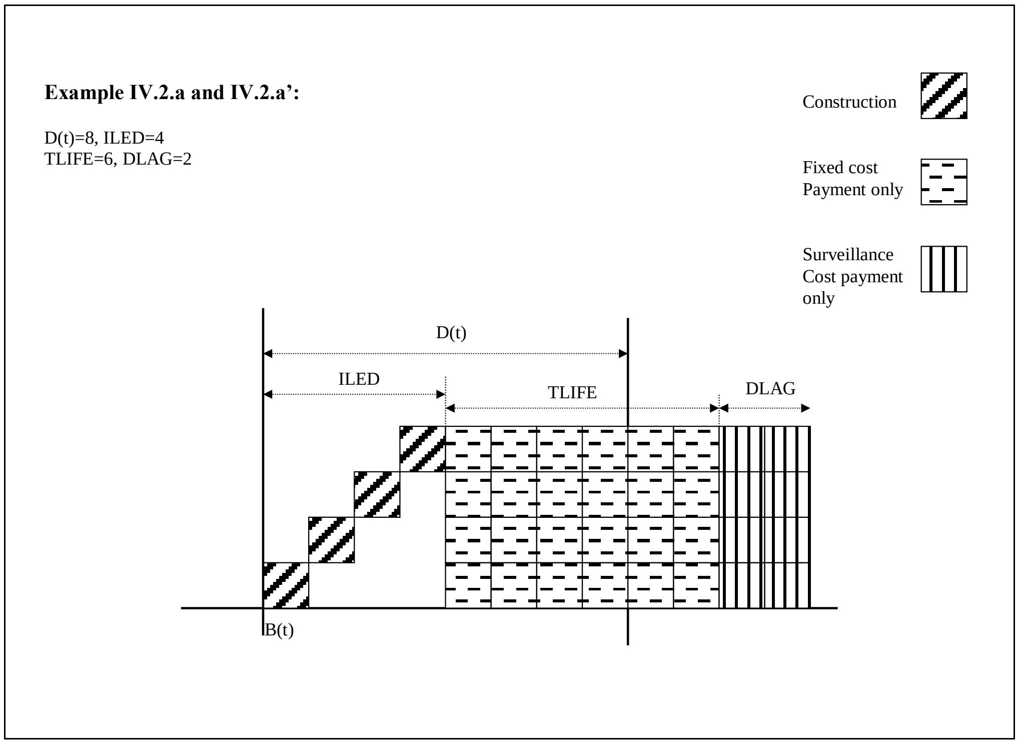
ii) SURVCOST (Surveillance cost for same case 2.a. See same example)

$$SURVCOST(y) = \sum_{\substack{t \in MILESTONYR, \\ t \leq T(y)}} INDIC(2.a) \times \left(VAR\_NCAP_t\right) \times NCAP\_DLAGC_{B(t)+ILED_t}$$
$$\times \begin{cases} 1 \ if \ B(t) + ILED_t + TLIFE_t \leq y \leq B(t) + ILED_t + TLIFE_t + DLAG_t - 1 \\ \ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad otherwise \end{cases}$$

$$+\sum_{t\in PASTYEARS} INDIC(2.a)\times\left(NCAP\_PASTI_t\right)\times NCAP\_DLAGC_t$$
$$\times\begin{cases}1\, if\, t+TLIFE_t\le y\le t+TLIFE_t+DLAG_t-1\\ 0 \qquad otherwise\end{cases}$$

(IV.2.a')

$$Useful\ Range\ for\ y:$$
$$\left\{B(t)+ILED_t+TLIFE_t,\ same\ +DLAG_t-1\right\}$$
$$note\ that\ y\ may\ be\ larger\ than\ EOH$$



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)

$$SURVCOST(y) = \sum_{\substack{t \in MILESTONES \\ t \le T(y)}} INDIC(2.b) \times \left(VAR\_NCAP_t\right) \times NCAP\_DLAGC_{B(t)+ILED_t+I \cdot TLIFE_t}$$

$$\times \begin{cases} 1 \ \ if\ B(t) + ILED_t + (I+1) \times TLIFE_t \le y \le same + DLAG_t - 1\ and\ 0 \le I \le C-1 \\ 0 \qquad otherwise \end{cases}$$

$$\textit{where}:$$

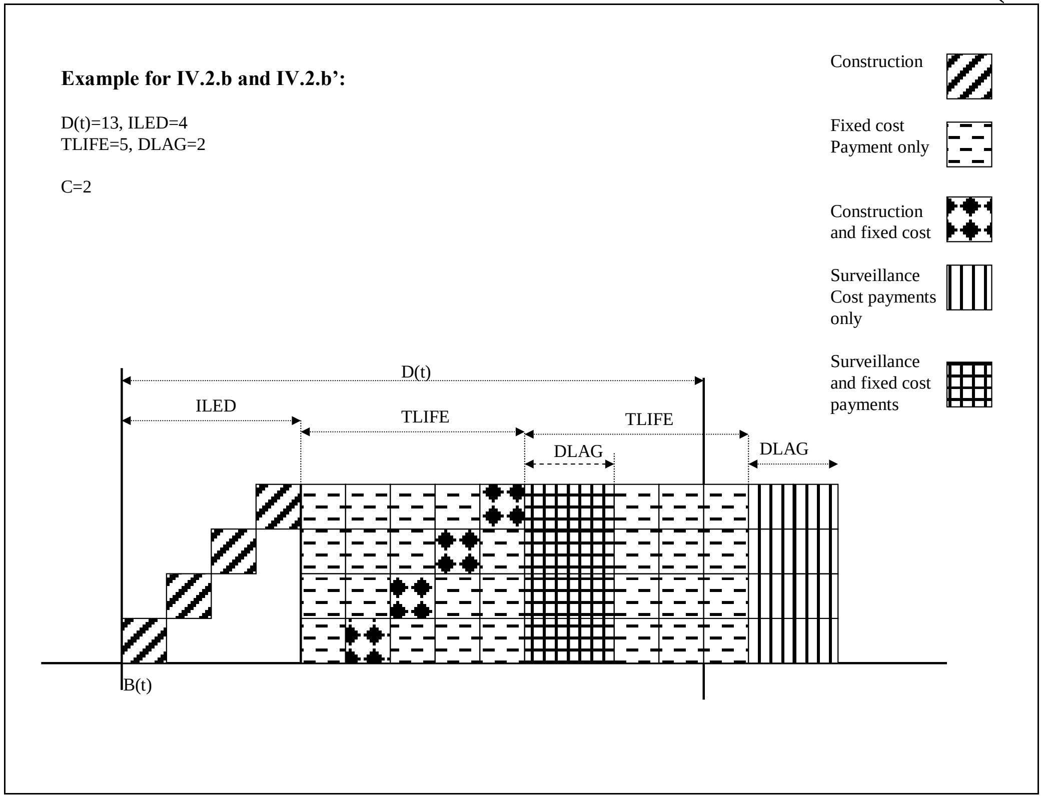
$$I = \left\lceil \frac{y - B(t) - ILED_t - TLIFE_t}{TLIFE_t} \right\rceil$$

$$\textit{and}$$

$$C = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

$$\textit{Note that } y \textit{ may exceed } EOH$$

(IV.2.b')



Example for IV.2.b and IV.2.b':
D(t)=13, ILED=4
TLIFE=5, DLAG=2
C=2
Construction
Fixed cost Payment only
Construction and fixed cost
Surveillance Cost payments only
Surveillance and fixed cost payments
D(t)
ILED
TLIFE
TLIFE
DLAG
DLAG
B(t)

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.

$$
\begin{aligned}
&VARCOST(y) = VAR\_XXX_{v,T(y)} \times XXX\_COST_y \\
&VARTAXSUB(y) = VAR\_XXX_{v,T(y)} \times (XXX\_TAX_y - XXX\_SUB_y) \\
&y \le EOH
\end{aligned}
\tag{VI}
$$

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, and Appendex D for formulations involving more generalized demand fuctions.

$$
\begin{aligned}
&ELASTCOST(y) = \\
&\sum_{j=1}^{COM\_STEP_{lo}} COM\_BPRICE_{T(y)} \times \left\{ \left(1 - \frac{(j-1/2)\times COM\_VOC_{lo,T(y)}}{COM\_STEP_{lo}} \right)^{\frac{1}{COM\_ELAST_{lo,T(y)}}} \right\} \times VAR\_ELAST_{lo,j,T(y)} \\
&- \sum_{j=1}^{COM\_STEP_{up}} COM\_BPRICE_{T(y)} \times \left\{ \left(1 + \frac{(j-1/2)\times COM\_VOC_{up,T(y)}}{COM\_STEP_{up}} \right)^{\frac{1}{COM\_ELAST_{up,T(y)}}} \right\} \times VAR\_ELAST_{up,j,T(y)} \\
&y \le EOH
\end{aligned}
\tag{VII}
$$

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.

However, the user can also request more accelerated functional depreciation in the value of the capacity, by defining $NCAP\_FDR_{r,v,p}$ (representing additional annual depreciation in the value). For simplicity, we apply the functional depreciation as an additional exponential discounter.

The computation of the salvage value therefore 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:

$$S(k,TL,FDR) = 0 \qquad if\, k + TL \le EOH$$

$$S(k,TL,FDR) = 1 \qquad if\, k > EOH$$

$$S(k,TL,FDR) = \frac{\left((1+d)\cdot \exp(FDR)\right)^{TL-EOH-1+k} - 1}{\left((1+d)\cdot \exp(FDR)\right)^{TL} - 1} \quad otherwise$$

where d is the general discount rate and FDR is the optional functional depreciation 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:

$$\left(1+d\right)^{EOH+1-k}$$

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:

$$\frac{CRF_s}{CRF} = \frac{\left(1-\frac{1}{1+i_s}\right)\times\left(1-\frac{1}{(1+i)^{ELIFE}}\right)}{\left(1-\frac{1}{1+i}\right)\times\left(1-\frac{1}{\left(1+i_s\right)^{ELIFE}}\right)}$$

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

$$SAL(k,TL) = 0 \qquad if\, k+TL \le EOH$$

$$SAL(k,TL) = \frac{CRF_s}{CRF} \qquad if\; k \ge EOH+1$$

$$SAL(k,TL) = \frac{1-(1+d)^{EOH+1-k-TL}}{1-(1+d)^{-TL}} \times \frac{CRF_s}{CRF} \times \frac{S(k,TL,FDR)}{S(k,TL,0)} \qquad otherwise$$

where d is the general discount rate, CRF_s is the technology-specific capital recovery factor and FDR is the functional depreciation 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 $v+TLIFE_t \le EOH+1$

(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)

$$SALVINV(EOH+1) = \sum_{t \in MILESTONESYEARS} VAR\_NCAP_t \times NCAP\_COST_{B(t)+ILED_t} \times SAL(B(t)+ILED_t)$$

$$Note\ that\ SAL\ (B(t)+ILED_t) = 0\ whenever\ B(t)+ILED_t + TLIFE_t \leq EOH+1$$

(VIII.2.a)

Case 2.b: $ILED > ILED_{Min,t}$ and $TLIFE_t + ILED_t < D(t)$
(Large, indivisible Projects, repeated investments in period)

$$SALVINV(EOH+1) = \sum_t VAR\_NCAP_t \times NCAP\_COST_{B(t)+(C-1)\times TLIFE_t + ILED_t} \times SAL\big(B(t)+(C-1)\times TLIFE_t + ILED_t\big)$$

$$Note\ again\ that\ SAL\big(B(t)+(C-1)\times TLIFE_t + ILED_t\big) = 0\ \ whenever\ B(t)+(C-1)\times TLIFE_t + ILED_t + TLIFE_t \leq EOH+1$$

(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:

$(1+d)^{EOH+1-l}$ where l 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 $ILED_t \leq ILED_{Min,t}$ and $TLIFE_t + ILED_t \geq D(t)$

(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 $v + TLIFE \leq EOH + 1$ **(IX.1.a)**

Case 1.b $ILED_t \leq ILED_{Min,t}$ and $TLIFE_t + ILED < D(t)$

(Small Projects, repeated investments in period)

$$SALVDECOM(EOH+1) =$$
$$\sum_t INDIC(1.b) \times \sum_{v=B(t)-\rangle TL/2\langle+(C-1)\times TLIFE_t}^{B(t)-\rangle TL/2\langle+C\times TLIFE_t-1} \frac{VAR\_NCAP_t}{TLIFE_t} \times NCAP\_DCOST_v \times SAL(v, v+TLIFE_t)$$

$Note\ again\ here\ that\ SAL(k,l)\ equals\ 0\ if\ k + TLIFE \leq EOH + 1$

(IX.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)

$$SALVDECOM(EOH+1) =$$
$$\sum_{t \in MILESTONESYEARS} INDIC(2.a) \times VAR\_NCAP_t \times NCAP\_COST_{B(t)+ILED_t} \times \sum_{l=B(t)+TLIFE+DLAG}^{same+DLIFE-1} SAL(B(t)+ILED_t, l)$$

$Note\ that\ SAL\ is\ 0\ whenever\ B(t) + ILED_t + TLIFE_t \leq EOH + 1$

(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* $B(t) + C \times TLIFE_t + ILED_t \leq EOH + 1$

(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 $ILED_t + TLIFE_t \geq D(t)$

(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 $k + m \leq EOH + 1$.

(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 $k + m \le EOH + 1$.

(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) $y \ge EOH+1$

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:

$$CRF_{1,mid} = CRF_{beg} \times (1+d(T(\mathrm{y})))^{0.5}$$

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:

$$CRF_{mid}^{proper} = CRF_{beg} \times (1+d_S(T(y)))^{0.5} \qquad \text{(XI.1)}$$

$$\begin{aligned} CRF_{1,mid} &= CRF_{mid}^{proper} \times (1+d(T(y)))^{0.5} \times (1+d(T(y)))^{-0.5} = \\ &\quad CRF_{beg} \times (1+d_S(T(y)))^{0.5} \end{aligned} \qquad \text{(XI.2)}$$

$$\begin{aligned} CRF_{2,mid} &= CRF_{mid}^{proper} \times (1+d(T(y)))^{-0.5} = \\ &\quad CRF_{beg} \times (1+d(T(y)))^{-0.5} \times (1+d_S(T(y)))^{0.5} \end{aligned} \qquad \text{(XI.3)}$$

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). The constraints related to the advanced unit commitment formulation are described in detail in a separate document "*Dispatching and Unit Commitment in TIMES*", and the constraints related to the balancing services extension are described in a separate document "*Enhancing the flexibility in TIMES: Introducing Ancillary Services Markets*".

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)
EQ_ACTRMPC	Defines the change in the load by ramping up or down in the dispatching phase, for which ramping costs are to be applied (ACT_CSTRMP).
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

Equation Name	Short description
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
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)_FLOMRK	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_REFIT	Implements the retrofit and life extension constraints, such that the capacity of the retrofit/life-extension options is, when commissioned, at most equal to the available remaining capacity of the host process.
EQL_SCAP	Bounds the amount of capacity salvaged if early retirements are active.
EQ_SLSIFT	Implements the load shifting constraints (see 6.3.37 EQ_SLSIFT).
EQ_STGAUX	Establishes an equality relationship between storage main flows or activity and an auxiliary storage flow
EQL_STGCCL	Defines an upper bound for storage cycling, by imposing a replacement cost if the annual output divided by storage capacity exceeds the average number of storage cycles in a year, as calculated from STG_MAXCYC.
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

Equation Name	**Short description**
EQ_SDLOGIC	Logical relationship between decision variables in the advanced unit commitment formulation (see separate documentation).
EQ_SUDUPT	Selection of start up type a according to non-operational time in the advanced unit commitment formulation (see separate documentation).
EQ_SDSLANT	Slanting equation for start-up and shut-down phase in the advanced unit commitment formulation (see separate documentation).
EQ_SDMINON	Minimum on-line capacity constraints in the advanced unit commitment formulation (see separate documentation).
EQ_SUDLOAD	Load during start-up/shut down phase of the unit (linear growth) in the advanced unit commitment formulation (see separate documentation).
EQ_SUDTIME	Minimum on-line / off-line time constraint in the advanced unit commitment formulation (see separate documentation).
EQ_SUDPLL	Efficiency losses due to start-up/shut-down of the unit in the advanced unit commitment formulation (see separate documentation).
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
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: $\leq$

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:

$$BND\_ELAST_{r,t,c,s,j,l} \ni COM\_STEP_{r,c,l} \wedge (s \in \mathbf{com\_ts_{r,c,s}})$$

$$VAR\_ELAST_{r,t,c,s,j,l} \leq \frac{COM\_PROJ_{r,t,c} \times COM\_FR_{r,t,c,s} \times COM\_VOC_{r,t,c,l}}{COM\_STEP_{r,c,l}}$$

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.

Activity must exist and process is available in period t

All timeslices at or above **prc_tsl**

Equation:

$$EQ(l)\_ACTBND_{r,t,p,s} \quad \ni ACT\_BND_{r,t,p,s,bd} \wedge \mathbf{rtp\_vara_{r,t,p}} \wedge \mathbf{rps\_prcts_{r,p,s}}$$

$$\wedge \left(p \in \mathbf{prc\_vint_{r,p}} \vee s \notin \mathbf{prc\_ts_{r,p,s}}\right)$$

either p is vintaged or the bound is applied for a n exact slice of p

$$\sum_{v \in \mathbf{rtp\_vintyr}} \quad \sum_{s2 \in \mathbf{prc\_ts}} VAR\_ACT_{r,v,t,p,s2} \; \{=;\leq;\geq\} \; ACT\_BND_{r,t,p,s,l}$$

*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:

$$EQE\_ACTEFF_{r,v,t,p,cg,io,s} \ni (\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \wedge \neg \mathbf{rp\_inout}_{\mathbf{r,p,io}} \wedge ACT\_EFF_{r,v,p,cg,s})$$

$$\sum_{\substack{\mathbf{com\_gmap}_{\mathbf{r,cg,c}} \\ \mathbf{rtcp\_varf}_{\mathbf{r,t,p,c,s}}}} \left(\begin{array}{l} VAR\_FLO_{r,v,t,c,ts} \times \\ \left(\begin{array}{ll} ACT\_EFF_{r,v,p,c,ts} & if\ ACT\_EFF_{r,v,p,c,ts}\ given \\ \quad 1 & otherwise \end{array} \right) \\ \times RTCS\_TSFR_{r,t,c,s,ts} \end{array} \right)$$

$$=$$

$$\sum_{\substack{\mathbf{rpc\_pg}_{\mathbf{r,p,c}} \\ \mathbf{prc\_ts}_{\mathbf{r,p,ts}}}} \left(\begin{array}{l} \left(\begin{array}{ll} VAR\_ACT_{r,v,t,p,ts} & if\ RP\_PGACT_{r,p} \\ \dfrac{VAR\_FLO_{r,v,t,p,c,ts}}{PRC\_ACTFLO_{r,v,p,c}} & otherwise \end{array} \right) \times \\ \left(\begin{array}{ll} 1/ACT\_EFF_{r,v,p,cg,ts} & if\ ACT\_EFF_{r,v,p,cg,ts}\ given \\ \quad 1 & otherwise \end{array} \right) \times \\ \left(\begin{array}{ll} 1/ACT\_EFF_{r,v,p,c,ts} & if\ ACT\_EFF_{r,v,p,c,ts}\ given \\ \quad 1 & otherwise \end{array} \right) \\ \times RTCS\_TSFR_{r,t,c,s,ts} \end{array} \right) +$$

$$\sum_{\mathbf{prc\_ts}_{\mathbf{r,p,ts}}} \left(\begin{array}{ll} VAR\_UPS_{r,v,t,p,ts,'FX'} \times & \\ ACT\_LOSPL_{r,v,p,'FX'} & if\ ACT\_LOSPL_{r,v,p,'FX'}\ given \\ \times RS\_FR_{r,s,ts} & \end{array} \right)$$

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:

$$EQ\_ACTRAMP_{r,v,t,p,s,'UP'} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,'UP'} > 0))$$

$$\left(\frac{VAR\_ACT_{r,v,t,p,s}}{G\_YRFR_{r,s}} - \frac{VAR\_ACT_{r,v,t,p,s-1}}{G\_YRFR_{r,s-1}} - \left(VAR\_UPS_{r,v,t,p,s,'UP'} - VAR\_UPS_{r,v,t,p,s,'LO'}\right)\cdot ACT\_UPS_{r,v,p,s,'FX'}\right)\times$$

$$\frac{2\cdot RS\_STGPRD_{r,s}}{8760\times\left(G\_YRFR_{r,s} + G\_YRFR_{r,s-1}\right)} \le \left(VAR\_NCAP_{r,v,p} - \sum_{ts\in SUP(s)\cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'}\right)\times$$

$$COEF\_CPT_{r,v,t,p} \times PRC\_CAPACT_{r,p} \times ACT\_UPS_{r,v,p,s,'UP'}$$

$$EQ\_ACTRAMP_{r,v,t,p,s,'LO'} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,'LO'} > 0))$$

$$\left(\frac{VAR\_ACT_{r,v,t,p,s-1}}{G\_YRFR_{r,s-1}} - \frac{VAR\_ACT_{r,v,t,p,s}}{G\_YRFR_{r,s}} - \left(VAR\_UPS_{r,v,t,p,s,'LO'} - VAR\_UPS_{r,v,t,p,s,'UP'}\right)\cdot ACT\_UPS_{r,v,p,s,'FX'}\right)\times$$

$$\frac{2\cdot RS\_STGPRD_{r,s}}{8760\times\left(G\_YRFR_{r,s} + G\_YRFR_{r,s-1}\right)} \le \left(VAR\_NCAP_{r,v,p} - \sum_{ts\in SUP(s-1)\cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'}\right)\times$$

$$COEF\_CPT_{r,v,t,p} \times PRC\_CAPACT_{r,p} \times ACT\_UPS_{r,v,p,s,'LO'}$$

6.3.7 Equation: EQ_ACTRAMPC

Indices: **region (r), vintage year (v), period (t), process (p), time slice (s)**

Type: =

Related variables: **VAR_ACT, VAR_UPS, VAR_UDP**

Related equations: **EQE_CAPLOAD**

Purpose: The increase or decrease in output load of a process may result in ramping costs per unit of load (i.e. unit of capacity). The costs are directly applied to the differences in the load level between successive time slices during the dispatching phase and at the process operating level. This equation defines the changes in loads by ramping up or down in the dispatching phase, for which ramping costs are to be applied. The ramping costs are specified with the input parameter *ACT_CSTRMP*(r,v,p,bd,cur), per unit of load change between successive timeslices.

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*.

The following equation calculates the changes in the load $var\_ldc_{r,v,t,p,s,bd}$ during the dispatching phase:

$$\left(\frac{var\_act_{r,v,t,p,s-1}}{G\_YRFR_{r,s-1}}-\frac{var\_act_{r,v,t,p,s}}{G\_YRFR_{r,s}}\right)\cdot\frac{1}{CAPACT_{r,p}}=var\_ldc_{r,v,t,p,s,LO}-var\_ldc_{r,v,t,p,s,UP}-$$
$$ACT\_MINLD_{r,v,p}\times\left(var\_off_{r,v,t,p,s-1}-var\_off_{r,v,t,p,s}\right),\ \forall s\in PRC\_TS_{r,p}$$

In the above equation the variable $var\_ldc_{r,v,t,p,s,LO}$ holds load decreases, while the variable $var\_ldc_{r,v,t,p,s,UP}$ holds load increases. The two variables appear together in the equation since at each time slice s only one of the two variables can be set (i.e. the load can either increase or decrease).

Having calculated the changes in the dispatchable load, the associated ramping costs are entered into the objective function as the sum of the load changes $var\_ldc_{r,v,t,p,s,UP}$ multiplied by the cost attribute $ACT\_CSTRMP_{r,v,p,UP,cur}$ for the ramping up costs, and the sum of the load changes $var\_ldc_{r,v,t,p,s,LO}$ multiplied by the cost attribute $ACT\_CSTRMP_{r,v,p,LO,cur}$ for the ramping down costs. The costs are discounted to the base year:

$$obj_{RMPC}=$$
$$\sum_{r,t}\left(NPV_{r,t}\cdot\sum_{p,v,bd\in\{LO,UP\}}ACT\_CSTRMP_{r,v,p,bd,cur}\cdot\sum_{s\in PRC\_TS_{r,p}}var\_ldc_{r,v,t,p,s,bd}\right)$$

6.3.8 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

$$EQ\_ACTUPC_{r,v,t,p,tsl,'UP',s} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge \mathbf{ts\_group}_{r,tsl,s} \wedge (ACT\_TIME_{r,t,p,'UP'} > 0))$$

$$\sum_{ts \in C(P(s))} VAR\_UPS_{r,v,t,p,ts'UP'} \times (\mathrm{mod}(Hour(s) - Hour(ts), 24) < ACT\_TIME_{r,v,p,'UP'}) \leq$$

$$\left(VAR\_NCAP_{r,v,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \right)$$

$$EQ\_ACTUPC_{r,v,t,p,tsl,'LO',s} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge \mathbf{ts\_group}_{r,tsl,s} \wedge (ACT\_TIME_{r,t,p,'LO'} > 0))$$

$$\sum_{ts \in C(P(s))} VAR\_UPS_{r,v,t,p,ts,'LO'} \times (\mathrm{mod}(Hour(s) - Hour(ts),24) < ACT\_TIME_{r,v,p,'LO'}) \leq$$

$$\left(VAR\_UPS_{r,v,t,p,s,'N'}\right)$$

Case B: Maximum number of start-up cycles within parent timeslice cycle

$$EQ\_ACTUPC_{r,v,t,p,tsl,'N',s} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \left(s \in \bigcup_{ts} \left\{ P(ts) \mid \mathbf{prc\_ts}_{r,p,ts} \right\} \right)$$

$$\wedge \mathbf{ts\_group}_{r,tsl,s} \wedge (ACT\_UPS_{r,v,p,'ANNUAL','N'} > 0))$$

$$\sum_{ts \in C(s)} VAR\_UPS_{r,v,t,p,ts,'UP'} \leq$$

$$\left(VAR\_NCAP_{r,v,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \right) \times ACT\_UPS_{r,v,p,'ANNUAL','N'}$$

6.3.9 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:

$$EQE\_ACTUPS_{r,v,t,p,tsl,s} \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge UPS^{+}_{r,p,tsl} \wedge \mathbf{ts\_group}_{r,tsl,s})$$

$$VAR\_UPS_{r,v,t,p,s,'UP'} - VAR\_UPS_{r,v,t,p,s,'LO'}$$

$$=$$

$$VAR\_UPS_{r,v,t,p,s-1,'N'} - VAR\_UPS_{r,v,t,p,s,'N'}$$

6.3.10 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'

$$EQL\_ACTUPS_{r,v,t,p,tsl,'FX',s} \ni \left(\begin{array}{l} \mathbf{rtp\_vintyr}_{r,v,t,p} \wedge UPS^{+}_{r,p,tsl} \wedge \\ \mathbf{ts\_group}_{r,tsl,s} \wedge s \in \left\{ \bigcup_{sl} P(sl) \mid sl \in UPS^{+}(p) \right\} \end{array}\right)$$

$$VAR\_UPS_{r,v,t,p,s,'FX'} \leq \sum_{ts \in C(s)} VAR\_UPS_{r,v,t,p,ts,'UP'}$$

6.3.11 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.*

$$EQ(l)\_ASHAR_{r,v,t,p,c,cg,s} \quad \ni \left(\begin{array}{l} \mathbf{rtp\_vintyr_{r,v,t,p}} \wedge \left(\mathbf{rpcs\_var_{r,p,c,s}} \vee \left((c='ACT') \wedge \mathbf{annual_s}\right)\right) \\ \wedge \sum\limits_{ts\_map_{r,s,ts}} FLO\_ASHAR_{r,v,p,c,cg,ts,bd} \end{array} \right)$$

Case A: Standard processes:

$$\sum_{\mathbf{rps\_s2_{r,p,sl}}} FLO\_ASHAR_{r,v,p,c,cg,sl,bd} \times RS\_FR_{r,s,sl} \times$$

$$\left(\begin{array}{l} \sum\limits_{com \in cg} \sum\limits_{\mathbf{rtpcs\_varf_{r,t,p,com,ts}}} VAR\_FLO_{r,v,t,p,com,ts} \times RTCS\_TSFR_{r,t,p,com,sl,ts} + \\ \sum\limits_{com \in \left\{ \begin{array}{l} \mathbf{rpc\_pg_{r,p,com}} \\ |cg='ACT' \end{array} \right\}} \sum\limits_{\mathbf{prc\_ts_{r,p,ts}}} \dfrac{VAR\_FLO_{r,v,t,p,com,ts}}{PRC\_ACTFLO_{r,v,p,com}} \times RTCS\_TSFR_{r,t,p,com,sl,ts} \end{array} \right)$$

$$\{=;\leq;\geq\}$$

$$\sum_{\mathbf{rtpcs\_varf_{r,t,p,c,ts}}} VAR\_FLO_{r,v,t,p,c,ts} \times RTCS\_TSFR_{r,t,p,c,s,ts} +$$

$$\sum_{com \in \left\{ \begin{array}{l} \mathbf{rpc\_pg_{r,p,com}} \\ |c='ACT' \end{array} \right\}} \sum_{\mathbf{prc\_ts_{r,p,ts}}} \frac{VAR\_FLO_{r,v,t,p,com,ts}}{PRC\_ACTFLO_{r,v,p,com}} \times RTCS\_TSFR_{r,t,p,com,s,ts}$$

Case B: Storage processes:

$$\sum_{\mathbf{rps\_s2_{r,p,sl}}} FLO\_ASHAR_{r,v,p,c,'ACT',sl,bd} \times RS\_FR_{r,s,sl} \times$$

$$\left(\sum_{com \in \left\{ \begin{array}{l} \mathbf{rpc\_pg_{r,p,com}} \\ |cg='ACT' \end{array} \right\}} \sum_{\mathbf{prc\_ts_{r,p,ts}}} VAR\_FLO_{r,v,t,p,com,ts} \times RTCS\_TSFR_{r,t,p,com,sl,ts} \right)$$

$$\{=;\leq;\geq\}$$

$$\sum_{\substack{\mathbf{rpc\_stg_{r,p,c}} \\ \mathbf{rpcs\_var_{r,p,c,ts}}}} \frac{VAR\_SOUT_{r,v,t,p,c,ts}}{PRC\_ACTFLO_{r,v,p,c}} \times RTCS\_TSFR_{r,t,p,c,s,ts}$$

6.3.12 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:

$$EQ(l)\_BLND_{r,t,ble,spe} \qquad \ni bl\_type_{r,ble,spe}$$

$$\sum_{opr \in ble\_opr_{r,ble,opr}} BL\_COM_{r,ble,opr,spe} \cdot RU\_CVT_{r,ble,spe,opr} \cdot VAR\_BLND_{r,t,ble,opr}$$

$$\{\leq;=;\geq\}$$

$$\sum_{opr \in ble\_opr_{r,ble,opr}} BL\_SPEC_{r,ble,opr,spe} \cdot RU\_CVT_{r,ble,spe,opr} \cdot VAR\_BLND_{r,t,ble,opr}$$

6.3.13 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
- **cost_map$_{\textbf{agg,costagg}}$** = 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:

$$EQ\_BNDCST_{r,y1,t,y2,agg,cur} \quad \ni \left(\left(\sum_{bd} REG\_CUMCST_{r,y1,y2,agg,cur,bd} <> 0\right) \wedge \left(t = \underset{\{tt|M(tt-1)<y2\}}{\arg\max}\,(M(tt))\right)\right)$$

$$\begin{aligned}
&VAR\_CUMCST_{r,y1,y2,agg,cur} = \\
&\qquad \sum_{y1\le y\le y2} INVCOST(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'INV',type}}\right) \\
&\qquad \sum_{y1\le y\le y2} INVTAX(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'INVTAX'}}\right) \\
&\qquad \sum_{y1\le y\le y2} INVSUB(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'INVSUB'}}\right) \\
&\qquad \sum_{y1\le y\le y2} FIXCOST(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'FOM'}}\right) \\
&\qquad \sum_{y1\le y\le y2} FIXTAX(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'FOMTAX'}}\right) \\
&\qquad \sum_{y1\le y\le y2} FIXSUB(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'FOMSUB'}}\right) \\
&\qquad \sum_{y1\le y\le y2} COMTAX(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'COMTAX'}}\right) \\
&\qquad \sum_{y1\le y\le y2} COMSUB(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'COMSUB'}}\right) \\
&\qquad \sum_{y1\le y\le y2} FLOTAX(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'FLOTAX'}}\right) \\
&\qquad \sum_{y1\le y\le y2} FLOSUB(r,y) \quad \times\left(\mathbf{cost\_map}_{\mathbf{agg,'FLOSUB'}}\right)
\end{aligned}$$

6.3.14 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

$$EQ(l)\_BND(NET/PRD)_{r,t,c,s}$$

$$\ni \left\{\mathbf{rcs\_comts_{r,c,s}} \wedge (NOT\ \mathbf{com\_ts_{r,c,s}}) \wedge COM\_BND(NET/PRD)_{r,t,c,s,bd}\right\}$$

$$\sum_{ts \in \mathbf{rtcs\_varc_{r,t,c,ts}} \cap \mathbf{ts\_map_{r,s,ts}}} VAR\_COM(NET/PRD)_{r,t,c,ts}$$

$$(\leq / \geq / =)$$

$$COM\_BND(NET/PRD)_{r,t,c,s,bd}$$

Sign according to the *l* equation index (must coincide with the **bd** index in parameter COM_BNDNET/PRD).

6.3.15 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, $\mathbf{prc\_map_{r,'DISTR',p}}$ 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:

$$SX_{r,v,p,c,s} = \left\{ \begin{array}{ll} 0 & if\ \mathbf{top_{OUT}}(c) \wedge \mathbf{top_{IN}}(c) \wedge \neg NCAP\_AFCS_{r,v,p,cg,s} \\ \left(\dfrac{NCAP\_AFCS_{r,v,p,c,s}}{\sum\limits_{\mathbf{rp\_pg}_{r,p,cg}} NCAP\_AFCS_{r,v,p,cg,s}} \right) & \begin{array}{c} if\ \mathbf{top_{OUT}}(c) \wedge \mathbf{top_{IN}}(c) \wedge \\ NCAP\_AFCS_{r,v,p,c,s} \wedge \\ NCAP\_AFCS_{r,v,p,cg,s} \end{array} \\ 1 & otherwise \end{array} \right\}$$

- $IX_{r,v,p,c,s}$ denotes an adjustment coefficient for trade process exports:

$$IX_{r,v,p,c,s} = \begin{Bmatrix} \left(\dfrac{NCAP\_AFCS_{r,v,p,c,s}}{\sum\limits_{\mathbf{rp\_pg}_{\mathbf{r,p,cg}}} NCAP\_AFCS_{r,v,p,cg,s}} \right) & \begin{matrix} if\ \mathbf{imp}(c) \wedge \mathbf{exp}(c) \wedge \\ NCAP\_AFCS_{r,v,p,c,s} \wedge \\ NCAP\_AFCS_{r,v,p,cg,s} \end{matrix} \\ 1 & otherwise \end{Bmatrix}$$

Equation:

$$EQ(l)\_CAFLAC_{r,v,t,p,s} \quad \ni \begin{pmatrix} \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \wedge NCAP\_AFCS_{r,v,p,s} \wedge \\ \left(NCAP\_AF_{r,v,p,s,bd} \vee NCAP\_AFS_{r,v,p,s,bd} \vee NCAP\_AFA_{r,t,p,bd}\right) \end{pmatrix}$$

$$\sum_{\substack{\mathbf{rpc\_pg}_{\mathbf{r,p,c}} \\ \mathbf{prc\_ts}_{\mathbf{r,p,ts}}}} \begin{pmatrix} \dfrac{1}{PRC\_ACTFLO_{r,v,p,c}} \times \\ \begin{pmatrix} 1/NCAP\_AFCS_{r,v,p,c,s} & if\ NCAP\_AFCS_{r,v,p,c,s}\ given \\ \sum\limits_{\mathbf{rp\_pg}_{\mathbf{r,p,cg}}} \dfrac{1}{NCAP\_AFCS_{r,v,p,cg,s}} & elseif\ NCAP\_AFCS_{r,v,p,cg,s}\ given \\ 1 & otherwise \end{pmatrix} \times \\ \begin{pmatrix} VAR\_FLO_{r,v,t,p,c,ts} & if\ \mathbf{rp\_std}_{\mathbf{r,p}} \\ \begin{pmatrix} VAR\_SOUT_{r,v,t,p,c,ts} + \\ VAR\_SIN_{r,v,t,p,c,ts} \times SX_{r,v,p,c,ts} \end{pmatrix} & if\ \mathbf{rp\_stg}_{\mathbf{r,p}} \\ \begin{pmatrix} VAR\_IRE_{r,v,t,p,c,ts,'IMP'} + \\ VAR\_IRE_{r,v,t,p,c,ts,'EXP'} \times IX_{r,v,p,c,ts} \end{pmatrix} & if\ \mathbf{rp\_ire}_{\mathbf{r,p}} \end{pmatrix} \\ \times RTCS\_TSFR_{r,t,c,s,ts} \end{pmatrix} +$$

$$\{\leq,=\}$$

$$\sum_{\{v2|\mathbf{rtp\_cptyr}_{\mathbf{r,v2,t,p}}\}} \begin{pmatrix} COEF\_AF_{r,v2,t,p,s} \times COEF\_CPT_{r,v2,t,p} \times \\ \begin{pmatrix} VAR\_NCAP_{r,v2,p} + NCAP\_PASTI_{r,v2,p} - \\ \left(VAR\_SCAP_{r,v2,t,p} \quad if\ PRC\_RCAP_{r,p}\right) \end{pmatrix} \\ \times PRC\_CAPACT_{r,p} \end{pmatrix} \quad if\ \neg\mathbf{prc\_vint}_{r,p}$$

$$+ \begin{pmatrix} COEF\_AF_{r,v,t,p,s} \times COEF\_CPT_{r,v,t,p} \times \\ \begin{pmatrix} VAR\_NCAP_{r,v,p} + NCAP\_PASTI_{r,v,p} - \\ \left(VAR\_SCAP_{r,v,t,p} \quad if\ PRC\_RCAP_{r,p}\right) \end{pmatrix} \\ \times PRC\_CAPACT_{r,p} \end{pmatrix} \quad if\ \mathbf{prc\_vint}_{r,p}$$

6.3.16 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 $\leq$.
- *l* = 'G' for **bd** = 'LO' (lower bound) yields $\geq$.

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
  $B(v) \geq B(t) - (COEF\_RPTI*TLIFE) - ILED + 1$
  and
  $B(v) \leq E(t) - ILED$
- **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:

$$EQ(l)\_CAPACT_{r,v,t,p,s} \ni \mathbf{rtp\_vintyr_{r,v,t,p}} \wedge \mathbf{prc\_ts_{r,p,s}} \wedge \mathbf{rtp\_vara_{r,t,p}} \wedge \left(NCAP\_AF_{r,t,p,s} \vee NCAP\_AFS_{r,t,p,s} \vee NCAP\_AFA_{r,t,p}\right)$$

$$\left\{\begin{array}{ll} \sum\limits_{ts \in \left(\mathbf{prc\_ts_{r,p,ts}} \cap \mathbf{ts\_map_{r,s,ts}}\right)} VAR\_ACT_{r,v,t,p,ts} & if\ \neg \mathbf{prc\_map_{r,'STG'p}} \\ \sum\limits_{ts \in \left(\mathbf{prc\_ts_{r,p,ts}}\right)} \dfrac{VAR\_ACT_{r,v,t,p,ts}}{RS\_STGPRD_{r,ts}} \times RS\_FR_{r,ts,s} & if\ \mathbf{prc\_map_{r,'STG'p}} \end{array}\right\}$$

$$\{\leq;=;\geq\}$$

$$\text{Case1: Non - vintaged process (v = t):}$$

$$\sum_{\{v2|\mathbf{rtp\_cptyr_{r,v2,t,p}}\}} \begin{pmatrix} COEF\_AF_{r,v2,t,p,s,bd} \times COEF\_CPT_{r,v2,t,p} \times \\ \begin{pmatrix} VAR\_NCAP_{r,v2,p} + NCAP\_PASTI_{r,v2,p} - \\ \left(VAR\_SCAP_{r,v2,t,p} \quad if\ PRC\_RCAP_{r,p}\right) \end{pmatrix} \\ \times PRC\_CAPACT_{r,p} \end{pmatrix} \quad if\ \neg \mathbf{prc\_vint}_{r,p}$$

$$\text{Case2: Vintaged process (v = vintage):}$$

$$+ \quad \begin{pmatrix} COEF\_AF_{r,v,t,p,s,bd} \times COEF\_CPT_{r,v,t,p} \times \\ \begin{pmatrix} VAR\_NCAP_{r,v,p} + NCAP\_PASTI_{r,v,p} - \\ \left(VAR\_SCAP_{r,v,t,p} \quad if\ PRC\_RCAP_{r,p}\right) \end{pmatrix} \\ \times PRC\_CAPACT_{r,p} \end{pmatrix} \quad if\ \mathbf{prc\_vint}_{r,p}$$

$$\times \left[G\_YRFR_{r,s} \times (p \notin \mathbf{prc\_map_{r,'STG',p}}) + 1 \times (p \in \mathbf{prc\_map_{r,'STG',p}})\right]$$

$COEF\_CPT_{r,v,t,p}$:

$if \;\; v = t$

$$= Max\begin{pmatrix} \dfrac{D(t) - NCAP\_ILED}{D(t)} \\ 0 \end{pmatrix}$$

If **v** *has been a long time period, and* **t** *is close enough to encounter a capacity created at the end of* **v**.

$else$

$if \;\; t \geq v \wedge D(v) > IL + TL \wedge B(t) < E(v) + TL$

$$= Max\begin{pmatrix} \dfrac{Min\left(B(v) + IL + COEF\_RPTI_{r,v,p} \times TL, E(t) + 1\right) - B(t)}{D(t)} \\ 0 \end{pmatrix}$$

Number of years of existence within period **t**, *divided by the period duration*

$else$

$$= Max\begin{pmatrix} \dfrac{Min\left(B(v) + IL + TL, E(t) + 1\right) - Max\left(B(v) + IL, B(t)\right)}{D(t)} \\ 0 \end{pmatrix}$$

$endif$

$endif$

This step blocks out the investments that have already retired, which may be evaluated with a negative remaining life

Where,

Simply counts the number of investments in a long time period.

$$COEF\_RPTI_{r,v,p} = \left\langle \frac{D(v) - IL}{TL} \right\rangle$$

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) = 1st year of the period containing **t**
E(t) = Last year of the period containing **t**
D(t) = Duration of the period containing **t**

6.3.17 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):

$$EQ\_CAPFLO_{r,v,t,p,c,s} \ni \left(\mathbf{rtp\_vintyr_{r,v,t,p}} \wedge NCAP\_AFC_{r,v,p,c,s} \wedge \neg \mathbf{rpc\_pg_{r,p,c}}\right)$$

$$\left(\begin{array}{ll} \sum\limits_{\mathbf{rtpcs\_varf_{r,t,p,c,ts}}} VAR\_FLO_{r,v,t,p,c,s} & if\ c \neq \text{'ACT'} \\ \sum\limits_{\mathbf{prc\_ts_{r,p,ts}}} \dfrac{VAR\_ACT_{r,v,t,p,ts} \times RS\_FR_{r,ts,s} \times G\_YRFR_{r,s}}{G\_YRFR_{r,P(ts)}} & if\ c = \text{'ACT'} \end{array}\right)$$

$$\leq$$

$$NCAP\_AFC_{r,v,t,p,c,s} \times \left(\begin{array}{l} VAR\_NCAP_{r,v,p} + NCAP\_PASTI_{r,v,p} - \\ \sum\limits_{\mathbf{prc\_rcap_{r,p}}} VAR\_SCAP_{r,v,t,p} - \sum\limits_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \end{array}\right) \cdot$$

$$COEF\_CPT_{r,v,p,t} \times PRC\_CAPACT_{r,p} \times G\_YRFR_{r,s}$$

6.3.18 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:

$$EQ\_CAPLOAD_{r,v,t,p,s,UP} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,s,'FX'} > 0))$$

$$VAR\_ACT_{r,v,t,p,s} \leq$$

$$AF\_MAX_{r,v,t,p,s} \times \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} - \\ \sum_{\mathbf{prc\_rcap_{r,p}}} VAR\_SCAP_{r,v,t,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \end{array} \right) \cdot$$

$$COEF\_CPT_{r,v,p,t} \cdot PRC\_CAPACT_{r,p} \cdot G\_YRFR_{r,s}$$

$$EQ\_CAPLOAD_{r,v,t,p,s,LO} \quad \ni (\mathbf{rtp\_vintyr}_{r,v,t,p} \wedge \mathbf{prc\_ts}_{r,p,s} \wedge (ACT\_UPS_{r,v,p,s,'FX'} > 0))$$

$$VAR\_ACT_{r,v,t,p,s} \geq$$

$$AF\_MIN_{r,v,p,s} \times \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} - NCAP\_PASTI_{r,v,p} - \\ \sum_{\mathbf{prc\_rcap_{r,p}}} VAR\_SCAP_{r,v,t,p} - \sum_{ts \in SUP(s) \cap UPS(p)} VAR\_UPS_{r,v,t,p,ts,'N'} \end{array} \right) \cdot$$

$$COEF\_CPT_{r,v,p,t} \cdot PRC\_CAPACT_{r,p} \cdot G\_YRFR_{r,s}$$

6.3.19 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:

$$EQ(l)\_CPT_{r,t,p} \ni CAP\_BND_{r,t,p,bd} \vee \mathbf{teg_p} \vee \mathbf{rtp\_varp_{r,t,p}}$$

$$\begin{aligned}
&VAR\_CAP_{r,t,p} \times \left(\mathbf{rtp\_varp_{r,t,p}} \vee CAP\_BND_{r,t,p,'FX'} \vee \mathbf{teg_p}\right) \\
&+ CAP\_BND_{r,t,p,'LO'} \times \left[\left(NOT\ \mathbf{rtp\_varp_{r,t,p}}\right) \wedge CAP\_BND_{r,t,p,'LO'}\right] \\
&+ CAP\_BND_{r,t,p,'UP'} \times \left[\left(NOT\ \mathbf{rtp\_varp_{r,t,p}}\right) \wedge CAP\_BND_{r,t,p,'UP'}\right]
\end{aligned}$$

$$\{\leq; =; \geq\}$$

$$\sum_{v \in \mathbf{rtp\_cptyr_{r,v,t,p}}} COEF\_CPT_{r,v,t,p} \times \begin{pmatrix} VAR\_NCAP_{r,v,p} \times (v \in MILESTONYR) \\ + NCAP\_PASTI_{r,v,p} \times (v \in PASTYEAR) \\ - VAR\_SCAP_{r,v,t,p} \times ((r,p) \in \mathbf{prc\_rcap}) \end{pmatrix}$$

where

$COEF\_CPT_{r,v,t,p}$ *as defined in equation* $EQ(l)\_CAPACT$

6.3.20 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 ≥ 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 {$\geq$ 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:

*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**.*

$$EQ(l)\_COMBAL_{r,t,c,s} \ni \left[\mathbf{rcs\_combal_{r,t,c,s,bd}}\right]$$

$$COM\_IE_{r,t,c} \times \left(\sum_{p \in \mathbf{top_{r,p,c,'OUT'}}} \sum_{v \in \mathbf{rtp\_vintyr_{r,v,t,p}}} CAL\_FLOFLO_{r,v,t,p,c,s,'OUT'} \right.$$

Output flow of ordinary processes

$$+ \sum_{p \in \mathbf{rpc\_ire_{r,p,c,'IMP'}}} \sum_{v \in \mathbf{rtp\_vintyr_{r,v,t,p}}} CAL\_IRE_{r,v,t,p,c,s,'IMP'} + AUX\_IRE_{r,t,c,s,'OUT'}$$

Import of the commodity

$$+ \sum_{\substack{p \in \mathbf{rpc\_stg_{r,p,c}} \\ (p,v) \in \mathbf{rtp\_vintyr_{r,v,t,p}}}} \sum_{s1 \in \mathbf{prc\_ts_{r,p,ts}}} \left(VAR\_SOUT_{r,v,t,p,c,ts} \times \left(RS\_FR_{r,s,ts} \right) \times STG\_EFF_{r,v,p} \right)$$

Storage output

$$+ \sum_{opr \in \mathbf{ble\_opr_{r,c,opr}}} \left(BLE\_BAL_{r,t,c,opr} \times VAR\_BLND_{r,t,c,opr} \times RTCS\_TSFR_{r,t,c,s,'ANNUAL'} \right)$$

Output of blending process; the parameter BLE_BAL converts the blending streams to energy units

$$+ \left[\sum_{\substack{(p,v) \in \mathbf{rpc\_capflo_{r,v,p,c}} \\ if\,(\mathbf{rtp\_cptyr_{r,v,t,p}} \\ \wedge \\ NCAP\_COM_{r,v,p,c,'OUT'})}} \left(\begin{matrix} NCAP\_COM_{r,v,p,c,'OUT'} \times COEF\_CPT_{r,v,t,p} \times \\ \left(\begin{matrix} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \\ VAR\_SCAP_{r,tt(v),t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{matrix} \right) \end{matrix} \right) \right] \times G\_YRFR_{r,s}$$

Flow produced by Technology Capacity

$$+ \left[\sum_{\substack{(p,v) \in \mathbf{rpc\_capflo_{r,v,p,c}} \\ if\ COEF\_OCOM_{r,v,t,p,c}}} \left(\begin{matrix} COEF\_OCOM_{r,v,t,p,c} \times \\ \left(\begin{matrix} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \\ VAR\_SCAP_{r,tt(v),t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{matrix} \right) \end{matrix} \right) \right] \times G\_YRFR_{r,s}$$

Flow produced by Technology Investment/Dismantling

$$\left. + \sum_{\substack{(com,ts) \in \\ \mathbf{com\_ts_{r,com,ts}}}} COM\_AGG_{r,t,com,c} \times \left(\begin{matrix} VAR\_COMNET_{r,t,com,ts} \\ \text{or, if } \mathbf{com\_lim_{com}} = \text{FX/N:} \\ VAR\_COMPRD_{r,t,com,ts} \end{matrix} \right) \times RTCS\_TSFR_{r,t,com,s,ts} \right)$$

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*)

$$
-\left(\begin{array}{l}
\sum_{p\in\mathbf{top}_{\mathbf{r,p,c,'OUT'}}} \sum_{v\in\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_FLOFLO_{r,v,t,p,c,s,'IN'} \\
+ \sum_{p\in\mathbf{rpc\_ire}_{\mathbf{r,p,c,'EXP'}}} \sum_{v\in\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_IRE_{r,v,t,p,c,s,'EXP'} + AUX\_IRE_{r,t,c,s,'IN'} \\
+ \sum_{\substack{p\in\mathbf{rpc\_stg}_{\mathbf{r,p,c}} \\ (p,v)\in\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}}} \sum_{ts\in\mathbf{prc\_ts}_{\mathbf{r,p,ts}}} \left(VAR\_SIN_{r,v,t,p,c,ts} \times \left(RS\_FR_{r,s,ts} \right)\right) \\
+ \sum_{ble\in\mathbf{ble\_opr}_{\mathbf{r,ble,c}}} \left(BLE\_BAL_{r,t,ble,c} \times VAR\_BLND_{r,t,ble,c} \times RTCS\_TSFR_{r,t,c,s,'ANNUAL'} \right) \\
+ \left[\sum_{\substack{(p,v)\in\mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}} \\ if\ (\mathbf{rtp\_cptyr}_{\mathbf{r,v,t,p}} \\ \wedge \\ NCAP\_COM_{r,v,p,c,'IN'})}} \left(\begin{array}{l} NCAP\_COM_{r,v,p,c,'IN'} \times COEF\_CPT_{r,v,t,p} \times \\ \left(VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \right) \\ VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{array} \right) \right] \times G\_YRFR_{r,s} \\
+ \left[\sum_{\substack{(p,v)\in\mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}} \\ if\ COEF\_ICOM_{r,v,t,p,c}}} \left(\begin{array}{l} COEF\_ICOM_{r,v,t,p,c} \times \\ \left(VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \right) \\ VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{array} \right) \right] \times G\_YRFR_{r,s}
\end{array}\right)
$$

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)

$$CAL\_FLOFLO_{r,v,t,p,c,s,io} \ni \mathbf{rp\_flo_{r,p}} \wedge NOT\ \mathbf{rpc\_conly_{r,t,p,c}}$$

The process has regular flow variables (VAR_FLO).

RPC_CONLY contains commodities ONLY involved in NCAP_I/O/COM

$$= \sum_{s1 \in \mathbf{rtpcs\_varf_{r,t,p,c,s1}}} VAR\_FLO_{r,v,t,p,c,s1} \times RTCS\_TSFR_{r,t,c,s,s1}$$

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 $\mathbf{ts\_map_{r,s,s1}}$

$$=1$$

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)**.

$$CAL\_IRE_{r,v,t,p,c,s,ie} \ni \mathbf{rpc\_ire_{r,p,c,ie}} \wedge NOT\ \mathbf{rpc\_conly_{r,t,p,c}}$$

$$= \sum_{s1 \in \mathbf{rtpcs\_varf_{r,t,p,c,s1}}} VAR\_IRE_{r,v,t,p,c,s1,ie} \times RTCS\_TSFR_{r,t,c,s,s1}$$

Adjusts the time-slice of IRE for COM_BAL

Computes the Auxiliary flows associated with an inter-regional process

$$AUX\_IRE_{r,t,c,s,io}$$

$$=$$

$$= \sum_{(p,com,ie) \in \left(\begin{array}{l}\mathbf{rpc\_ire_{r,p,com,ie}} \\ \wedge IRE\_FLOSUM_{r,t,p,com,s,ie,c,io}\end{array}\right)} \sum_{v \in \mathbf{rtp\_vintyr_{r,v,t,p}}} \sum_{s1 \in \mathbf{rtpcs\_varf_{r,t,p,com,s1}}} \left(\begin{array}{l} IRE\_FLOSUM_{r,t,p,com,s1,ie,c,io} \times VAR\_IRE_{r,v,t,p,c,s1,ie} \\ if\ \mathbf{ts\_map_{r,s,s1}} \\ \times 1 \\ else \\ \times \dfrac{G\_YRFR_{r,s}}{G\_YRFR_{r,s1}} \end{array} \right)$$

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

$$where \ \ COEF\_RPTINV = \left\langle \frac{D(t) - ILED_t}{TLIFE_t} \right\rangle$$

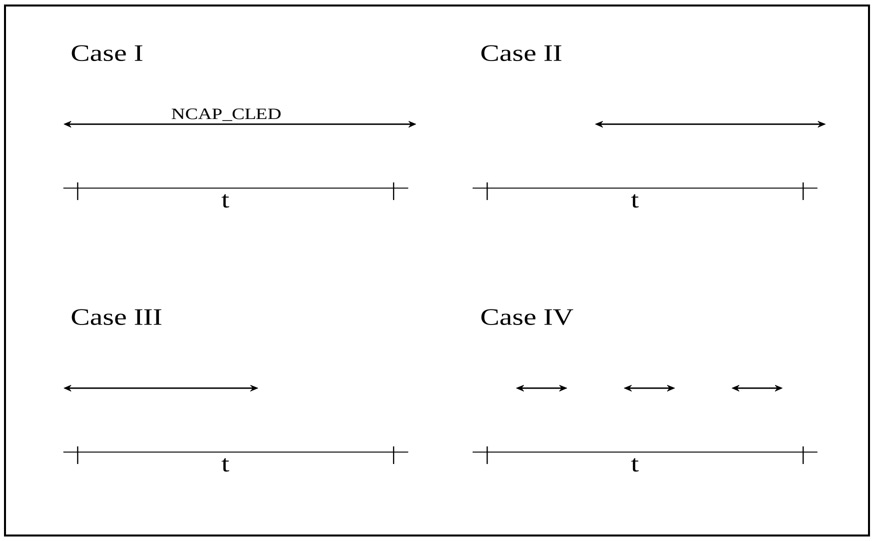
Counts the number of investments in a long period

$else$

$$= Max\left(\frac{1 + Min(ECF, E(t)) - Max(BCF, B(t))}{D(t)} \times \frac{NCAP\_ICOM_v}{NCAP\_CLED_v}, 0 \right)$$

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'.

$COEF\_OCOM$:

$if\ \ t \geq v \wedge D(v) > IL + TL \wedge B(t) < E(v) + TL + DLAG + DLIFE$

$$= \sum_{i=1}^{COEF\_RPTINV} \left(Max \left(\begin{matrix} \left(\dfrac{Min\big(B(v) + IL + (i \times TL) + DLAG + DLIFE - 1, E(t)\big)}{D(t)} - \dfrac{Max\big(B(v) + IL + (i \times TL) + DLAG, B(t)\big)}{D(t)} \right) \\ 0 \end{matrix} \right) \right) \times \frac{NCAP\_OCOM_v}{NCAP\_DLIFE_v}$$

else

$$= Max \left(\begin{matrix} \dfrac{1 + Min(ECF, E(t)) - Max(BCF, B(t))}{D(t)} \times \dfrac{NCAP\_OCOM_v}{NCAP\_DLIFE_v} \\ 0 \end{matrix} \right)$$

endif

6.3.21 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

6.3.22 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:

$$EQ\_CUMFLO_{r,p,c,y1,t,y2} \quad \ni \left(\mathbf{rpc_{r,p,c}} \wedge \mathbf{rpc\_cumflo_{r,p,c,y1,y2}}\right)$$

$if\ c \neq$ 'ACT':

$$\sum_{t=T(y1)}^{t=T(y2)} \sum_{\substack{s\in\mathbf{rtpcs\_varf_{r,t,p,c,s}} \\ \mathbf{v}\in\mathbf{rtp\_vintyr_{r,v,t,p}}}} \left[Min\{E(t),y2\} - Max\{B(t),y1\} + 1\right] \times VAR\_FLO_{r,v,t,p,c,s} = VAR\_CUMFLO_{r,p,c,y1,y2}$$

$if\ c =$ '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:

$VAR\_CUMFLO.LO_{r,p,'ACT',y1,y2} = ACT\_CUM_{r,y1,y2,'LO'}$

$VAR\_CUMFLO.UP_{r,p,'ACT',y1,y2} = ACT\_CUM_{r,y1,y2,'UP'}$

$VAR\_CUMFLO.FX_{r,p,'ACT',y1,y2} = ACT\_CUM_{r,y1,y2,'FX'}$

$VAR\_CUMFLO.LO_{r,p,c,y1,y2} = FLO\_CUM_{r,p,c,y1,y2,'LO'}$

$VAR\_CUMFLO.UP_{r,p,c,y1,y2} = FLO\_CUM_{r,p,c,y1,y2,'UP'}$

$VAR\_CUMFLO.FX_{r,p,c,y1,y2} = FLO\_CUM_{r,p,c,y1,y2,'FX'}$

6.3.23 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:

$$EQ(l)\_CUMNET_{r,y1,y2,c} \ni COM\_CUMNET_{r,y1,y2,c,l}$$

$$\sum_{t=T(y1)}^{t=T(y2)} \sum_{s \in \mathbf{rtcs\_varc}_{r,t,c,s}} \left[Min\{E(t),y2\} - Max\{B(t),y1\} + 1\right] \times VAR\_COMNET_{r,t,c,s} = VAR\_CUMCOM_{r,c,'NET',y1,y2}$$

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**.

$$EQ(l)\_CUMPRD_{r,y1,y2,c,s} \ni COM\_CUMPRD_{r,y1,y2,c,l}$$

$$\sum_{t=T(y1)}^{t=T(y2)} \sum_{s \in \mathbf{rtcs\_varc}_{r,t,c,s}} \left[Min\{E(t),y2\} - Max\{B(t),y1\} + 1\right] \times VAR\_COMPRD_{r,t,c,s} = VAR\_CUMCOM_{r,c,'PRD',y1,y2}$$

Bounds:

$VAR\_CUMCOM.LO_{r,c,'NET',y1,y2} = COM\_CUMNET_{r,y1,y2,c,'LO'}$

$VAR\_CUMCOM.UP_{r,c,'NET',y1,y2} = COM\_CUMNET_{r,y1,y2,c,'UP'}$

$VAR\_CUMCOM.FX_{r,c,'NET',y1,y2} = COM\_CUMNET_{r,y1,y2,c,'FX'}$

$VAR\_CUMCOM.LO_{r,c,'PRD',y1,y2} = COM\_CUMPRD_{r,y1,y2,c,'LO'}$

$VAR\_CUMCOM.UP_{r,c,'PRD',y1,y2} = COM\_CUMPRD_{r,y1,y2,c,'UP'}$

$VAR\_CUMCOM.FX_{r,c,'PRD',y1,y2} = COM\_CUMPRD_{r,y1,y2,c,'FX'}$

6.3.24 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:

$$EQ\_CUMRET_{r,v,t,p} \quad \ni (\mathbf{rtp\_cptyr}_{r,v,t,p} \wedge \mathbf{prc\_rcap}_{r,p})$$

$$VAR\_SCAP_{r,v,t,p} = VAR\_RCAP_{r,v,t,p} + \sum_{t-1 \in \{tt | \mathbf{rtp\_cptyr}_{r,v,tt,p}\}} VAR\_SCAP_{r,v,t-1,p}$$

6.3.25 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:

$$EQ\_DSCNCAP_{r,t,p} \quad \ni \left(\mathbf{rp\_dscncap_{r,p}} \wedge \mathbf{rtp_{r,t,p}}\right)$$

$$VAR\_NCAP_{r,t,p} = \sum_{u \in \mathbf{unit}} \left(VAR\_DNCAP_{r,t,p,u} \times NCAP\_DSC_{r,t,p,u}\right) + \left(VAR\_SNCAP_{r,t,p} \quad if \;\; NCAP\_SEMI_{r,t,p} \; given\right)$$

6.3.26 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

$$EQ\_DSCONE_{r,t,p} \quad \ni \left(\mathbf{rp\_dscncap_{r,p}} \wedge \mathbf{rtp_{r,t,p}}\right)$$

$$\sum_{u \in \mathbf{unit}} VAR\_DNCAP_{r,t,p,u} = 1$$

Note that VAR_DNCAP must be declared as a binary variable (taking values 0 or 1 only)

6.3.27 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* × *RCAP_BLK, n=0,1,2,3,…*.

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:

$$EQ\_DSCRET_{r,v,t,p} \quad \ni (\mathbf{rtp\_cptyr}_{r,v,t,p} \wedge RCAP\_BLK_{r,v,p})$$

$$\begin{aligned} & VAR\_SCAP_{r,v,t,p} - RTFORC_{r,v,t,p} = \\ & \qquad RCAP\_BLK_{r,v,p} \times VAR\_DRCAP_{r,v,t,p,2} + \\ & \qquad (NCAP\_PASTI_{r,v,p} - RTFORC_{r,v,t,p}) \times VAR\_DRCAP_{r,v,t,p,1} \end{aligned}$$

6.3.28 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.

$$EQ(l)\_FLOBND_{r,t,p,cg,s} \ni \left\{ \begin{array}{l} \mathbf{rtp}_{\mathbf{r,t,p}} \wedge FLO\_BND_{r,t,p,cg,s,bd} \wedge \\ \left(\mathbf{prc\_vint}_{\mathbf{r,p}} \vee \sum\limits_{c \in \mathbf{com\_gmap}_{\mathbf{r,cg,c}}} \sum\limits_{ts \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \mathbf{rs\_below}_{\mathbf{r,s,ts}} \vee \neg \mathbf{com}_{r,cg} \right) \end{array} \right\}$$

$$\sum_{c \in \mathbf{com\_gmap}_{\mathbf{r,cg,c}}} \sum_{ts \in \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,c,ts}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \begin{pmatrix} VAR\_FLO_{r,v,t,p,c,ts} & if\ \mathbf{rp\_flo}_{\mathbf{r,p}} \\ \sum\limits_{ie} VAR\_IRE_{r,v,t,p,c,ts,ie} \times XS_{cg,ie} & if\ \mathbf{rp\_ire}_{\mathbf{r,p}} \end{pmatrix}$$

$$(\le / \ge / =) \qquad FLO\_BND_{r,t,p,cg,s,bd}$$

where the equation sign is indicated by equation index **l** based on the bound type **bd**.

6.3.29 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

$$EQ(l)\_FLOFR_{r,t,p,c,s} \quad \ni \left\{ \sum_{ts \in \mathbf{rpcs\_var}_{r,p,c,ts}} \mathbf{ts\_map}_{r,s,ts} \wedge FLO\_FR_{r,t,p,c,s,bd} \right\}$$

*The timeslices of the process flow (**ts**) have to be below the timeslice (**s**) of the bound.*

$$\sum_{ts \in \mathbf{rtpcs\_varf}_{r,t,p,c,ts}} \sum_{v \in \mathbf{rtp\_vintyr}_{r,v,t,p}} \left(VAR\_FLO_{r,v,t,p,c,ts} \times RTCS\_TSFR_{r,t,c,s,ts} \right)$$

$$(\leq / \geq / =)$$

$$\sum_{ts \in \mathbf{rtpcs\_varf}_{r,t,p,c,ts}} \sum_{v \in \mathbf{rtp\_vintyr}_{r,v,t,p}} \left[VAR\_FLO_{r,v,t,p,c,ts} \times FLO\_FR_{r,t,p,c,s,bd} \right]$$

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

$$EQ(l)\_FLOFR_{r,t,p,c,s} \quad \ni \left\{ \mathbf{rtpcs\_var_{r,t,p,c,s}} \wedge FLO\_FR_{r,t,p,c,s,bd} \right\}$$

$$\sum_{v \in \mathbf{rtp\_vintyr_{r,v,t,p}}} \left(\frac{VAR\_FLO_{r,v,t,p,c,s}}{G\_YRFR_{r,s}} \right)$$

$$(\leq / \geq / =)$$

$$\sum_{ts \in \mathbf{rs\_below1_{r,ts,s}}} \sum_{v \in \mathbf{rtp\_vintyr_{r,v,t,p}}} \left(\frac{\sum\limits_{sl \in \mathbf{rs\_below1_{r,ts,sl}}} VAR\_FLO_{r,v,t,p,c,sl}}{G\_YRFR_{r,ts}} \right) \times FLO\_FR_{r,t,p,c,s,bd}$$

where the equation sign is indicated by equation index **l**.

6.3.30 Equation: EQ(*l*)_FLOMRK

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:

$$EQ(l)\_FLMRK_{r,t,grp,c,s} \forall (r,t,grp,c,s) \in \left(\left\{\mathbf{rtpc_{r,t,p,c}} \mid PRC\_MARK_{r,t,p,grp,c,s,l} \neq 0\right\} \cap \mathbf{mrk\_ts_{r,grp,c,s}}\right)$$

$$\sum_{\substack{(com,v,ts)\in \\ RPC_p \cap COM\_GMAP_c \\ \cap RTP\_VINTYR_{p,t} \\ \cap RPCS\_VAR_p}} \left\{ \left[VAR\_FLO_{r,v,t,p,com,ts} \times \begin{bmatrix} COM\_IE_{r,com,ts} & if\ output \\ 1 & if\ input \end{bmatrix} + \right.\right.$$

$$\begin{pmatrix} VAR\_IRE_{r,v,t,p,com,ts,imp} \\ VAR\_SOUT_{r,v,t,p,com,ts} \times STG\_EFF_{r,v,p} \end{pmatrix} \times \begin{bmatrix} COM\_IE_{r,com,ts} & if\ PRC\_MARK_{r,t,p,grp,c,s,l} \geq 0 \\ 0 & if\ PRC\_MARK_{r,t,p,grp,c,s,l} < 0 \end{bmatrix} -$$

$$\left.\left. \begin{pmatrix} VAR\_IRE_{r,v,t,p,com,ts,exp} \\ VAR\_SIN_{r,v,t,p,com,ts} \end{pmatrix} \times \begin{bmatrix} 1 & if\ PRC\_MARK_{r,t,p,grp,c,s,l} \leq 0 \\ 0 & if\ PRC\_MARK_{r,t,p,grp,c,s,l} > 0 \end{bmatrix} \right] \times \left(\frac{RS\_FR_{r,s,ts}}{PRC\_MARK_{r,t,p,grp,c,s,l}} \right) \right\}$$

$$\{=;\leq;\geq\}$$

$$\sum_{\substack{com \in \\ RPC \cap COM\_GMAP_c}} \sum_{\substack{ts \in \\ RHS\_COMPRD_{t,com}}} \left\{ VAR\_COMPRD_{r,t,com,ts} \times \left(RS\_FR_{r,s,ts} \right) \right\}$$

6.3.31 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).

6.3.31.1 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.

6.3.31.1.1 Case A. Bi-lateral or multilateral unidirectional trade to a single import region

Equation:

$$EQ\_IRE_{r,t,p,c,s} \ni \{r,t,p,c,s \in (\mathbf{rtp}_{\mathbf{r,t,p}} \wedge \mathbf{rpcs\_var}_{\mathbf{r,p,c,s}} \wedge \mathbf{rpc\_eqire}_{\mathbf{r,p,c}})\}:$$

This is the efficiency of process **p** *for the pair of regions and commodity* **c**.

$$\sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} VAR\_IRE_{r,v,t,p,c,s,'IMP'} =$$

$$\sum_{(r2,c2)\in \mathbf{top\_ire}_{\mathbf{r2,c2,r,c,p}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r2,v,t,p}}} \sum_{s2\in IRE\_TSCVT_{r2,s2,r,s}} \sum_{ts\in\left(\begin{smallmatrix}\mathbf{rtpcs\_varf}_{\mathbf{r2,t,p,c2,ts}}\\ \cap\, \mathbf{rs\_tree}_{\mathbf{r,s2,ts}}\end{smallmatrix}\right)} \left(\begin{array}{l} VAR\_IRE_{r2,v,t,p,c2,ts,'EXP'} \times IRE\_FLO_{r2,v,p,c2,r,c,s} \times \\ IRE\_TSCVT_{r2,s2,r,s} \times IRE\_CCVT_{r2,c2,r,c} \times \\ RTCS\_TSFR_{r2,t,c,s2,ts} \end{array}\right)$$

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.

6.3.31.1.2 Case B. Multidirectional and market-based trade between regions.

Equation:

$$EQ\_IRE_{r,t,p,c,s} \quad \ni \left\{ r,t,p,c,s \in (\mathbf{rtp}_{\mathbf{r,t,p}} \wedge \mathbf{rpcs\_var}_{\mathbf{r,p,c,s}} \wedge \mathbf{rpc\_eqire}_{\mathbf{r,p,c}}) \right\}:$$

$$\sum_{\substack{(r2,c1,c2)\in \\ (\mathbf{top\_ire}_{\mathbf{r,c1,r2,c2,p}} \cap \mathbf{top\_ire}_{\mathbf{r,c1,r,c,p}} \cap \mathbf{rpc\_market}_{\mathbf{r,p,c1}})}} \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r2,v,t,p}}} \sum_{s2\in IRE\_TSCVT_{r2,s2,r,s}} \sum_{ts\in \begin{pmatrix} \mathbf{rtpcs\_varf}_{\mathbf{r2,t,p,c2,ts}} \\ \cap\, \mathbf{rs\_tree}_{\mathbf{r,s2,ts}} \end{pmatrix}} \begin{pmatrix} VAR\_IRE_{r2,v,t,p,c2,s2,'IMP'} \times IRE\_CCVT_{r,c1,r,c} \\ \times IRE\_CCVT_{r2,c2,r,c1} \times IRE\_TSCVT_{r2,s2,r,s} \\ \times RTCS\_TSFR_{r2,t,c,s2,ts} \end{pmatrix} \times$$

$$=$$

$$\sum_{(r2,c2)\in \mathbf{top\_ire}_{\mathbf{r2,c2,r,c,p}}} \sum_{v\in \mathbf{rtp\_vintyr}_{\mathbf{r2,v,t,p}}} \sum_{s2\in IRE\_TSCVT_{r2,s2,r,s}} \sum_{ts\in \begin{pmatrix} \mathbf{rtpcs\_varf}_{\mathbf{r2,t,p,c2,ts}} \\ \cap\, \mathbf{rs\_tree}_{\mathbf{r,s2,ts}} \end{pmatrix}} \begin{pmatrix} VAR\_IRE_{r2,v,t,p,c2,ts,'EXP'} \times IRE\_FLO_{r2,v,p,c2,r,c,s} \times \\ IRE\_TSCVT_{r2,s2,r,s} \times IRE\_CCVT_{r2,c2,r,c} \times \\ RTCS\_TSFR_{r2,t,c,s2,ts} \end{pmatrix}$$

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.

6.3.31.2 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

$$EQ(l)\_IREBND_{r,t,c,s,all\_r,ie} \quad \forall \left\{ \begin{array}{l} r,t,c,s,all\_r,ie:(RCS\_COMTS_{r,c,s} \wedge \\ (\exists p: RPC\_IE_{r,p,c,ie}) \wedge IRE\_BND_{r,t,c,s,all\_r,ie}) \end{array} \right\}:$$

$$\sum_{p:(\exists c2:TOP\_IRE_{r,c,all\_r,c2,p})} \sum_{v \in RTP\_VINTYR_{r,v,t,p}} \sum_{s2} VAR\_IRE_{r,v,t,p,c,s2,exp} \times$$

$$\left(\begin{array}{ll} 1 & if\ s2 \in TS\_MAP(r,s,s2) \\ \dfrac{FR(s)}{FR(s2)} & if\ s2 \in RS\_BELOW(r,s2,s) \end{array} \right)$$

$$\{\leq;=;\geq\}\ IRE\_BND_{r,t,c,s,all\_r,ie}$$

Case D. Exports from a market region to an internal region

$$EQ(l)\_IREBND_{r,t,c,s,all\_r,ie} \quad \forall \left\{ \begin{array}{l} r,t,c,s,all\_r,ie:(RCS\_COMTS_{r,c,s} \wedge \\ (\exists p: RPC\_IE_{r,p,c,ie}) \wedge IRE\_BND_{r,t,c,s,all\_r,ie}) \end{array} \right\}:$$

$$\sum_{(c2,p) \in TOP\_IRE_{r,c,all\_r,c2,p}} \sum_{v \in RTP\_VINTYR_{all\_r,v,t,p}} \sum_{s2} VAR\_IRE_{all\_r,v,t,p,c2,s2,exp} \times$$

$$IRE\_CCVT_{all\_r,c2,r,c} \times IRE\_TSCVT_{all\_r,s2,r,s} \times \left(\begin{array}{ll} 1 & if\ s2 \in TS\_MAP(r,s,s2) \\ \dfrac{FR(s)}{FR(s2)} & if\ s2 \in RS\_BELOW(r,s2,s) \end{array} \right)$$

$$\{\leq;=;\geq\}\ IRE\_BND_{r,t,c,s,all\_r,ie}$$

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**.

6.3.31.3 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:

$$EQ(l)\_XBND_{all\_r,t,c,s,ie} \ni IRE\_XBND_{all\_r,t,c,s,ie,bd}$$

$$\boxed{all\_r \text{ is an internal}}$$

$$\sum_{p \in \mathbf{rpc\_ire}_{\mathbf{all\_r,p,c,ie}}} \sum_{s2 \in \left(\mathbf{rtpcs\_varf}_{\mathbf{all\_r,t,p,c,s2}} \cap \mathbf{rs\_tree}_{\mathbf{all\_r,s,s2}}\right)} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{all\_r,v,t,p}}} \left[\begin{array}{ll} VAR\_IRE_{all\_r,v,t,p,c,s2,ie} \times & \\ 1 & if\ s2 \in \mathbf{ts\_map}_{\mathbf{all\_r,s,s2}} \\ \dfrac{G\_YRFR(s)}{G\_YRFR(s2)} & if\ s2 \in \mathbf{rs\_below}_{\mathbf{all\_r,s2,s}} \end{array} \right]$$

$$\{=;\leq;\geq\}\ IRE\_XBND_{all\_r,t,c,s,ie,bd}$$

$$\boxed{all\_r \text{ is an external}}$$

$$\sum_{p \in \mathbf{rpc\_ire}_{\mathbf{r,p,com,impex}}} \sum_{(ts,s2) \in \left(\mathbf{rs\_tree}_{\mathbf{r,ts,s2}} \cap \mathbf{rtpcs\_varf}_{\mathbf{r,t,p,com,s2}} \cap IRE\_TSCVT_{r,ts,all\_r,s}\right)} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \left[\begin{array}{l} VAR\_IRE_{r,v,t,p,com,s2,impexp} \times IRE\_CCVT_{r,com,all\_r,c} \\ \times IRE\_TSCVT(r,ts,all\_r,s) \end{array} \right]$$

All regions r with impex ≠ ie
Having import/export from/to all_r

$$\{=;\leq;\geq\}\ IRE\_XBND_{all\_r,t,c,s,ie,bd}$$

6.3.32 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:

$$EQ(l)\_IN/OUTSHR_{r,v,t,p,c,cg,s} \quad \ni (c \in cg) \wedge \left(t \in \mathbf{rtp\_vintyr_{r,v,t,p}}\right) \wedge \mathbf{top_{r,p,c,'IN'/'OUT'}} \wedge$$
$$(s \in \mathbf{rps\_s1_{r,p,s}}) \wedge FLO\_SHAR_{r,v,p,c,cg,s,bd}$$

$$FLO\_SHAR_{r,v,p,c,cg,s,bd} \times \sum_{com \in cg} \sum_{s2 \in \mathbf{rtpcs\_varf_{r,t,p,com,s2}}} \left[VAR\_FLO_{r,v,t,p,com,s2} \times RTCS\_TSFR_{r,t,p,com,s,s2}\right]$$

$$\{=;\leq;\geq\}$$

$$\sum_{s2 \in \mathbf{rtpcs\_varf_{r,t,p,c,s2}}} VAR\_FLO_{r,v,t,p,c,s2} \times RTCS\_TSFR_{r,t,p,c,s,s2}$$

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.*

6.3.33 Equation: EQ_PEAK

Indices: region (r), period (t), commodity group (cg), time-slice (s)

Type: ≥

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:

$$EQ\_PEAK_{r,t,cg,s} \quad \ni \mathbf{com\_peak_{r,cg}} \wedge s \in \mathbf{com\_pkts_{r,cg,s}}$$

$$\left\{ \begin{array}{l}
\sum_{c\in cg} 1/(1+COM\_PKRSV_{r,t,c}) \times COM\_IE_{r,t,c} \times \sum_{p\in\left(\mathbf{top_{r,p,c,'OUT'}} \cup \mathbf{rpc\_ire_{r,p,c,'IMP'}}\right)} \\
\left[\begin{array}{l}
if\ \left(\mathbf{rpc\_pkc_{r,p,c}} \wedge \mathbf{prc\_cap_{r,p}}\right) \\
\left(G\_YRFR_{r,s} \times \sum_{v\in \mathbf{rtp\_cptyr_{r,v,t,p}}} \left[\begin{array}{l}
NCAP\_PKCNT_{r,v,p,s} \times COEF\_CPT_{r,v,t,p} \\
\times \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} - \\ VAR\_SCAP_{r,v,t,p} \times (\ni \mathbf{prc\_rcap_{r,p}}) \end{array} \right) \\
PRC\_CAPACT_{r,p} \times PRC\_ACTFLO_{r,v,p,c}
\end{array} \right] \right) \\
else \\
\quad \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t,p}}} CAL\_FLOFLO_{r,v,t,p,c,s,'OUT'} \times NCAP\_PKCNT_{r,v,p,s} \\
\quad + \\
\quad \sum_{\substack{(p,c)\in \\ \mathbf{rpc\_stg_{r,p,c}}}} \sum_{\substack{\mathbf{rtp\_vintyr_{r,v,t,p}} \\ \mathbf{rpcs\_var_{r,p,c,ts}}}} VAR\_SOUT_{r,v,t,p,c,ts} \times RS\_FR_{r,s,ts} \times NCAP\_PKCNT_{r,v,p,s} \\
\quad + \\
\quad \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t,p}}} CAL\_IRE_{r,v,t,p,c,s,'IMP'} \times NCAP\_PKCNT_{r,v,p,s} \\
\quad - \\
\quad \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t,p}}} CAL\_IRE_{r,v,t,p,c,s,'EXP'} \times NCAP\_PKCNT_{r,v,p,s} \qquad if\ \mathbf{prc\_pkno_{r,p}}
\end{array} \right]
\end{array} \right\}$$

$$\geq$$

$$\sum_{c \in cg} (1 + COM\_PKFLX_{r,t,c,s}) \times$$

$$\left[\begin{array}{l} \displaystyle\sum_{p \in \left(\mathbf{top}_{\mathbf{r,p,c,'IN'}} \cup \mathbf{rpc\_ire}_{\mathbf{r,p,c,'EXP'}}\right) \wedge \left(NOT\ \mathbf{prc\_pkno}_{\mathbf{r,p}}\right)} \left\{ \begin{array}{l} \displaystyle\sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_FLOFLO_{r,v,t,p,c,s,'OUT'} \times FLO\_PKCOI_{r,t,p,c,s} \\ + \\ \displaystyle\sum_{p \in \mathbf{rpc\_ire}_{\mathbf{r,p,c,'EXP'}}} \sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} CAL\_IRE_{r,v,t,p,c,s,'EXP'} \times FLO\_PKCOI_{r,t,p,c,s} \\ + \\ + \left[\displaystyle\sum_{(p,v) \in \mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}}} \left(\begin{array}{l} NCAP\_COM_{r,v,p,c,'IN'} \times COEF\_CPT_{r,v,t,p} \times \\ \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \\ VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{array} \right) \end{array} \right) \right] \times G\_YRFR_{r,s} \\ \\ + \left[\displaystyle\sum_{(p,v) \in \mathbf{rpc\_capflo}_{\mathbf{r,v,p,c}}} \left(\begin{array}{l} COEF\_ICOM_{r,v,t,p,c} \times \\ \left(\begin{array}{l} VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} \\ VAR\_SCAP_{r,v,t,p} \ni (r,p) \in \mathbf{prc\_rcap} \end{array} \right) \end{array} \right) \right] \times G\_YRFR_{r,s} \end{array} \right\} + \\ \displaystyle\sum_{\substack{(p,c) \in \\ \mathbf{rpc\_stg}_{\mathbf{r,p,c}}}} \sum_{\substack{\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \\ \mathbf{rpcs\_var}_{\mathbf{r,p,c,ts}}}} VAR\_SIN_{r,v,t,p,c,ts} \times RS\_FR_{r,s,ts} + \\ \\ \displaystyle\sum_{c \in \mathbf{dem}_{\mathbf{r,c}}} \left(\begin{array}{l} COM\_PROJ_{r,t,c} \times COM\_FR_{r,t,c,s} - \\ \displaystyle\sum_{j=1}^{COM\_STEP_{r,c,'LO'}} VAR\_ELAST_{r,t,c,s,j,'LO'} + \sum_{j=1}^{COM\_STEP_{r,c,'UP'}} VAR\_ELAST_{r,t,c,s,j,'UP'} \end{array} \right) \end{array} \right]$$

6.3.34 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:

$$EQ\_PTRANS_{r,v,t,p,cg1,cg2,s1} \ni (r,v,t,p) \in \left(\mathbf{rp\_flo_{r,p}} \cap \mathbf{rtp\_vintyr_{r,v,t,p}}\right) \wedge s1 \in \mathbf{rps\_s1_{r,p,s1}}$$

$$\wedge \left(s2 \in \mathbf{ts\_map_{r,s2,s1}} \wedge \left(\begin{array}{l} FLO\_SUM_{r,t,p,cg1,c,cg2,s2} \qquad \vee \\ FLO\_FUNC_{r,t,p,cg1,cg2,s2} \wedge NOT\left(FLO\_SUM_{r,t,p,cg1,c,cg2,s2} \vee FLO\_SUM_{r,t,p,cg2,c,cg1,s2}\right) \end{array} \right) \right)$$

$$\sum_{c \in cg2} \sum_{s \in \left(\mathbf{ts\_map_{r,s,s1}} \cap \mathbf{rtpcs\_varf_{r,t,p,c,s}}\right)} VAR\_FLO_{r,v,t,p,c,s} \times RTCS\_TSFR_{r,t,c,s1,s} \quad =$$

$$\sum_{c \in cg1} \sum_{s \in \left(\mathbf{ts\_map_{r,s,s1}} \cap \mathbf{rtpcs\_varf_{r,t,p,c,s}}\right)} \left(COEF\_PTRAN_{r,v,t,p,cg1,c,cg2,s} \times VAR\_FLO_{r,v,t,p,c,s} \times RTCS\_TSFR_{r,t,c,s1,s} \right)$$

$$\times \left(1 + RTP\_FFCX_{r,v,t,p,cg1,cg2} \times \left(if\ \mathbf{prc\_vint_{r,p}}\right)\right)$$

$$COEF\_PTRAN_{r,v,t,p,cg1,c,cg2,ts} \quad ts \in \mathbf{rpcs\_varc_{r,p,c,ts}}$$

$$=$$

$$\sum_{s\in\mathbf{prc\_ts_{r,p,s}}} \left(1\times\left(if\ \mathbf{ts\_map}_{\mathbf{r,ts,s}}\right) + \frac{G\_YRFR_{r,ts}}{G\_YRFR_{r,s}} \times \left(if\ \mathbf{rs\_below_{r,s,ts}}\right) \right)$$

$$\frac{FLO\_FUNC_{r,v,t,p,cg1,cg2,s}}{FLO\_FUNC_{r,v,t,p,cg2,cg1,s} \times \left(if\ FLO\_SUM_{r,v,t,p,cg1,c,cg2,s}\right)} \times \begin{pmatrix} if\ FLO\_FUNC_{r,v,t,p,cg1,cg2,s} \\ \vee FLO\_FUNC_{r,v,t,p,cg2,cg1,s} \end{pmatrix} \times$$

$$\left(1 \times \left(if\ NOT\ FLO\_SUM_{r,v,t,p,cg1,c,cg2,s}\right) + FLO\_SUM_{r,v,t,p,cg1,c,cg2,s}\right)$$

Calculation of SHAPE parameter RTP_FFCX

Case A: Lifetime minus construction time is longer than the construction period

$$PRC\_YMIN_{r,v,p} = B_v + NCAP\_ILED_{r,v,p}$$

$$PRC\_YMAX_{r,v,p} = PRC\_YMIN_{r,v,p} + NCAP\_TLIFE_{r,v,p} - 1$$

$$RTP\_FFCX_{r,v,t,p,cg1,c,cg2} \quad \ni FLO\_FUNCX_{r,v,p,cg1,cg2}$$

$$=$$

$$\sum_{v\in\mathbf{rtp\_vintyr_{r,v,t,p}}} \frac{\sum\limits_{y\in\left(\mathbf{periodyr_{t,y}} \wedge \left[y \le MAX(B(t),PRC\_YMAX_{r,v,p})\right]\right)} SHAPE\left(FLO\_FUNCX_{r,v,p,cg1,cg2}, 1+MIN\left(y, PRC\_YMAX_{r,v,p}\right) - PRC\_YMIN_{r,v,p}\right)}{MAX\left[1, MIN(E(t), PRC\_YMAX_{r,v,p}) - MAX(B(t), PRC\_YMIN_{r,v,p}) + 1\right]} - 1$$

Case B: Lifetime minus construction time is shorter than the construction period => Investment is repeated in construction period

$$PRC\_YMAX_{r,v,p} = NCAP\_TLIFE_{r,v,p} - 1$$

$$RTP\_FFCX_{r,v,t,p,cg1,c,cg2} \quad \ni FLO\_FUNCX_{r,v,p,cg1,cg2}$$

$$=$$

$$\sum_{v\in\mathbf{rtp\_vintyr_{r,v,t,p}}} \frac{SHAPE\left(FLO\_FUNCX_{r,v,p,cg1,cg2}, PRC\_YMAX_{r,v,p}\right)}{PRC\_YMAX_{r,v,p}} - 1$$

6.3.35 Equation: EQL_REFIT

Indices: **region (r), period1 (tt), period2 (t), process (p)**

Type: $\leq$

Related variables: **VAR_NCAP, VAR_RCAP, VAR_SCAP**

Related equations: **EQ(l)_CPT**

Purpose: This equation bounds the investments into a retrofit (RF) and lifetime extension (LE) options defined for a host process according to the available capacity of the host process. The mapping of the RF/LE options is done be the attribute *PRC_REFIT(reg,prc,p).*

Remarks:

- Defining RF/LE options requires that early retirements are allowed in the model.
- Each host process, for which some RF/LE options are to be included, can be modeled to have any number of different RF/LE options;
- Each of the RF/LE options must be modeled in the same way as any new technologies (including the topology, process transformation parameters, availabilities, technical lifetime etc.);
- Whenever the model chooses to invest into a RF/LE option, the same amount of capacity of the host process will be simultaneously retired;
- The investment costs for the RF/LE options should include only the additional investments for the refurbishment in question, but the fixed and variable O&M costs must cover the full costs during the RF/LE operation.

Equation formulation

When retirements are enabled, the TIMES model generator will generate the following equations according to the user-defined PRC_REFIT parameters:

$$EQL\_REFIT_{r,tt,t,prc} \quad \ni \{\exists p : (PRC\_REFIT_{r,prc,p} <> 0 \wedge RTP\_CPTYR_{r,tt,t,p})\}$$

$$\sum_{p \in RTP\_CPTYR_{r,tt,t,p}} COEF\_CPT_{r,tt,t,p} \times (VAR\_NCAP_{r,tt,p} - VAR\_SCAP_{r,tt,t,p}) \times$$

$$\max(DIAG(tt,t), -SIGN(PRC\_REFIT(r,prc,p)))$$

$$\leq / =$$

$$\sum_{v \in RTP\_CPTYR_{r,v,tt,prc}} COEF\_CPT_{r,v,t,prc} \times (VAR\_SCAP_{r,v,tt,prc} - VAR\_SCAP_{r,v,tt-1,prc})$$

where

- $RTP\_CPTYR_{r,v,t,p}$ is the TIMES capacity transfer mapping set
- $COEF\_CPT_{r,v,t,p}$ is the TIMES capacity transfer coefficient parameter
- *DIAG(tt,t)* is the GAMS DIAG function, returning 1 iff *tt=t*
- *SIGN(x)* is the GAMS SIGN function returning ±1 or 0, according to the sign of *x*
- $VAR\_NCAP_{r,v,p}$ is the TIMES new capacity variable
- $VAR\_SCAP_{r,v,t,p}$ is the TIMES cumulative retirement variable

6.3.36 Equations: EQL_SCAP

Indices: region (r), vintage (v), process (p), indicator (ips)

Type: $\leq$

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*

Equation:

$$EQL\_SCAP_{r,v,p,ips} \ni \left(\mathbf{rtp}_{r,v,p} \wedge \left(\begin{array}{l} \left(\mathbf{prc\_rcap_{r,p}} \wedge \left(\neg \mathbf{prc\_vint_{r,p}} \vee \mathbf{obj\_sums_{r,v,p}}\right)\right) \\ \vee \left(\mathbf{prc\_vint_{r,p}} \wedge NCAP\_OLIFE_{r,v,p}\right) \end{array} \right) \right)$$

$$\left(\begin{array}{ll} \sum\limits_{t=v+RVPRL_{r,v,p}} VAR\_SCAP_{r,v,t,p} & if\ ips = \text{'N'} \\ \sum\limits_{\mathbf{rtp\_cptyr_{r,v,t,p}}} \sum\limits_{\mathbf{prc\_ts_{r,p,s}}} \dfrac{VAR\_ACT_{r,v,t,p,s} \times D_t}{PRC\_CAPACT_{r,p} \times NCAP\_OLIFE_{r,v,p}} & otherwise \end{array} \right)$$

$$\leq$$

$$VAR\_NCAP_{r,tt(v),p} + NCAP\_PASTI_{r,v,p} - \sum_{\substack{\mathbf{obj\_sums_{r,v,p}} \\ \mathbf{prc\_rcap_{r,p}}}} VAR\_SCAP_{r,v,\text{'0'},p}$$

6.3.37 Equations: EQ_SLSIFT

This block of equations implements the load shifting constraints. Enabling load-shifting for demand or final energy commodities is supported by introducing load-shifting processes. Load shifting processes must be characterized as timeslice storage processes, and must have $STG\_SIFT_{r,t,p,D,s}$ specified for the load commodity D. The constraints for the load-shifting can be divided into different types, as described below.

Indices: **region (r), period (t), process (p), time slice (s)**

Related variables: **VAR_SIN, VAR_SOUT, VAR_UDP, VAR_UPS**

Notation:

- *P(s)* – Set of parent timeslice of timeslice *s* in the timeslice tree
- *C(s)* – Set of child timeslices of timeslice s in the timeslice tree
- S_i – Set of timeslices belonging to season *i* (e.g. summer, winter)
- $VAR\_DAC_{r,t,p,s,bd}$ – activities of bi-directional load storage (advance / delay); virtual variables internally implemented with *VAR_UDP*

A. Seasonal balances

Purpose: To impose an equality balance for the storage input and output flows on the seasonal or weekly level (the first level above DAYNITE in the model). See section 4.3.9 for some more detailed discussion of load-shifting processes.

Type: =

Assume that the output of the load-shifting process is commodity *D*, and the input is commodity *com*. Usually these would be the same commodity, i.e. *com*=*D*, but TIMES allows also the load-shifting process to convert an upstream commodity com into *D*, while shifting its load. The seasonal balances could then be written as follows:

$$\sum_{s \in S_i} VAR\_SIN(r,t,p,com,s) = \sum_{s \in S_i} VAR\_SOUT(r,t,p,D,s), \quad \forall (r,t), i = 1,...,NS$$

where the *Si*'s are the subsets of time-slices in each season $i = 1,\dots,NS$. For example, S_1 could be for winter, S_2 for summer and S_3 for intermediate. These constraints ensure that the entirety of the demand is met and forbid cross-seasonal load shifting.

B. Maximum allowed deviations from nominal demand loads

Purpose: To impose a limit for the deviation from the original demand loads, specified by *STG_SIFT* (in gross terms, i.e. including any optional efficiency loss that may have been defined by *STG_EFF*).

Type: ≤

As the model generator automatically aggregates the demand into *VAR_COMPRD* variables, on the process timeslices **s** the constraints for the maximum allowed deviations from the exogenous nominal demand levels can be formulated as follows:

$$VAR\_SIN(r,t,p,com,s)+VAR\_SOUT(r,t,p,D,s) \quad \leq \quad STG\_SIFT(r,t,p,D,s)\times$$
$$\sum_{ts\in RS\_TREE_{r,s}}\left(\frac{RTCS\_TSFR_{r,t,D,s,ts}}{COM\_IE_{r,t,D,ts}}\times VAR\_COMPRD_{r,t,D,ts}\right) \quad \forall(r,t,s),\ s\in\{ts\mid PRC\_TS_{r,p,ts}\}$$

In addition, the user can also define maximum fractions for the total load shifting within each season in proportion to the total demand in that season, by specifying *STG_SIFT(r,y,p,'ACT',s)*. In this case, the constraints for the maximum allowed shifting in proportion to the seasonal total demand in season *i* can be formulated as follows:

$$\sum_{ts\in S_i} VAR\_SOUT_{r,t,p,D,ts} \leq STG\_SIFT_{r,t,p,ACT,s}\cdot \sum_{\substack{ts\in\\ RS\_TREE_{S_i}}}\frac{RTCS\_TSFR_{r,t,D,s,ts}\cdot VAR\_COMPRD_{r,t,D,ts}}{COM\_IE_{r,t,D,ts}}$$

C. Balance over user-defined time-window

Purpose: To impose the requirement that withing each N consequtive hours, the total demand (after load shifting) must be at least equal to the original (unshifted) demand. This requirement can be specified by *ACT_TIME(r,y,p,'N')*=N.

Type: $\geq$

Equation:

$$\sum_{ts\in C(P(s))}\left[\begin{array}{l}\left(\sum_{v} VAR\_SIN_{r,v,t,p,com,ts} - VAR\_SOUT_{r,v,t,p,D,ts}\right)\cdot\\ \left(\mathrm{mod}(Hour(s)-Hour(ts),24) < ACT\_TIME^{N}_{r,t,p}\right)\end{array}\right] \geq 0 \quad \forall(r,t,s), s\in\{sl\mid PRC\_TS_{r,p}\}$$

D. Maximum advance or delay for meeting the shited loads

Purpose: To impose the requirement that the shifted loads have to be met within at most N hours of advance of delay. This requirement can be specified by *ACT_TIME(r,y,p,bd)*=N, where bd = FX / UP / LO, such that LO=advance, UP=delay, and FX means that the limits for advance and delay are symmetric.

Type: $= / \leq$

Equations (illustrated here for the symmetric case bd=FX only):

$$VAR\_DAC_{r,t,p,s,LO} - VAR\_DAC_{r,t,p,s-1,LO} + VAR\_DAC_{r,t,p,s-1,UP} - VAR\_DAC_{r,t,p,s,UP}$$
$$= \sum_{v}\left(VAR\_SIN_{r,v,t,p,com,s} - VAR\_SOUT_{r,v,t,p,D,s}\right) \quad \forall(r,t,s), s\in\{ts\mid PRC\_TS_{r,p}\}$$

$$VAR\_DAC_{r,t,p,s,UP} + VAR\_DAC_{r,t,p,s,LO} -$$
$$\sum_{ts\in C(P(s))}\left[\begin{array}{c}\left(\sum_{v} VAR\_SIN_{r,v,t,p,ts} + VAR\_DAC_{r,t,p,ts-1,UP} - VAR\_DAC_{r,t,p,ts,UP}\right)\cdot\\ \left(\mathrm{mod}(Hour(s)-Hour(ts),24) < ACT\_TIME^{FX}_{r,t,p}\right)\end{array}\right] \leq 0 \quad \forall(r,t,s)$$

6.3.38 Equation: EQ_STGCCL

Indices: **region (r), vintage year (v), period (t), process (p)**

Type: $\geq$

Related variables: **VAR_NCAP, VAR_SOUT, VAR_UPS**

Related equations: **EQ(l)_CPT**

Purpose: This equation approximates the impact of the storage degradation on the investment decision, by assuming a targeted maximum number of cycles per year, *STG_MAXCYC*(r,v,p)/*NCAP_TLIFE*(r,v,p), where *STG_MAXCYC* is the maximum number of cycles (e.g. 3000 or 4500) over the lifetime and *NCAP_TLIFE* is the technical lifetime of the storage process (e.g. 15 years).

Remarks:

- This feature becomes of importance if the investment in a storage process is related to the investment of another process, e.g. the battery of an electric car. In this example and in the case of excessive cycling, the optimiser will opt to pay the replacement cost of the battery than re-investing in a car, if the car has not reached the end of its lifetime.
- If the number of cycles in a year exceeds the annual targeted cycles, then a replacement of the storage has implicitly happened and the replacement capacity that supports this extensive cycling is represented with variable *VAR_STGCC*, which is actually a virtual cariable (internally implemented with $VAR\_UPS_{r,v,t,p,s}$). The variable is included in the objective function multiplied with the annualised investment cost to account for the replacement cost of the storage.

Equation formulation:

$$\left(COEF\_CPT_{r,v,t,p} \cdot VAR\_NCAP_{r,v,p} + VAR\_STGCC_{r,v,t,p}\right) \times \max_s\left(NCAP\_AF_{r,v,p,s}\right) \times PRC\_CAPACT_{r,p} \geq$$

$$\left(\frac{ncap\_tlife_{r,v,p}}{stg\_maxcyc_{r,v,p}} \times \frac{\sum_{c,s} VAR\_SOUT_{r,v,t,p,c,s}}{prc\_actflo_{r,v,p,c}}\right), \quad \forall (r,v,t,p) \in RTP\_VINTYR$$

6.3.39 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:

$$EQ\_STGAUX_{r,v,t,p,c,s} \ni \left(\mathbf{rtp\_vintyr_{r,v,t,p}} \wedge \mathbf{rpcs\_var_{r,p,c,s}} \wedge \mathbf{prc\_map_{r,'STG',p}} \wedge \neg \mathbf{rpc\_stg_{r,p,c}}\right)$$

$$VAR\_FLO_{r,v,t,p,c,s} =$$

$$PRC\_ACTFLO_{r,v,p,c} \times \left(\left(\begin{array}{l} VAR\_ACT_{r,v,t-1,p,s} \times \dfrac{G\_YRFR_{r,s}}{RS\_STGPRD_{r,s}} - \\ \left(\sum_{c \in \left\{ \begin{smallmatrix} \mathbf{top}_{IN} \\ \mathbf{prc\_stgips} \end{smallmatrix} \right\}} \dfrac{VAR\_SIN_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} - \sum_{c \in \left\{ \begin{smallmatrix} \mathbf{top}_{OUT} \\ \mathbf{prc\_stgips} \end{smallmatrix} \right\}} \dfrac{VAR\_SOUT_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} \right) \times \\ \left(\sum_{y \in \left\{ \begin{smallmatrix} \mathbf{periodyr_{t,y}} \\ y \geq M(t) \end{smallmatrix} \right\}} \left(1 - STG\_LOSS_{r,v,p,s}\right)^{(E(t)-y+0.5)} \right) \end{array} \right) \times \left(\left(1 - STG\_LOSS_{r,v,p,s}\right)^{(M(t)-E(t)-Mod(D(t)/2,1))} \right)^{\left(1\, if\ \mathbf{prc\_map_{r,'STK',p}}\right)} \right)$$

$$+ \sum_{com \in \mathbf{top}_{IN}} VAR\_SIN_{r,v,t,p,com,s} \times \left(COEF\_PTRAN_{r,v,p,com,com,c,s} \right)$$

$$+ \sum_{com \in \mathbf{top}_{OUT}} VAR\_SOUT_{r,v,t,p,com,s} \times \left(\frac{1}{COEF\_PTRAN_{r,v,p,c,c,com,s}}\ if\ COEF\_PTRAN_{r,v,p,c,c,com,s} > 0 \right)$$

6.3.40 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.

6.3.40.1 EQ_STGTSS: Storage between timeslices (including night-storage devices):

Equation:

$$EQ\_STGTSS_{r,v,t,p,s} \quad \forall (r,v,t,p,s) \in \left(\mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}} \wedge \mathbf{rps\_stg}_{\mathbf{r,p,s}} \wedge \mathbf{prc\_map}_{\mathbf{r,'STG',p}} \right)$$

$$VAR\_ACT_{r,v,t,p,s} =$$

$$\left[\begin{array}{l} VAR\_ACT_{r,v,t,p,s-1} + \\ \sum\limits_{c \in \mathbf{rpc\_stg}} \\ \quad \left(\begin{array}{l} \text{if p is a night-storage device:} \\ \quad \dfrac{VAR\_SIN_{r,v,t,p,c,s-1}}{PRC\_ACTFLO_{r,v,p,c}} \times \left(if \;\; s-1 \in \mathbf{prc\_nstts}_{\mathbf{r,p,s-1}} \wedge \mathbf{top}_{\mathbf{r,p,c,'IN'}} \right) - \\ \quad \dfrac{VAR\_SOUT_{r,v,t,p,c,s-1}}{PRC\_ACTFLO_{r,v,p,c}} \times \left(if \;\; s-1 \notin \mathbf{prc\_nstts}_{\mathbf{r,p,s-1}} \wedge \mathbf{top}_{\mathbf{r,p,c,'OUT'}} \right) \\ \text{if p is not a night-storage device} \\ \quad + \sum\limits_{top_{r,p,c,'IN'}} \dfrac{VAR\_SIN_{r,v,t,p,c,s-1}}{PRC\_ACTFLO_{r,v,p,c}} - \sum\limits_{top_{r,p,c,'OUT'}} \dfrac{VAR\_SOUT_{r,v,t,p,c,s-1}}{PRC\_ACTFLO_{r,v,p,c}} \end{array} \right) \\ - \sum\limits_{ts \in \{sl | \mathbf{prc\_ts}_{r,p,sl} \cap \mathbf{rs\_below}_{r,sl,s}\}} VAR\_SOUT_{r,v,p,'ACT',ts} \times RS\_FR_{r,s-1,ts} \\ - \left[\left(\dfrac{VAR\_ACT_{r,v,t,p,s} + VAR\_ACT_{r,v,t,p,s-1}}{2} \right) \right] \times STG\_LOSS_{r,v,p,s} \times \dfrac{G\_YRFR_{r,s}}{RS\_STGPRD_{r,s}} \end{array} \right]$$

$$+ STG\_CHRG_{r,t,p,s-1}$$

6.3.40.2 EQ_STGIPS: Storage between periods

Equation:

$$EQ\_STGIPS_{r,v,t,p} \quad \forall (r,v,t,p) \in \left(\mathbf{rtp\_vintyr_{r,v,t,p}} \cap \mathbf{prc\_map_{r,'STK',p}}\right)$$

$$VAR\_ACT_{r,v,t,p,'ANNUAL'} =$$

$$\sum_{\substack{v \in \mathbf{rtp\_vintyr_{r,v,t-1,p}} \\ p \in \mathbf{prc\_vint}}} \begin{bmatrix} VAR\_ACT_{r,v,t-1,p,'ANNUAL'} \\ \times (1 - STG\_LOSS_{r,v,p,'ANNUAL'})^{D(t)} \end{bmatrix} + \sum_{p \notin \mathbf{prc\_vint}} \begin{bmatrix} VAR\_ACT_{r,t-1,t-1,p,'ANNUAL'} \\ \times (1 - STG\_LOSS_{r,v,p,'ANNUAL'})^{D(t)} \end{bmatrix}$$

$$+ \left[\sum_{\substack{y \in \mathbf{periodyr_{t,y}} \\ c \in \mathbf{rpc\_stg}_{r,p,c}}} \left(\begin{matrix} \displaystyle\sum_{c \in \mathbf{top}_{IN}} \frac{VAR\_SIN_{r,v,t,p,c,'ANNUAL'}}{PRC\_ACTFLO_{r,v,p,c}} - \\ \displaystyle\sum_{c \in \mathbf{top}_{OUT}} \frac{VAR\_SOUT_{r,v,t,p,c,'ANNUAL'}}{PRC\_ACTFLO_{r,v,p,c}} \end{matrix} \right) \times (1 - STG\_LOSS_{r,v,p,'ANNUAL'})^{(E(t)-y+0.5)} \right]$$

$$+ \quad STG\_CHRG_{r,t,p,'ANNUAL'} (when\, ORD(t) = 1)$$

6.3.41 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:

$$EQ(l)\_STGIN/OUT_{r,t,p,c,s} \quad \ni \left\{ \begin{array}{l} (r,t,p,c) \in \mathbf{rtpc_{r,t,p,c}} \wedge STGIN/OUT\_BND_{r,t,p,c,s,bd} \wedge s \in \underset{\uparrow}{\mathbf{rps\_prcts_{r,p,s}}} \wedge \\ \left(\mathbf{prc\_vint_{r,p}} \vee \left(NOT\ \mathbf{prc\_ts_{r,p,s}}\right)\right) \end{array} \right\}$$

All timeslices **s** *at or above the timeslice level of the process* (**prc_tsl**).

$$\sum_{ts \in \left(\mathbf{prc\_ts_{r,p,ts}} \wedge \mathbf{ts\_map_{r,s,ts}}\right)} \sum_{v \in \mathbf{rtp\_vintyr_{r,v,t,p}}} VAR\_SIN/SOUT_{r,v,t,p,c,ts}$$

$$(\leq/\geq/=) \qquad STGIN/OUT\_BND_{r,t,p,c,s,bd}$$

where the equation sign is indicated by equation index **l** based on the bound type **bd**.

6.3.42 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:

$$EQ\_STSBAL_{r,v,t,p,s} \quad \forall (r,v,t,p,s) \in \left(\mathbf{rtp\_vintyr_{r,v,t,p}} \cap \mathbf{prc\_ts_{r,p,s}} \cap \neg \mathbf{rps\_stg_{r,p,s}}\right)$$

$$\sum_{\mathbf{rs\_below_{r,'ANNUAL',s}}} VAR\_ACT_{r,v,t,p,s} =$$

$$\sum_{\mathbf{rs\_below_{r,'ANNUAL',s-1}}} \left(\begin{array}{l} VAR\_ACT_{r,v,t,p,s-1} + VAR\_SOUT_{r,v,t,p,'ACT',s-1} - \\ \sum\limits_{ts \in \{sl | \mathbf{prc\_ts}_{r,p,sl} \wedge \mathbf{rs\_below1_{r,sl,s}}\}} VAR\_SOUT_{r,v,t,p,'ACT',ts} \times RS\_FR_{r,s-1,ts} - \\ \left(\frac{VAR\_ACT_{r,v,t,p,s} + VAR\_ACT_{r,v,t,p,s-1}}{} \right) \times STG\_LOSS_{r,v,p,s} \times \frac{G\_YRFR_{r,s}}{RS\_STGPRD_{r,s}} \end{array} \right) +$$

$$+ \sum_{s \in \{sl | \mathbf{annual}(sl)\}} \left[\begin{array}{l} \sum\limits_{c \in \left\{ \begin{array}{l} \mathbf{stgips} \cap \\ \mathbf{top_{'IN'}} \end{array} \right\}} \frac{VAR\_SIN_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} - \sum\limits_{c \in \left\{ \begin{array}{l} \mathbf{stgips} \cap \\ \mathbf{top_{'OUT'}} \end{array} \right\}} \frac{VAR\_SOUT_{r,v,t,p,c,s}}{PRC\_ACTFLO_{r,v,p,c}} - \\ VAR\_SOUT_{r,v,t,p,'ACT',s} \end{array} \right]$$

6.3.43 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:

$$EQ(l)\_UCRTP_{uc\_n,r,t,p,'CAP',bd} \quad \ni \left(\mathbf{rtp}_{r,t,p} \wedge \mathbf{uc\_dynbnd}_{uc\_n,bd} \wedge \left(\sum_{side} \left(UC\_CAP_{uc\_n,side,r,t,p} \right) > 0 \right) \right)$$

$$VAR\_CAP_{r,t,p}$$

$$\{\le;=;\ge\}$$

$$VAR\_CAP_{r,t-1,p} \times \left(UC\_CAP_{uc\_n,'LHS',r,t,p} \right)^{(M(t)-M(t-1))} + UC\_CAP_{uc\_n,'RHS',r,t,p} \times (M(t)-M(t-1))$$

Case B. For NCAP:

$$EQ(l)\_UCRTP_{uc\_n,r,t,p,'NCAP',bd} \quad \ni \left(\mathbf{rtp}_{r,t,p} \wedge \mathbf{uc\_dynbnd}_{uc\_n,bd} \wedge \left(\sum_{side} \left(UC\_NCAP_{uc\_n,side,r,t,p} \right) > 0 \right) \right)$$

$$VAR\_NCAP_{r,t,p}$$

$$\{\le;=;\ge\}$$

$$VAR\_NCAP_{r,t-1,p} \times \left(UC\_NCAP_{uc\_n,'LHS',r,t,p} \right)^{(M(t)-M(t-1))} + UC\_NCAP_{uc\_n,'RHS',r,t,p} \times (M(t)-M(t-1))$$

Case C. For ACT:

$$EQ(l)\_UCRTP_{uc\_n,r,t,p,'ACT',bd} \quad \ni \left(\mathbf{rtp}_{r,t,p} \wedge \mathbf{uc\_dynbnd}_{uc\_n,bd} \wedge \left(\sum_{side} \left(UC\_ACT_{uc\_n,side,r,t,p,'ANNUAL'} \right) > 0 \right) \right)$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \sum_{s \in \mathbf{prc\_ts}} VAR\_ACT_{r,t,p,s} \quad \{\le;=;\ge\}$$

$$\sum_{v \in \mathbf{rtp\_vintyr}_{\mathbf{r,v,t,p}}} \sum_{s \in \mathbf{prc\_ts}} VAR\_ACT_{r,t-1,p,s} \times \left(UC\_ACT_{uc\_n,'LHS',r,t,p,'ANNYAL'} \right)^{(M(t)-M(t-1))}$$

$$+UC\_ACT_{uc\_n,'RHS',r,t,p,'ANNUAL'} \times (M(t)-M(t-1))$$

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:

$$EQ(l)\_UC(R)(T)(S)_{(r),uc\_n,(t),(s)} \quad \forall \left\{ \begin{array}{l} UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each_{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_each_{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each_{r,uc\_n,s}}\right) \end{array} \right\}$$

$$\left(\sum_{r \in \mathbf{uc\_r\_sum_{r,uc\_n}}}\right)\left(\sum_{t \in \mathbf{uc\_t\_sum_{r,uc\_n,t}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum_{r,uc\_n,s}}}\right) LHS_{r,t,s} \{= / \geq / \leq\} \; UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd}$$

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.

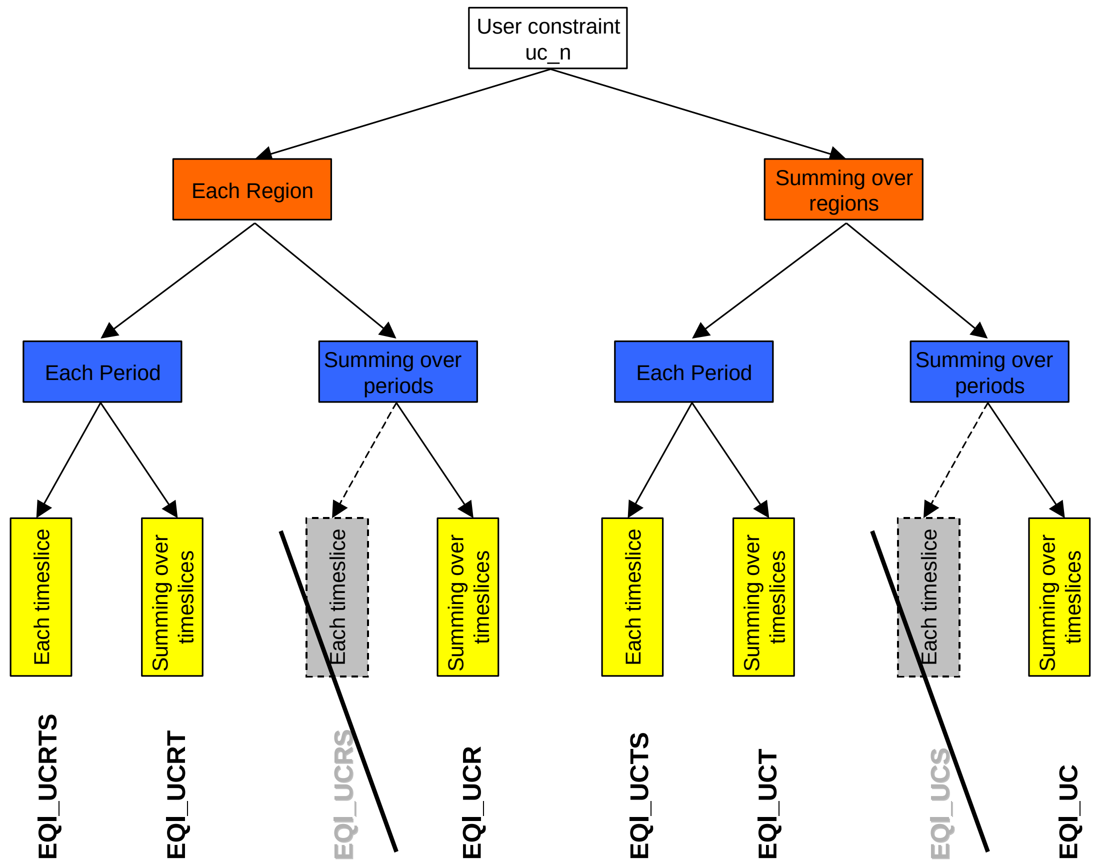


Figure 14: The allowed combinations of region, period and timeslice for user constraints.

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.

$$EQE\_UC(R)(T)(S)_{(r),uc\_n,(t),(s)} \quad \forall \left\{ \begin{array}{l} UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}}\right) \end{array} \right\}$$

$$\left(\sum_{r \in uc\_r\_sum_{r,uc\_n}}\right)\left(\sum_{t \in uc\_t\_sum_{r,uc\_n,t}}\right)\left(\sum_{s \in uc\_ts\_sum_{r,uc\_n,s}}\right) LHS_{r,t,s} = VAR\_UC(R)(T)(S)_{(r),uc\_n,(t),(s)}$$

$$\begin{aligned} VAR\_UC(R)(T)(S).LO_{(r),uc\_n,(t),(s)} &= UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),'LO'} \\ VAR\_UC(R)(T)(S).UP_{(r),uc\_n,(t),(s)} &= UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),'UP'} \\ VAR\_UC(R)(T)(S).FX_{(r),uc\_n,(t),(s)} &= UC\_RHS(R)(T)(S)_{(r),uc\_n,(t),(s),'FX'} \end{aligned}$$

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:

$$EQ(l)\_UCRS_{r,uc\_n,t,tsl,s} \ni \left\{ \begin{array}{c} UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \left(r \in \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}\right) \wedge \left(t \in \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}\right) \\ \wedge (s \in \{ts \mid \mathbf{ts\_grp}_{r,tsl,s} \wedge \bigcup_{side} \mathbf{uc\_tsl}_{r,uc\_n,side,tsl}\}) \end{array} \right\}$$

$$LHS_{r,t,s} \qquad \{= / \geq / \leq\} \sum_{\mathbf{uc\_tsl}(r,ucn,'RHS',tsl)} RHS_{r,t,s-RS\_STG(r,s)} + UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd}$$

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. As an alternative to using **uc_tsl**, $\mathbf{uc\_attr}_{r,uc\_n,side,uc\_grptype,tslvl}$ can also be used, with any **uc_grptype** (**ucn** recommended). 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:

$$EQ(l)\_UC(R)SU(S)_{(r),uc\_n,t,(s)} \ni \left\{ \begin{array}{l} UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_succ}_{\mathbf{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}}\right) \end{array} \right\}$$

$$\left(\sum_{r \in \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}}}\right) LHS_{r,t,s} \{= / \geq / \leq\} \left(\sum_{r \in \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}}}\right) RHS_{r,t+1,s} + UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd}$$

Similarly, dynamic user constraints of type (**t–1,t**) can be written as follows:

$$EQ(l)\_UC(R)SU(S)_{(r),uc\_n,t,(s)} \ni \left\{ \begin{array}{l} UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd} \wedge \left(r \in \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}}\right) \\ \wedge \left(t \in \mathbf{uc\_t\_succ}_{\mathbf{r,uc\_n,t}}\right) \wedge \left(s \in \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}}\right) \end{array} \right\}$$

$$\left(\sum_{r \in \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}}}\right) LHS_{r,t,s} \{= / \geq / \leq\} \left(\sum_{r \in \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}}\right)\left(\sum_{s \in \mathbf{uc\_ts\_sum}_{\mathbf{r,uc\_n,s}}}\right) RHS_{r,t-1,s} + UC\_RHS(R)T(S)_{(r),uc\_n,t,(s),bd}$$

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

List of user constraints and variables

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*, *COMNET*, *COMPRD*, *NCAP*, *CAP*, *UCN*, and 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*, *BUILDUP*, *CAPACT*, *CAPFLO*, *NEWFLO*, *ONLINE*, *EFF*, *NET*, *CUMSUM*, *PERDISC*, *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,
• **uc_r_each**$_{\mathbf{r,uc\_n}}$:	regions **r** for which the user constraint **uc_n** is generated,
• **uc_r_sum**$_{\mathbf{r,uc\_n}}$:	regions **r** being summed over in the user constraint **uc_n**,
• **uc_t_each**$_{\mathbf{r,uc\_n,t}}$:	periods **t** for which the user constraint **uc_n** is generated,
• **uc_t_sum**$_{\mathbf{r,uc\_n,t}}$:	periods **t** being summed over in the user constraint **uc_n**,
• **uc_ts_each**$_{\mathbf{r,uc\_n,ts}}$:	timeslices **ts** for which the user constraint **uc_n** is generated,
• **uc_ts_sum**$_{\mathbf{r,uc\_n,ts}}$:	timeslices **ts** being summed over in the user constraint **uc_n**,
• **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.

$$ACT_{r,t,s,'LHS'}$$

$$=$$

$$\sum_{(v,p)\in\mathbf{rtp\_vintyr_{r,v,t,p}}} \sum_{ts\in\mathbf{prc\_ts_{r,p,ts}}} \left(\begin{array}{l} VAR\_ACT_{r,v,t,p,ts} \times UC\_ACT_{uc\_n,'LHS',r,t,p,ts} \times \left(RS\_FR_{r,s,ts}\right) \\ \times \\ \left(\sum_{cur\in\mathbf{rdcur_{r,cur}}} OBJ\_ACOST_{r,t,p,cur} \right) \quad if\ UC\_ATTR_{r,uc\_n,'LHS','ACT','COST'}\ is\ given \end{array} \right)$$

$$FLO_{r,t,s,'LHS'}$$

$$=$$

$$\sum_{(p,c,ts)\in\mathbf{rtpcs\_varf_{r,t,p,c,ts}}} \sum_{v\in\mathbf{rtp\_vintyr_{r,v,t,p}}}$$

$$\left\{ \begin{array}{l} VAR\_FLO_{r,v,t,p,c,ts} \times UC\_FLO_{uc\_n,'LHS',r,t,p,c,ts} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\ \times \\ \left[\sum_{cur\in\mathbf{rdcur_{r,cur}}} \left(\begin{array}{l} OBJ\_FCOST_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','COST'}\ is\ given \\ + \\ OBJ\_FDELV_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','DELIV'}\ is\ given \\ - \\ OBJ\_FSUB_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','SUB'}\ is\ given \\ + \\ OBJ\_FTAX_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','TAX'}\ is\ given \end{array} \right) \right] \end{array} \right\}$$

$$IRE_{r,t,s,'LHS'}$$
$$= \sum_{(p,c,ts)\in \mathbf{rtpcs\_varf_{r,t,p,c,ts}}} \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t,p}}} \sum_{ie\in \mathbf{rpc\_ire_{r,p,c,ie}}}$$
$$\left[\begin{array}{l} VAR\_IRE_{r,v,t,p,c,ts,ie} \times UC\_IRE_{uc\_n,'LHS',r,t,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\ \times \\ \sum\limits_{cur\in \mathbf{rdcur}_{r,cur}} \left(\begin{array}{l} OBJ\_FCOST_{r,t,p,c,s,cur} \;\; if \; \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','COST'} \\ + \\ OBJ\_FDELV_{r,t,p,c,s,cur} \;\; if \; \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','DELIV'} \\ - \\ OBJ\_FSUB_{r,t,p,c,s,cur} \;\; if \; \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','SUB'} \\ + \\ OBJ\_FTAX_{r,t,p,c,s,cur} \;\; if \; \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','TAX'} \end{array} \right) \end{array}\right]$$

$$COMPRD_{r,t,s,'LHS'}$$
$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc_{r,t,c,ts}}} \left(\begin{array}{l} VAR\_COMPRD_{r,t,c,ts} \times UC\_COMPRD_{uc\_n,'LHS',r,t,c,s} \\ \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\ \sum\limits_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t,c,ts,uc\_cost,cur} \quad if \; \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMPRD',uc\_cost} \right) \end{array} \right)$$

$$COMNET_{r,t,s,'LHS'}$$
$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc_{r,t,c,ts}}} \left(\begin{array}{l} VAR\_COMNET_{r,t,c,ts} \times UC\_COMNET_{uc\_n,'LHS',r,t,c,s} \\ \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\ \sum\limits_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMNT_{r,t,c,ts,uc\_cost,cur} \quad if \; \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMNET',uc\_cost} \right) \end{array} \right)$$

$$NCAP_{r,t,'LHS'}$$

$$=$$

$$\sum_{p}\left[\begin{array}{l} VAR\_NCAP_{r,t,p} \times UC\_NCAP_{uc\_n,'LHS',r,t,p} \times \\ \left(\begin{array}{l} \sum\limits_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','COST'}\ is\ given \\ - \\ \sum\limits_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','SUB'}\ is\ given \\ + \\ \sum\limits_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','TAX'}\ is\ given \end{array}\right) \\ \end{array}\right]$$

$$CAP_{r,t,p,'LHS'}$$

$$=$$

$$\sum_{p}\left[\begin{array}{l} VAR\_CAP_{r,t,p} \times UC\_CAP_{uc\_n,'LHS',r,t,p} \\ \times \\ PRC\_CAPACT_{r,p} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','CAP','CAPACT'}\ is\ given \end{array}\right]$$

$$CUMCOM_{r}$$

$$=$$

$$\sum_{\mathbf{rc\_cumcom_{r,com\_var,c}}} VAR\_CUMCOM_{r,c,com\_\mathrm{var},y1,y2} \times UC\_CUMCOM_{uc\_n,r,com\_\mathrm{var},c,y1,y2}$$

$$CUMFLO_{r}$$

$$=$$

$$\sum_{\mathbf{rpc\_cumflo_{r,p,c}}} \left(VAR\_CUMFLO_{r,p,c,y1,y2} \times UC\_CUMFLO_{uc\_n,r,p,c,y1,y2}\right) +$$

$$\sum_{\mathbf{rpc\_cumflo_{r,p,'ACT'}}} \left(VAR\_CUMFLO_{r,p,'ACT',y1,y2} \times UC\_CUMACT_{uc\_n,r,p,y1,y2}\right)$$

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:

$$EQ(l)\_UCT_{uc\_n,t} \ni UC\_RHST_{uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum_{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_each_{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'} \right)$$

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:

$$\{\leq;=;\geq\}$$
$$UC\_RHSRT_{r,uc\_n,t,l} + UC\_TIME_{uc\_n,r,t} \times D_t$$

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:

$$EQE\_UCTS_{uc\_n,t,s} \ni UC\_RHSRTS_{uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each}_{\mathbf{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum}_{\mathbf{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'} \right)$$

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:

$$EQ(l)\_UCRS_{r,uc\_n,t,tsl,s} \ni UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \mathbf{uc\_r\_each}_{\mathbf{r,uc\_n}} \wedge \mathbf{uc\_t\_each}_{\mathbf{r,uc\_n,t}}$$
$$\wedge (s \in \{ts \mid \mathbf{ts\_grp}_{r,tsl,ts} \wedge \bigcup_{side} \mathbf{uc\_tsl}_{r,uc\_n,side,tsl}\})$$

$$\begin{pmatrix} ACT_{r,t,s,'LHS'} + FLO_{r,t,s,'LHS'} + IRE_{r,t,s,'LHS'} \\ + COMNET_{r,t,s,'LHS'} + COMPRD_{r,t,s,'LHS'} \end{pmatrix} \times \frac{1}{G\_YRFR_{r,s}}$$
$$+$$
$$\left(NCAP_{r,t,'LHS'} + CAP_{r,t,'LHS'}\right)$$

When control parameter VAR_UC=NO:

$$\{\le;=;\ge\}$$

$$UC\_RHSRTS_{r,uc\_n,t,s,l} + UC\_TIME_{uc\_n,r,t} \times D_t$$
$$+$$
$$\left(\begin{array}{l} \left(\begin{array}{l} ACT_{r,t,d(s),'RHS'} + FLO_{r,t,d(s),'RHS'} + IRE_{r,t,d(s),'RHS'} \\ + COMNET_{r,t,d(s),'RHS'} + COMPRD_{r,t,d(s),'RHS'} \end{array} \right) \times \frac{1}{G\_YRFR_{r,d(s)}} \\ + \\ \left(NCAP_{r,t,'RHS'} + CAP_{r,t,'RHS'} \right) \\ \end{array} \right) if\ \mathbf{uc\_tsl}_{r,uc\_n,'RHS',s}$$

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.

$$=$$
$$VAR\_UCRTS_{r,uc\_n,t,s} + UC\_TIME_{uc\_n,r,t} \times D_t$$
$$+$$
$$\left(\begin{array}{l} \left(\begin{array}{l} ACT_{r,t,d(s),'RHS'} + FLO_{r,t,d(s),'RHS'} + IRE_{r,t,d(s),'RHS'} \\ + COMNET_{r,t,d(s),'RHS'} + COMPRD_{r,t,d(s),'RHS'} \end{array} \right) \times \frac{1}{G\_YRFR_{r,d(s)}} \\ + \\ \left(NCAP_{r,t,'RHS'} + CAP_{r,t,'RHS'} \right) \\ \end{array} \right) if\ \mathbf{uc\_tsl}_{r,uc\_n,'RHS',s}$$

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:

$$ACT\_GROW_{r,t,s,'LHS'}$$
$$=$$
$$\sum_{(p,v)\in \mathbf{rtp\_vintyr_{r,v,t,p}}} \sum_{ts\in \mathbf{prc\_ts_{r,p,ts}}} \left(\begin{array}{l} VAR\_ACT_{r,v,t,p,ts} \times UC\_ACT_{uc\_n,'LHS',r,t,p,ts} \times \left(RS\_FR_{r,s,ts}\right) \\ \times \\ \left(\sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ACOST_{r,t,p,cur} \right) \quad if\ UC\_ATTR_{r,uc\_n,'LHS','ACT','COST'}\ is\ given \\ \times \\ \left(UC\_ACT_{uc\_n,'LHS',r,t,p,ts}\right)^{M(t+1)-M(t)-1} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','ACT','GROWTH'}\ is\ given \end{array} \right)$$

$$FLO\_GROW_{r,t,s,'LHS'}$$

$$=$$

$$\sum_{(p,c,ts)\in\mathbf{rtpcs\_varf_{r,t,p,c,ts}}}\ \sum_{v\in\mathbf{rtp\_vintyr_{r,v,t,p}}}$$

$$\left\{\begin{array}{l} VAR\_FLO_{r,v,t,p,c,ts} \times UC\_FLO_{uc\_n,'RHS',r,t,p,c,ts} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\ \times \\ \left[\sum_{cur\in\mathbf{rdcur_{r,cur}}} \left(\begin{array}{l} OBJ\_FCOST_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','COST'}\ is\ given \\ + \\ OBJ\_FDELV_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','DELIV'}\ is\ given \\ + \\ OBJ\_FSUB_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','SUB'}\ is\ given \\ + \\ OBJ\_FTAX_{r,t,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','TAX'}\ is\ given \end{array} \right) \right] \\ \times \\ \left(UC\_FLO_{uc\_n,'LHS',r,t,p,c,ts} \right)^{M(t+1)-M(t)-1} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','FLO','GROWTH'}\ is\ given \end{array} \right\}$$

$$IRE\_GROW_{r,t,s,'LHS'}$$

$$= \sum_{(p.c,ts)\in\mathbf{rtpcs\_varf_{r,t,p,c,ts}}}\ \sum_{v\in\mathbf{rtp\_vintyr_{r,v,t,p}}}\ \sum_{ie\in\mathbf{rpc\_ire_{r,p,c,ie}}}$$

$$\left[\begin{array}{l} VAR\_IRE_{r,v,t,p,c,ts,ie} \times UC\_IRE_{uc\_n,'LHS',r,t,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\ \times \\ \sum_{cur\in\mathbf{rdcur}_{r,cur}} \left(\begin{array}{l} OBJ\_FCOST_{r,t,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','COST'} \\ + \\ OBJ\_FDELV_{r,t,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','DELIV'} \\ - \\ OBJ\_FSUB_{r,t,p,c,s,cur}\ \ if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','SUB'} \\ + \\ OBJ\_FTAX_{r,t,p,c,s,cur}\ \ if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','IRE','TAX'} \end{array} \right) \\ \times \left(UC\_IRE_{uc\_n,'LHS',r,t,p,c,ts,ie} \right)^{M(t+1)-M(t)-1} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','IRE','GROWTH'}\ is\ given \end{array}\right]$$

$$COMPRD\_GROW_{r,t,s,'LHS'}$$

$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc_{r,t,c,ts}}} \left(\begin{array}{l} VAR\_COMPRD_{r,t,c,ts} \times UC\_COMPRD_{uc\_n,'LHS',r,t,c,s} \\ \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMPRD',uc\_cost} \right) \\ \times \left(UC\_COMPRD_{uc\_n,'LHS',r,t,c,s} \right)^{M(t+1)-M(t)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMPRD','GROWTH'}\ given \end{array} \right)$$

$$COMNET\_GROW_{r,t,s,'LHS'}$$

$$= \sum_{(c,ts)\in \mathbf{rtcs\_varc_{r,t,c,ts}}} \left(\begin{array}{l} VAR\_COMNET_{r,t,c,ts} \times UC\_COMNET_{uc\_n,'LHS',r,t,c,s} \\ \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMNT_{r,t,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMNET',uc\_cost} \right) \\ \times \left(UC\_COMNET_{uc\_n,'LHS',r,t,c,s} \right)^{M(t+1)-M(t)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'LHS','COMNET','GROWTH'}\ given \end{array} \right)$$

$$NCAP\_GROW_{r,t,p,'LHS'}$$

$$=$$

$$\sum_{p} \left[\begin{array}{l} VAR\_NCAP_{r,t,p} \times UC\_NCAP_{uc\_n,'LHS',r,t,p} \times \\ \left(\begin{array}{l} \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','COST'}\ is\ given \\ - \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','SUB'}\ is\ given \\ + \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','TAX'}\ is\ given \end{array} \right) \\ \times \left(UC\_NCAP_{uc\_n,'LHS',r,t,p} \right)^{M(t+1)-M(t)-1} \quad if\ UC\_ATTR_{r,uc\_n,'LHS','NCAP','GROWTH'}\ given \end{array} \right]$$

$$
\begin{aligned}
&CAP\_GROW_{r,t,'LHS'} \\
&= \\
&\sum_{p}\left(\begin{array}{l}
VAR\_CAP_{r,t,p} \times UC\_CAP_{uc\_n,'LHS',r,t,p} \\
\times \\
PRC\_ACTFLO_{r,p} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','CAP','CAPACT'} \text{ is given} \\
\times \\
\left(UC\_CAP_{uc\_n,'LHS',r,t,p}\right)^{M(t+1)-M(t)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'LHS','CAP','GROWTH'} \text{ is given}
\end{array}\right)
\end{aligned}
$$

The expressions for the variable terms on the RHS can be written as follows:

$$
\begin{aligned}
&ACT\_GROW_{r,t+1,s,'RHS'} \\
&= \\
&\sum_{(p,v)\in \mathbf{rtp\_vintyr_{r,v,t+1,p}}} \; \sum_{ts\in \mathbf{prc\_ts_{r,p,ts}}} \left(\begin{array}{l}
VAR\_ACT_{r,v,t+1,p,ts} \times UC\_ACT_{uc\_n,'RHS',r,t+1,p,ts} \times \left(RS\_FR_{r,s,ts}\right) \\
\times \\
\left(\sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ACOST_{r,t+1,p,cur}\right) \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','ACT','COST'} \text{ is given} \\
\times \\
\left(UC\_ACT_{uc\_n,'RHS',r,t+1,p,ts}\right)^{M(t)-M(t+1)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','ACT','GROWTH'} \text{ is given}
\end{array}\right)
\end{aligned}
$$

$$FLO\_GROW_{r,t+1,s,'RHS'}$$

$$=$$

$$\sum_{(p,c,ts)\in\mathbf{rtpcs\_varf_{r,t+1,p,c,ts}}} \sum_{v\in\mathbf{rtp\_vintyr_{r,v,t+1,p}}}$$

$$\left\{\begin{array}{l}
VAR\_FLO_{r,v,t+1,p,c,ts} \times UC\_FLO_{uc\_n,'RHS',r,t+1,p,c,ts} \times \left(RTCS\_TSFR_{r,t,c,s,ts} \right) \\
\times \\
\left[\sum_{cur\in\mathbf{rdcur_{r,cur}}} \left(\begin{array}{l}
OBJ\_FCOST_{r,t+1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','COST'}\ is\ given \\
+ \\
OBJ\_FDELV_{r,t+1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','DELIV'}\ is\ given \\
+ \\
OBJ\_FSUB_{r,t+1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','SUB'}\ is\ given \\
+ \\
OBJ\_FTAX_{r,t+1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','TAX'}\ is\ given
\end{array} \right) \right] \\
\times \\
\left(UC\_FLO_{uc\_n,'RHS',r,t+1,p,c,ts}\right)^{M(t)-M(t+1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','GROWTH'}\ is\ given
\end{array}\right\}$$

$$IRE\_GROW_{r,t+1,s,'RHS'}$$

$$= \sum_{(p,c,ts)\in\mathbf{rtpcs\_varf_{r,t+1,p,c,ts}}} \sum_{v\in\mathbf{rtp\_vintyr_{r,v,t+1,p}}} \sum_{ie\in\mathbf{rpc\_ire_{r,p,c,ie}}}$$

$$\left[\begin{array}{l}
VAR\_IRE_{r,v,t+1,p,c,ts,ie} \times UC\_IRE_{uc\_n,'RHS',r,t+1,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t+1,c,s,ts} \right) \\
\times \\
\sum_{cur\in\mathbf{rdcur}_{r,cur}} \left(\begin{array}{l}
OBJ\_FCOST_{r,t+1,p,c,s,cur} \ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','COST'} \\
+ \\
OBJ\_FDELV_{r,t+1,p,c,s,cur} \ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','DELIV'} \\
- \\
OBJ\_FSUB_{r,t+1,p,c,s,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','SUB'} \\
+ \\
OBJ\_FTAX_{r,t+1,p,c,s,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','TAX'}
\end{array} \right) \\
\times \left(UC\_IRE_{uc\_n,'RHS',r,t+1,p,c,ts,ie}\right)^{M(t)-M(t+1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','GROWTH'}\ given
\end{array}\right]$$

$$
\begin{array}{l}
COMPRD\_GROW_{r,t+1,s,'RHS'} \\
= \sum\limits_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t,+1c,ts}}} \left(\begin{array}{l}
VAR\_COMPRD_{r,t+1,c,ts} \times UC\_COMPRD_{uc\_n,'RHS',r,t+1,c,s} \\
\times \left(RTCS\_TSFR_{r,t+1,c,s,ts} \right) \times \\
\sum\limits_{\substack{cur \in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t+1,c,ts,uc\_\cos t,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD',uc\_cost} \right) \\
\times \left(UC\_COMPRD_{uc\_n,'RHS',r,t+1,c,s} \right)^{M(t)-M(t+1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD','GROWTH'}\ given
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
COMNET\_GROW_{r,t+1,s,'RHS'} \\
= \sum\limits_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t+1,c,ts}}} \left(\begin{array}{l}
VAR\_COMNET_{r,t+1,c,ts} \times UC\_COMNET_{uc\_n,'RHS',r,t+1,c,s} \\
\times \left(RTCS\_TSFR_{r,t+1,c,s,ts} \right) \times \\
\sum\limits_{\substack{cur \in \mathbf{rdcur}_{r,cur} \\ uc\_\cos t}} \left(OBJ\_COMNT_{r,t+1,c,ts,uc\_\cos t,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET',uc\_\cos t} \right) \\
\times \left(UC\_COMNET_{uc\_n,'RHS',r,t+1,c,s} \right)^{M(t)-M(t+1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET','GROWTH'}\ given
\end{array} \right)
\end{array}
$$

$$
\begin{array}{l}
NCAP\_GROW_{r,t+1,'RHS'} \\
= \\
\sum\limits_{p} \left(\begin{array}{l}
VAR\_NCAP_{r,t+1,p} \times UC\_NCAP_{uc\_n,'RHS',r,t+1,p} \\
\times \\
\left(\begin{array}{l}
\sum\limits_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t+1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','COST'}\ is\ given \\
- \\
\sum\limits_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t+1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','SUB'}\ is\ given \\
+ \\
\sum\limits_{cur \in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t+1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','TAX'}\ is\ given
\end{array} \right) \\
\times \\
\left(UC\_NCAP_{uc\_n,'RHS',r,t+1,p} \right)^{M(t)-M(t+1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','GROWTH'}\ is\ given
\end{array} \right)
\end{array}
$$

$$CAP\_GROW_{r,t+1,'RHS'}$$
$$=$$
$$\sum_{p}\left(\begin{array}{l} VAR\_CAP_{r,t+1,p} \times UC\_CAP_{uc\_n,'RHS',r,t+1,p} \\ \times \\ PRC\_CAPACT_{r,p} \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','CAP','CAPACT'} \text{ is given} \\ \times \\ \left(UC\_CAP_{uc\_n,'RHS',r,t+1,p}\right)^{M(t)-M(t+1)-1} \quad \text{if } UC\_ATTR_{r,uc\_n,'RHS','CAP','GROWTH'} \text{ is given} \end{array}\right)$$

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.

$$=$$

$$VAR\_UCT_{uc\_n,t}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \left(\begin{array}{l} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{array} \right)$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'} \right)$$

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:

$$\{\leq;=;\geq\}$$
$$UC\_RHSRT_{r,uc\_n,t,l}$$
$$+$$
$$\sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'}\right)$$

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.

$$=$$

$$VAR\_UCRT_{r,uc\_n,t}$$

$$+$$

$$\sum_{s \in \mathbf{uc\_ts\_sum}} \left(\begin{array}{l} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{array} \right)$$

$$+$$

$$\left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'} \right)$$

with

$$VAR\_UCRT.LO_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'LO'}$$

$$VAR\_UCRT.UP_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'UP'}$$

$$VAR\_UCRT.FX_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'FX'}$$

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:

$$EQ(l)\_UCRSUS_{r,uc\_n,t,s} \ni UC\_RHSRTS_{r,uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each_{r,uc\_n}}$$
$$\wedge\, \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$
$$\begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'}\right)$$

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

$$VAR\_UCRTS.LO_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'LO'}$$

$$VAR\_UCRTS.UP_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'UP'}$$

$$VAR\_UCRTS.FX_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'FX'}$$

6.4.4.4 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:

$$EQ(l)\_UCSUS_{uc\_n,t,s} \ni UC\_RHSTS_{uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum_{r,uc\_n}}$$
$$\wedge \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'} \right)$$

When control parameter VAR_UC=NO:

$$\{\le ; = ; \ge\}$$
$$UC\_RHSTS_{uc\_n,t,s,l}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'} \right)$$

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.

$$=$$

$$VAR\_UCTS_{uc\_n,t,s}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(\begin{array}{l} ACT\_GROW_{r,t+1,s,'RHS'} + FLO\_GROW_{r,t+1,s,'RHS'} + IRE\_GROW_{r,t+1,s,'RHS'} \\ + COMNET\_GROW_{r,t+1,s,'RHS'} + COMPRD\_GROW_{r,t+1,s,'RHS'} \end{array} \right)$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t+1,'RHS'} + CAP\_GROW_{r,t+1,'RHS'} \right)$$

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:

$$ACT\_GROW_{r,t-1,s,'RHS'}$$

$$=$$

$$\sum_{(p,v)\in\mathbf{rtp\_vintyr_{r,v,t-1,p}}}\ \sum_{ts\in\mathbf{prc\_ts_{r,p,ts}}}\left(\begin{array}{l} VAR\_ACT_{r,v,t-1,p,ts}\times UC\_ACT_{uc\_n,'RHS',r,t-1,p,ts}\times\left(RS\_FR_{r,s,ts}\right) \\ \times \\ \left(\sum\limits_{cur\in\mathbf{rdcur_{r,cur}}} OBJ\_ACOST_{r,t-1,p,cur}\right) \quad if\ UC\_ATTR_{r,uc\_n,'RHS','ACT','COST'}\ is\ given \\ \times \\ \left(UC\_ACT_{uc\_n,'RHS',r,t-1,p,ts}\right)^{M(t)-M(t-1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','ACT','GROWTH'}\ is\ given \end{array}\right)$$

$$FLO\_GROW_{r,t-1,s,'RHS'}$$

$$=$$

$$\sum_{(p,c,ts)\in \mathbf{rtpcs\_varf_{r,t-1,p,c,ts}}} \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t-1,p}}}$$

$$\left\{ \begin{array}{l} VAR\_FLO_{r,v,t-1,p,c,ts} \times UC\_FLO_{uc\_n,'RHS',r,t-1,p,c,ts} \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \\ \times \\ \left[\sum_{cur\in \mathbf{rdcur_{r,cur}}} \begin{pmatrix} OBJ\_FCOST_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','COST'}\ is\ given \\ + \\ OBJ\_FDELV_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','DELIV'}\ is\ given \\ + \\ OBJ\_FSUB_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','SUB'}\ is\ given \\ + \\ OBJ\_FTAX_{r,t-1,p,c,ts,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','TAX'}\ is\ given \end{pmatrix} \right] \\ \times \\ \left(UC\_FLO_{uc\_n,'RHS',r,t-1,p,c,ts} \right)^{M(t)-M(t-1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','FLO','GROWTH'}\ is\ given \end{array} \right\}$$

$$IRE\_GROW_{r,t-1,s,'RHS'}$$

$$= \sum_{(p,c,ts)\in \mathbf{rtpcs\_varf_{r,t-1,p,c,ts}}} \sum_{v\in \mathbf{rtp\_vintyr_{r,v,t-1,p}}} \sum_{ie\in \mathbf{rpc\_ire_{r,p,c,ie}}}$$

$$\left[\begin{array}{l} VAR\_IRE_{r,v,t-1,p,c,ts,ie} \times UC\_IRE_{uc\_n,'RHS',r,t-1,p,c,ts,ie} \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \\ \times \\ \sum_{cur\in \mathbf{rdcur}_{r,cur}} \begin{pmatrix} OBJ\_FCOST_{r,t-1,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','COST'} \\ + \\ OBJ\_FDELV_{r,t-1,p,c,s,cur}\ if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','DELIV'} \\ - \\ OBJ\_FSUB_{r,t-1,p,c,s,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','SUB'} \\ + \\ OBJ\_FTAX_{r,t-1,p,c,s,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','TAX'} \end{pmatrix} \\ \times \left(UC\_IRE_{uc\_n,'RHS',r,t-1,p,c,ts,ie} \right)^{M(t)-M(t-1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','IRE','GROWTH'}\ given \end{array} \right]$$

$$COMPRD\_GROW_{r,t-1,s,'RHS'} = \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t,-1c,ts}}} \begin{pmatrix} VAR\_COMPRD_{r,t-1,c,ts} \times UC\_COMPRD_{uc\_n,'RHS',r,t-1,c,s} \\ \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMPD_{r,t-1,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD',uc\_cost} \right) \\ \times \left(UC\_COMPRD_{uc\_n,'RHS',r,t-1,c,s} \right)^{M(t)-M(t-1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMPRD','GROWTH'}\ given \end{pmatrix}$$

$$COMNET\_GROW_{r,t-1,s,'RHS'} = \sum_{(c,ts)\in \mathbf{rtcs\_varc}_{\mathbf{r,t-1,c,ts}}} \begin{pmatrix} VAR\_COMNET_{r,t-1,c,ts} \times UC\_COMNET_{uc\_n,'RHS',r,t-1,c,s} \\ \times \left(RTCS\_TSFR_{r,t-1,c,s,ts} \right) \times \\ \sum_{\substack{cur\in \mathbf{rdcur}_{r,cur} \\ uc\_cost}} \left(OBJ\_COMNT_{r,t-1,c,ts,uc\_cost,cur} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET',uc\_cost} \right) \\ \times \left(UC\_COMNET_{uc\_n,'RHS',r,t-1,c,s} \right)^{M(t)-M(t-1)-1} \quad if\ \mathbf{uc\_attr}_{r,uc\_n,'RHS','COMNET','GROWTH'}\ given \end{pmatrix}$$

$$NCAP\_GROW_{r,t-1,'RHS'} = \sum_{p} \begin{pmatrix} VAR\_NCAP_{r,t-1,p} \times UC\_NCAP_{uc\_n,'RHS',r,t-1,p} \\ \times \\ \begin{pmatrix} \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ICOST_{r,t-1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','COST'}\ is\ given \\ - \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ISUB_{r,t-1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','SUB'}\ is\ given \\ + \\ \sum_{cur\in \mathbf{rdcur_{r,cur}}} OBJ\_ITAX_{r,t-1,p,cur} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','TAX'}\ is\ given \end{pmatrix} \\ \times \\ \left(UC\_NCAP_{uc\_n,'RHS',r,t-1,p} \right)^{M(t)-M(t-1)-1} \quad if\ UC\_ATTR_{r,uc\_n,'RHS','NCAP','GROWTH'}\ is\ given \end{pmatrix}$$

$$CAP\_GROW_{r,t-1,'RHS'}$$

$$=$$

$$\sum_{p}\left(\begin{array}{l} VAR\_CAP_{r,t-1,p} \times UC\_CAP_{uc\_n,'RHS',r,t-1,p} \\ \times \\ PRC\_CAPACT_{r,p} \quad \textit{if } UC\_ATTR_{r,uc\_n,'RHS','CAP','CAPACT'} \textit{ is given} \\ \times \\ \left(UC\_CAP_{uc\_n,'RHS',r,t-1,p}\right)^{M(t+1)-M(t)-1} \quad \textit{if } UC\_ATTR_{r,uc\_n,'RHS','CAP','GROWTH'} \textit{ is given} \end{array}\right)$$

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:

$$EQ(l)\_UCSU_{uc\_n,t} \ni UC\_RHST_{uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum_{r,uc\_n}}$$

$$\wedge\ \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'} \right)$$

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.

$$=$$

$$VAR\_UCT_{uc\_n,t}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \sum_{s \in \mathbf{uc\_ts\_sum}} \left(\begin{array}{l} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{array} \right)$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'} \right)$$

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:

$$EQ(l)\_UCRSU_{r,uc\_n,t} \ni UC\_RHSRT_{r,uc\_n,t,bd} \wedge \mathbf{uc\_ts\_sum_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_each_{r,uc\_n}}$$

$$\wedge\, \mathbf{uc\_t\_succ_{r,uc\_n,t}}$$

$$\sum_{s \in \mathbf{uc\_ts\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'}\right)$$

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.

$$=$$

$$VAR\_UCRT_{r,uc\_n,t}$$

$$+$$

$$\sum_{s \in \mathbf{uc\_ts\_sum}} \left(\begin{array}{l} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{array} \right)$$

$$+$$

$$\left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'} \right)$$

with

$$VAR\_UCRT.LO_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'LO'}$$

$$VAR\_UCRT.UP_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'UP'}$$

$$VAR\_UCRT.FX_{r,uc\_n,t} = UC\_RHSRT_{r,uc\_n,t,'FX'}$$

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:

$$\{\leq;=;\geq\}$$
$$UC\_RHSRTS_{r,uc\_n,t,s,l}$$
$$+$$
$$\begin{pmatrix} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{pmatrix}$$
$$+$$
$$\left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'}\right)$$

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.

$$=$$

$$VAR\_UCRTS_{r,uc\_n,t,s}$$

$$+$$

$$\begin{pmatrix} ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{pmatrix}$$

$$+$$

$$\left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'}\right)$$

with

$$VAR\_UCRTS.LO_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'LO'}$$

$$VAR\_UCRTS.UP_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'UP'}$$

$$VAR\_UCRTS.FX_{r,uc\_n,t,s} = UC\_RHSRTS_{r,uc\_n,t,s,'FX'}$$

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:

$$EQ(l)\_UCSUS_{uc\_n,t,s} \ni UC\_RHSTS_{uc\_n,t,s,bd} \wedge \mathbf{uc\_ts\_each_{r,uc\_n,s}} \wedge \mathbf{uc\_r\_sum_{r,uc\_n}}$$
$$\wedge\, \mathbf{úc\_t\_succ_{r,uc\_n,t}}$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \begin{pmatrix} ACT\_GROW_{r,t,s,'LHS'} + FLO\_GROW_{r,t,s,'LHS'} + IRE\_GROW_{r,t,s,'LHS'} \\ + COMNET\_GROW_{r,t,s,'LHS'} + COMPRD\_GROW_{r,t,s,'LHS'} \end{pmatrix}$$
$$+$$
$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t,'LHS'} + CAP\_GROW_{r,t,'LHS'} \right)$$

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.

$$=$$

$$VAR\_UCTS_{uc\_n,t,s}$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(\begin{aligned} & ACT\_GROW_{r,t-1,s,'RHS'} + FLO\_GROW_{r,t-1,s,'RHS'} + IRE\_GROW_{r,t-1,s,'RHS'} \\ & + COMNET\_GROW_{r,t-1,s,'RHS'} + COMPRD\_GROW_{r,t-1,s,'RHS'} \end{aligned} \right)$$

$$+$$

$$\sum_{r \in \mathbf{uc\_r\_sum}} \left(NCAP\_GROW_{r,t-1,'RHS'} + CAP\_GROW_{r,t-1,'RHS'} \right)$$

with

$$VAR\_UCTS.LO_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'LO'}$$

$$VAR\_UCTS.UP_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'UP'}$$

$$VAR\_UCTS.FX_{uc\_n,t,s} = UC\_RHSTS_{uc\_n,t,s,'FX'}$$

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 $tt \leq t$ 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
BUILDUP	Divide by the lead time from previous milestone year to current, for getting the annual build-up of capacity	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
PERDISC	Multiply by the NPV of period to get a discounted value	ACT,FLO,IRE, COM*
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)
<TSLVL>	Defines the timeslice level of the constraint, equivalent to UC_TSL(r,uc_n,side,tslvl). Works only under VEDA.	All
YES	Declares the constraint to be dynamic, of type (t–1, t)	All (RHS only)

6.4.6.4 BUILDUP modifier

The BUILDUP modifier can only be applied only to the NCAP terms of user constraints, when such exist. The NCAP term for each process is divided by the lead time (LEAD(T)) of the milestone year T of the NCAP variable, when the modifier is used, to get the annual build-up of the new capacity.

6.4.6.5 CAPACT modifier

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.

6.4.6.6 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.

6.4.6.7 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**.

6.4.6.8 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).

6.4.6.9 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.

6.4.6.10 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.

6.4.6.11 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.

6.4.6.12 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.

6.4.6.13 PERDISC modifier

The PERDISC modifier can be used for adding the period-wise discounting multiplier for components related to activities and flows. As a result, the component term will represent the discounted amount for each period, and so if a cost multiplier is also applied, the constraint component will represent the discounted costs related to activities or flows.

6.4.6.14 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.

6.4.6.15 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**).

6.4.6.16 <TSLVL> modifier

Timeslice levels can also be used as a modifier for any component, when the constraint is meant to be timeslice-dynamic. It is an alternative way to set the UC_TSL(r,uc_n,side,tslvl) attribute. Note that even though the modifier can be used with any component, it is important that only a single tslvl level is specified on a single side for the constraint where this modifier is used, otherwise the resulting constraint may be generated in unexpected ways not designed to be supported.

6.4.6.17 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 $SET 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:

$$M_{atm}(y) = E(y) + (1 - \varphi_{atm\text{-}up})\, M_{atm}(y-1) + \varphi_{up\text{-}atm}\, M_{up}(y-1) \qquad (1)$$
$$M_{up}(y) = (1 - \varphi_{up\text{-}atm} - \varphi_{up\text{-}lo})\, M_{up}(y-1) + \varphi_{atm\text{-}up}\, M_{atm}(y-1) + \varphi_{lo\text{-}up}\, M_{lo}(y-1) \qquad (2)$$
$$M_{lo}(y) = (1 - \varphi_{lo\text{-}up})\, M_{lo}(y-1) + \varphi_{up\text{-}lo}\, M_{up}(y-1) \qquad (3)$$

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):

$$CH4_{atm}(y) = (1 - \Phi_{CH4}) \cdot CH4_{atm}(y-1) + EA_{CH4}(y) \qquad (1a)$$
$$CH4_{up}(y) = CH4_{up}(y-1) \qquad (1b)$$
$$CH4_{tot}(y) = CH4_{atm}(y) + CH4_{up}(y) \qquad (1c)$$

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.

$$\Delta F_{CO2}(y) = \gamma \times \frac{\ln(M_{atm}(y)/M_0)}{\ln 2} \tag{4a}$$

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^{o}C to 10^{o}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 ^{o}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/^{o}C). 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/^{o}C).
- 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 (^{o}C).
- 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: ^{o}C

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) = 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:

$$EQ\_CLITOT_{cm\_tkind,t} \ \forall \left[\left(t \in \mathbf{milestonyr} \right) \right]$$

$$\sum_{\substack{cm\_tkind \in \mathbf{cm\_emis} \\ (r,c,s) \in \mathbf{rtcs\_varc_{r,c,t,s}}}} VAR\_COMNET_{r,t,c,s} \times CM\_GHGMAP_{r,c,cm\_tkind}$$

$$\sum_{\mathbf{cm\_emis_{cm\_tkind}}} \left(\begin{array}{l} CM\_LINFOR_{t,cm\_emis,'N'} \times \\ \left(\sum_{\substack{\mathbf{cm\_atbox_{cm\_emis,cm\_box}} \\ \mathbf{cm\_boxmap_{cm\_emis,cm\_var,cm\_box}}}} VAR\_CLIBOX_{cm\_var} \right) \\ CM\_LINFOR_{t,cm\_emis,'FX'} \end{array} \right) +$$

$$+ CM\_EXOFORC_t$$

$$\{=\}$$

$$VAR\_CLITOT_{cm\_tkind,t}$$

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:

$$EQ\_CLICONC_{cm\_emis,cm\_box,t} \quad \forall\left[\left(t \in \mathbf{milestonyr}\right)\right]$$

$$\sum_{\mathbf{cm\_boxmap}_{\mathbf{cm\_emis,cm\_var,cm\_box2}}} VAR\_CLIBOX_{cm\_var,t-1} \times CM\_AA_{cm\_emis,t,cm\_box,cm\_box2} +$$

$$CM\_BB_{cm\_emis,t,cm\_box} \times VAR\_CLITOT_{cm\_emis,t} +$$

$$CM\_CC_{cm\_emis,t,cm\_box} \times VAR\_CLITOT_{cm\_emis,t-1} +$$

$$\sum_{\substack{\mathbf{miyr\_1}_t \\ \mathbf{cm\_boxmap}_{\mathbf{cm\_emis,cm\_var,cm\_box2}}}} CM\_CONST_{cm\_var} \times CM\_AA_{cm\_emis,t,cm\_box,cm\_box2}$$

$$\{=\}$$

$$\sum_{\mathbf{cm\_boxmap}_{\mathbf{cm\_emis,cm\_var,cm\_box}}} VAR\_CLIBOX_{cm\_var,t}$$

$CM\_AA_{cm\_emis,t,i,j} = \{A_{ij}(t)\} = PHI^{n(t)}$ $(PHI^0 = I)$, where

PHI is the 3×3 matrix : $\begin{bmatrix} (1-PHI\_AT\_UP) & PHI\_UP\_AT & 0 \\ PHI\_AT\_UP & (1-PHI\_UP\_AT-PHI\_UP\_LO) & PHI\_LO\_UP \\ 0 & PHI\_UP\_LO & (1-PHI\_LO\_UP) \end{bmatrix}$

$CM\_BB_{cm\_emis,t,i} = \{BB_{i1}(t)\}$ is the first column of the matrix :

$$BB(t) = \sum_{i=0}^{p(t)-1} PHI^i \qquad if\ \ p(t) \geq 1$$

$$BB(t) = 0 \qquad if\ \ p(t) = 0$$

$CM\_CC_{cm\_emis,t,i} = \{CC_{i1}(t)\}$ 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 :

$$CC(t) = SIGMA1 \times \sum_{i=0}^{n(t)-1} \frac{i}{n(t)} \times PHI^i$$

$$n(t) = m(t) - m(t-1) \qquad if\ t \neq 1,$$

$$n(t) = m(t) - Y \qquad if\ t = 1$$

$D(t)$ 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 $y = \alpha_y$ (m(t)–y) + β_y (y–m(t–1)), for all y in the range m(t–1) < y ≤ 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)

$$\sum_{\substack{\textbf{cm\_atbox}_{\textbf{cm\_emis,cm\_box}} \\ \textbf{cm\_boxmap}_{\textbf{cm\_emis,cm\_var,cm\_box}}}} \alpha_y \times VAR\_CLIBOX_{cm\_var,t-1} + \beta_y \times VAR\_CLIBOX_{cm\_var,t}$$

$$\leq CM\_MAXC_{y,cm\_\mathrm{var}}$$

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):

$$\sum_{\textbf{cm\_boxmap}_{\textbf{'FORCING',cm\_var,'ATM'}}} \alpha_y \times VAR\_CLIBOX_{cm\_var,t-1} + \beta_y \times VAR\_CLIBOX_{cm\_var,t}$$

$$\leq CM\_MAXC_{y,cm\_\mathrm{var}}$$

Case E. For atmospheric GHG concentrations, beyond m(T):

$$\sum_{\substack{\textbf{cm\_atbox}_{\textbf{cm\_emis,cm\_box}} \\ \textbf{cm\_boxmap}_{\textbf{cm\_emis,cm\_var,cm\_box}}}} VAR\_CLIBOX_{cm\_var,y} \quad \leq \quad CM\_MAXC_{y,cm\_\mathrm{var}}$$

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):

$$\sum_{\textbf{cm\_boxmap}_{\textbf{'FORCING',cm\_var,'ATM'}}} VAR\_CLIBOX_{cm\_var,y} \quad \leq \quad CM\_MAXC_{y,cm\_\mathrm{var}}$$

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:

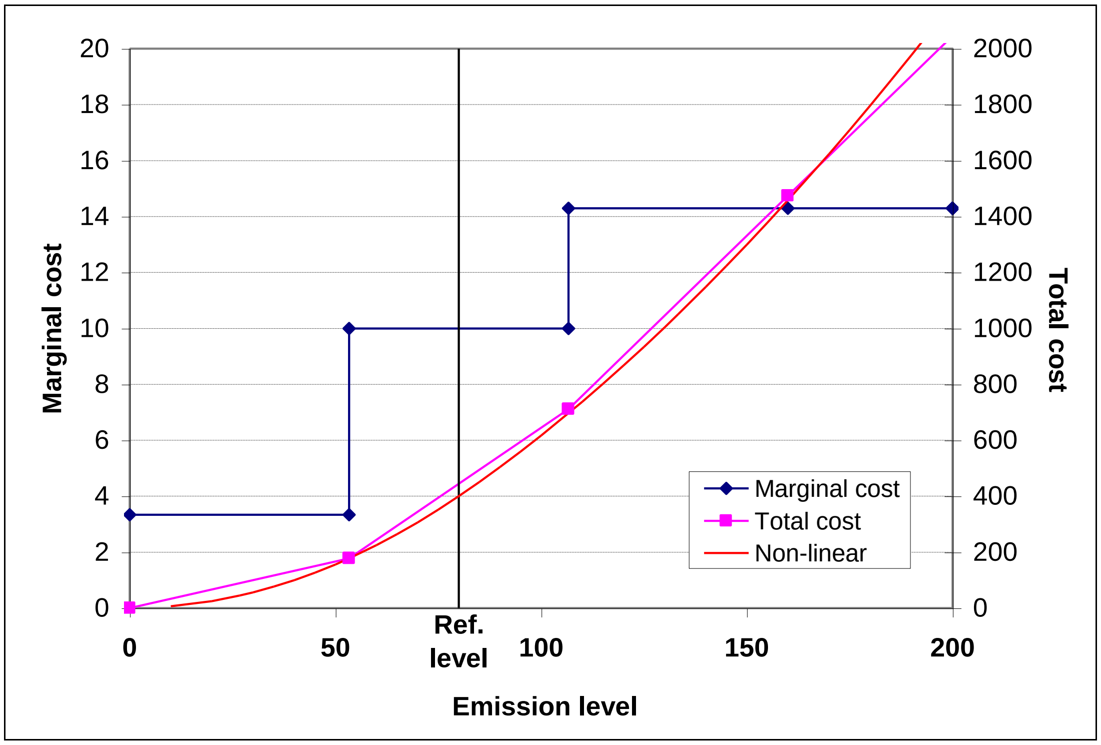


Figure 15. Example of a linearized damage function with 1+1+1 steps (1 lower step, 1 middle step, 1 upper step).

- 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.

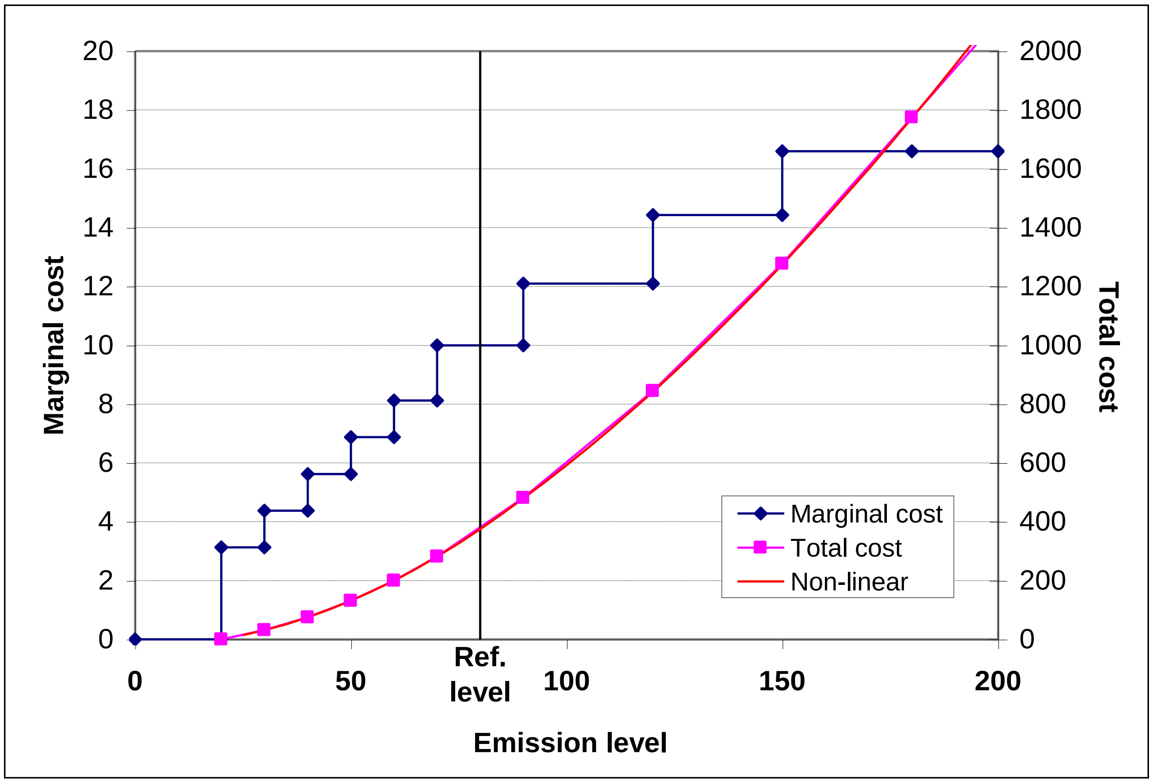


Figure B-16. Example of a linearized damage function with 1+5+1+3 steps (one zero cost step, 5 lower steps, one middle step, 3 upper steps).

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:

$$EQ\_DAMAGE_{r,t,c} \quad \ni \left(\mathbf{rtc}_{r,t,c} \wedge \exists(cur): DAM\_COST_{r,t,c,cur}\right)$$

$$\sum_{(jj,bd)\in \mathbf{dam\_num}_{r,c,jj,bd}} \sum_{j\le jj} VAR\_DAM_{r,t,c,bd,j}$$

$$\{=\}$$

$$\sum_{\mathbf{com\_ts_{r,c,ts}}} \left(\begin{array}{l} DAM\_COEF_{r,t,c,ts} \times \\ \left(\begin{array}{ll} VAR\_COMNET_{r,t,c,ts} & if\ DAM\_ELAST_{r,c,'N'}\ not\ given \\ VAR\_COMPRD_{r,t,c,ts} & otherwise \end{array} \right) \end{array} \right)$$

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

$$\sum_{(t,c)\in\left\{\mathbf{rtc_{r,t,c}} \mid (DAM\_COST_{r,t,} > 0)\right\}} DAM\_COST_{r,t,c,cur} \times OBJ\_PVT_{r,t,cur} \times$$

$$\left[\begin{array}{l} \displaystyle\sum_{\substack{jj\in\mathbf{dam\_num}_{r,c,jj,'LO'} \\ j\le jj}} \left(\begin{array}{l} \dfrac{VAR\_DAM_{r,t,c,'LO',j}}{DAM\_BQTY_{r,c}^{\;DAM\_ELAST_{r,c,'LO'}}} \times \\ \left(\begin{array}{l} DAM\_BQTY_{r,c} - DAM\_VOC_{r,c,'LO'} + \\ DAM\_SIZE_{r,c,'LO'} \times (j-0.5) \end{array} \right)^{DAM\_ELAST_{r,c,'LO'}} \end{array} \right) + \\ VAR\_DAM_{r,t,c,'FX',1} \\ \displaystyle\sum_{\substack{jj\in\mathbf{dam\_num}_{r,c,jj,'UP'} \\ j\le ORD(jj)}} \left(\begin{array}{l} \dfrac{VAR\_DAM_{r,t,c,'UP',j}}{DAM\_BQTY_{r,c}^{\;DAM\_ELAST_{r,c,'UP'}}} \times \\ \left(\begin{array}{l} DAM\_BQTY_{r,c} + \dfrac{DAM\_SIZE_{r,c,'FX'}}{2} + \\ DAM\_SIZE_{r,c,'UP'} \times (j-0.5) \end{array} \right)^{DAM\_ELAST_{r,c,'UP'}} \end{array} \right) \end{array}\right]$$

$$\{=\}$$

$$VAR\_OBJ_{r,'OBJDAM',cur}$$

Case B: Non-linear functions

$$\sum_{(t,c)\in\{\mathbf{rtc_{r,t,c}}|(DAM\_COST_{r,t,}>0)\}} DAM\_COST_{r,t,c,cur} \times OBJ\_PVT_{r,t,cur} \times$$

$$\left[\frac{\left(\begin{pmatrix} VAR\_DAM_{r,t,c,'LO',j} + \\ DAM\_BQTY_{r,c} - DAM\_VOC_{r,c,'LO'} \end{pmatrix}^{\left(DAM\_ELAST_{r,c,'LO'}+1\right)} - \\ \left(DAM\_BQTY_{r,c} - DAM\_VOC_{r,c,'LO'}\right)^{\left(DAM\_ELAST_{r,c,'LO'}+1\right)}\right)}{DAM\_BQTY_{r,c}{}^{DAM\_ELAST_{r,c,'LO'}} \times \left(DAM\_ELAST_{r,c,'LO'}+1\right)} + \frac{\left(\left(VAR\_DAM_{r,t,c,'UP',j} + DAM\_BQTY_{r,c}\right)^{\left(DAM\_ELAST_{r,c,'UP'}+1\right)} - \\ \left(DAM\_BQTY_{r,c}\right)^{\left(DAM\_ELAST_{r,c,'UP'}+1\right)}\right)}{DAM\_BQTY_{r,c}{}^{DAM\_ELAST_{r,c,'UP'}} \times \left(DAM\_ELAST_{r,c,'UP'}+1\right)}\right]$$

$$\{=\}$$

$$VAR\_OBJ_{r,'OBJDAM',cur}$$

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 kth 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 kth 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 kth 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 ∈ 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.

$$\sum_{\substack{p'\$\left(CLUSTER_{r,p,p'}>0\right)\wedge \\ (r,t,p'\in rtp)}} (CLUSTER_{r,p,p'} * VAR\_NCAP_{r,t,p'})$$

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

$$EQ\_COS_{r,t,p} \ \forall\left[(p \in teg) \wedge \left((r,t,p) \in rtp\right)\right]$$

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 kth interval.

$$\sum_{k}(ALPH_{k,p} * VAR\_DELTA_{r,t,p,k} + BETA_{k,p} * VAR\_LAMBD_{r,t,p,k})$$

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

$$EQ\_EXPE1_{r,t,p,k} \ \forall\left[(p \in teg) \wedge ((r,t,p) \in rtp) \wedge (t < TLAST)\right]$$

Sum over intervals j ≤ k of binary variables VAR_DELTA(r,t,p,j), for the kth 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 kth 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

$$EQ\_EXPE2_{r,t,p,k} \ \forall\left[(p \in teg) \wedge ((r,t,p) \in rtp) \wedge (t < TLAST)\right]$$

Sum over intervals j ≥ k of binary variables VAR_DELTA(r,t,p,j), for the kth interval, in period t.

$$\sum_{j \ge k} (VAR\_DELTA_{r,t,p,j})$$

$$\{\le\}$$

Sum over intervals j ≥ k of binary variables VAR_DELTA(r,t,p,j), for the kth interval, in period t+1.

$$\sum_{j \ge k} (VAR\_DELTA_{r,t+1,p,j})$$

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 kth 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

$$EQ\_LA1_{r,t,p,k}\ \forall\left[(p \in teg) \wedge ((r,t,p) \in rtp)\right]$$

Portion of the cumulative investment in capacity that lies in the kth interval (of the piecewise linear approximation of the cumulative cost curve), in the current period.

$$VAR\_LAMBD_{r,t,p,k}$$

$$\{\geq\}$$

Left hand end of the kth 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 kth 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

$$MR\_LA2_{r,t,p,k} \;\forall\left[(p \in teg) \wedge ((r,t,p) \in rtp)\right]$$

Portion of the cumulative investment in capacity that lies in the kth 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 kth 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.

$$+ \sum_{t,p \in teg} \left[DISC_{r,t,p} * VAR\_IC_{r,t,p} \right]$$

Appendix D TIMES Demand Functions

1 Introduction

As discussed in Chapters 3 and 4 of Part I, in TIMES the standard Demand Function formulation includes only sensitive of the demands to their own prices, modeled through a linearized formulation of the price elasticities. Until TIMES v4.0, only the linearized own-price elasticity formulation was available in the common code. In MARKAL, the corresponding non-linear formulation was also available (see Loulou & al. 2004), and it was therefore subsequently made available in TIMES v4.1 and above, as the first natural generalization of the original demand functions.

When substitution possibilities are to be modeled, demand functions involving Constant Elasticity of Substitution (CES) aggregates are very commonly used in economic models integrating engineering and bio-physical properties. Hence, the possibility to use CES-based demand functions were considered desirable also in TIMES. The non-linear option implemented for modeling CES aggregates is based on the old sketches that were found in the MARKAL GAMS code (but were not active in the code), designed by Dr. Denise Van Regemorter and implemented by Gary Goldstein. Just like under the own-price elasticity option, the calibration of the CES functions is based on the demand projections and the corresponding shadow prices from the solution of a Baseline TIMES run. When defining CES functions, the substitution elasticity between the demands within each CES aggregate is given as an input. The aggregate outputs of the CES functions may then be considered as the final useful demands, with the standard exogenous Baseline projections and own-price elasticities provided for the aggregate demands.

A linearization of the CES demand function formulation has also been implemented, and is available in three different variants. In the lienearized formulation, the CES demand functions can be also subsequently nested further into higher-level CES functions.

All the generalizations presented in this Appendix have been implemented in TIMES v4.1.0. For now, the implementation should be still considered experimental, and therefore any feedback, comments and suggestions from TIMES users concerning the formulation and implementation are welcome.

In this Appendix we provide the input attributes and modeling details associated with the generalized Demand Functions in TIMES. As mentioned above, the implementation of the Demand Functions in TIMES is based on the corresponding formulations originally designed for the MARKAL model generator. The next three sections of the Appendix will address the Sets, Parameters, Variables, and Equations related to the Demand Function options, including the special volume-preserving CES option where the aggregate volume of the components of the combined demand remains equal to the (optionally weighted) sum of the component demands also under any substitution taking place.

2 Mathematical formulation

For the own-price elasticities, we have the following relations (see Part I, Chapter 4), where U_i is the term in the objective function associated with the utility change due to the demand variation of demand *i*:

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

$$\boldsymbol{p_i = p_i^0 \cdot (DM_i / DM_i^{\ 0})^{1/E_i}} \tag{2}$$

$$U_i = \sum_t \left(\frac{p_i^0(t)}{(1+1/E_i)} \cdot \left[DM_i^0(t)\right]^{-1/E_i} \bullet DM_i(t)^{1+1/E_i} \right) \tag{3}$$

Consider then a utility function of the general CES form:

$$U_k = \left(\sum_i \alpha_i^{\frac{1}{\sigma}} x_i^{\frac{\sigma-1}{\sigma}} \right)^{\frac{\sigma}{\sigma-1}} \tag{4}$$

where:

- U_k is the total aggregate utility of demand *k*
- x_i is the demand for commodity *i* (component of the aggregate demand)
- α_i is a share parameter (the sum of which over *i* needs not be equal to 1)
- σ is the elasticity of substitution ($0 < \sigma < \infty$)

The demand functions for x_i can be derived from the utility function in terms of prices, and can be given by the formulas:

$$x_i = \frac{\alpha_i m}{p_i^{\sigma}} \left(\sum_i \alpha_i \ p_i^{1-\sigma} \right)^{-1} = \frac{\alpha_i m}{p_u} \left(\frac{p_u}{p_i} \right)^{\sigma} \tag{5}$$

where *m* is the income level, and p_u is the aggregate price, or unit cost, of the utility, can be given in terms of the individual prices p_i of the demands *i*:

$$p_u = \left(\sum_i \alpha_i \ p_i^{1-\sigma} \right)^{\frac{1}{1-\sigma}} \tag{6}$$

The share parameters α_i can be derived from the expenditure shares, as shown in Eq. (7) below. In the objective function, the utility change can then be calculated by the expression shown in Eq. (9) below.

$$\alpha_i^k = \frac{agg_i^k(t) \cdot DM_i^0(t)}{DM_k^0(t)} \cdot \left(\frac{p_i^0(t)}{agg_i^k(t) \cdot p_{u_k}^0(t)} \right)^{\sigma} \tag{7}$$

$$\beta_k = \left(\frac{p_{u_k}^0(t)}{1 + \frac{1}{E_k}} \right) \cdot \left(DM_k^0(t) \right)^{\frac{-1}{E_k}} \tag{8}$$

$$U_k = \sum_t \left(\beta_k(t) \cdot \left(\left(\left(\sum_i \left(\alpha_i^k(t)\right)^{\frac{1}{\sigma_k}} \cdot \left(agg_i^k(t) \cdot DM_i^k(t)\right)^{\frac{\sigma_k - 1}{\sigma_k}} \right)^{\frac{\sigma_k}{\sigma_k - 1}} \right)^{1+\frac{1}{E_k}} - \left(DM_k^0(t)\right)^{1+\frac{1}{E_k}} \right) \right) \qquad (9)$$

In the above, the coefficients *agg_i* are represent user-defined aggregation coefficients for defining the aggregation from the component demands to the aggregate demands. The constant term corresponding to the Baseline value is subtracted in order to reproduce the value of the Baseline objective function when no variation occurs from the Baseline demands. The non-linear formulation of the elastic demand functions implemented in TIMES follows these expressions.

The corresponding linearized formulations are based on piece-wise linear functions which approximate the integrals over the inverse demand curves, as explained in Part I, Chapter 4. The method described there has been generalized to linearize also the somewhat more complex CES demand functions, allowing also for nested CES functions. Each demand having own-price or substitution elasticities requires the definition of as many variables as there are steps in the discrete representation of the demand curve (both upward and downward), for each period and region. Each such variable has an upper bound, and in the CES formulation they are included in an additional balance equation. However, otherwise the step variables are not involved in other new constraints. Therefore, the linear program is augmented by a number of variables, but does not have any notable number of more constraints than the initial inelastic LP. For partial equilibrium models, volume-preserving demand functions may, however, be preferred over standard CES formulations, and therefore an option for using a simple volume-preserving variant of the CES linearization has been also implemented.

The resulting linearization has been verified to work well over a large range of demand elasticities and price changes, and indeed also with nested CES functions. Cobb-Douglas functions ($\sigma = 1$) are also supported. Using the same linearization approach, even the simple Macro general equilibrium model, which is integrated in TIMES-Macro and includes Cobb-Douglas function nested into a CES production function, might be in principle linearized into an LP problem.

It is also important to note again here that, instead of maximizing the net total surplus, TIMES minimizes its negative (plus a constant). 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.

3 Sets, Switches and Parameters

3.1 Switches

Besides the basic data parameters described in Table D-2 below, the user controls whether the linear or non-linear formulation is activated by means of the switches shown in Table D-1. These switches are provided by the data handling system when the user indicates that the option is to be included in a run.

Table D-1. **Switches**

Switch	Parameter Description
$SET TIMESED NO	Causes the Base Prices to be saved to a GDX file, for subsequent use in a policy analysis run based on any of the elastic demand options.
$SET TIMESED YES	Activates any of LP formulations used for Demand Functions (exact formulations depending on input data)
$SET MICRO YES	Activates any of NLP or mixed LP/NLP formulations used for Demand Functions (exact formulations depending on input data)

3.2 Sets and Parameters

Like all other aspects of TIMES the user describes the demand functions for the energy system model by means of a Set and the Parameters and Switches described in this chapter. Table D-2 below describes the User Input Parameters associated with defining the TIMES demand functions.

Table D-2. **Iput parameters specific to demand functions**

Parameter (Indexes)	Units & defaults	Parameter Description
COM_PROJ (r,y,c)	Commodity unit; [0,∞) default value:none Default i/e: STD	Exogenous reference (Baseline) demand projection of commodity **c** in region **r** and year **y**. In inelastic runs (Baseline runs, and any other model runs with non-elastic demands) the demands are met at the levels of the exogenous projections defined by COM_PROJ, usually exactly, but under certain circumstances some of them may also end up at a higher level than the projection.
COM_AGG (r,y,c,com)	Commodity units [open]; default value:none Default i/e: STD	Defines an aggregation of component demand **c** into an aggregate demand **com** in region **r** and year (period) **y**. Defining COM_AGG between the component demands and the aggregate demand is required for modeling substitution elasticities. If defined zero (e.g. by specifying IE=2), the values will be auto-generated according to the price ratios; defining the COM_AGG values zero is required for using the proper CES functions.
COM_VOC (r,y,c,bd)	Dimension-less; [0,∞); default value:none Default i/e: STD	Defines the maximum demand variation in the lower / upper direction (**bd**=LO/UP) for demand **c** in region **r** and year **y**. The value gives the maximum deviation in proportion to the Baseline demand. Different values may be provided for each direction, thus demand elasticity curves may be asymmetric.

Parameter (Indexes)	Units & defaults	Parameter Description
COM_STEP (r,c,bd)	Integer [1,∞); default value:none	Number of steps to use for the approximation of demand variation in the lower / upper direction (**bd**=LO/UP), and the associated change in producer/consumer surplus, for commodity **c** in region **r**, when using the elastic demand formulations. The shortcut **bd**=FX may be used for defining the same number of steps in both directions.
COM_ELAST (r,y,c,s,bd)	Dimension-less; [open] default value:none Default i/e: STD	Elasticity of demand for commodity ***c***, indicating the following: • For own-price elasticities: how much the demand rises/falls in response to a unit change in the marginal cost of meeting a demand that is elastic. • For substitution elasticities: responsiveness of the ratio in which the component demands are used to the ratio of the prices of those demands Defines elasticities for demand ***c*** in region ***r*** and year ***y***, timeslice ***s*** o ***lim*** = LO/UP : defines the own-price elasticity in the lower / upper direction in the linear formulation o ***lim*** = FX (s=ANNUAL): defines own-price elasticities in the non-linear formulation; can also be used in the linear formulation for defining the own-price elasticities for the aggregate demands, optionally also for defining component-differentiated susbstution elasticities o ***lim*** = N (s=ANNUAL): defines the substitution elasticity for component demands of the demand aggregation represented by commodity ***c***; positive values signify the standard variant, negative values signify the volume-preserving variant formulation

Important remarks:

- *COM_PROJ* should be explicitly defined by the user only for the component demands, and never for the aggregate demands.
- As mentioned in Table D-2, the substitution elasticities can be defined by specifying *COM_ELAST*(r,t,com,ANNUAL,'N') for the aggregate demands. However, 'FX' elasticities for the *component demands* can be optionally specified for defining component-differentiated substitution elasticities. Nonetheless, even when doing so, *COM_ELAST*(r,t,com, ANNUAL,'N') always defines the minimum substitution elasticity among the component demands of ***com***.
- Note that the aggregate demands are always at the ANNUAL level only, and thus only ANNUAL level own-price demand elasticities are supported for the demand aggregates.
- When using the non-linear formulation, demand substitution is supported only at the ANNUAL level for the component demands of the CES aggregates. The demand variations will thus be proportionally the same for all timeslices.
- Multi-level nested CES demand aggregations are also fully supported both in the non-linear and in the linearized case.
- Recursive CES demand aggregations are not supported, neither in the non-linear nor in the linearized case.
- The Cobb-Douglas case (σ_k=1) is also supported, but in the non-linear formulation it is handled by setting σ_k very close to unity.

4 Examples

Assume that we wish to define a non-linear CES demand function for the aggregate demand TLPKM (passenger land travel), having the following component demands:

- TRT – passenger car travel
- TRB – passenger bus travel
- TRW – passenger two-wheeler travel
- TTP – passenger rail travel

The demand function can be set up with the following input parameters (where **r** stands for regions, **t** for milestone years, and '**0**' for interpolation option placeholder):

Table D-3. **Non-linear CES demand function example**

Parameters	Description
`COM_AGG(r,'0','TRT','TLPKM') = 2;`	Aggregation of TRT into TPASS with price ratios
`COM_AGG(r,'0','TRB','TLPKM') = 2;`	Aggregation of TRB into TPASS with price ratios
`COM_AGG(r,'0','TRW','TLPKM') = 2;`	Aggregation of TRW into TPASS with price ratios
`COM_AGG(r,'0','TTP','TLPKM') = 2;`	Aggregation of TTP into TPASS with price ratios
`COM_ELAST(r,t,'TLPKM','ANNUAL','FX')=0.35;`	Own-price elasticity of aggregate demand
`COM_ELAST(r,t,'TLPKM','ANNUAL','N')=1.2;`	Elasticity of substitution between components
`COM_VOC(r,t,'TLPKM','UP')=1;`	Max. upper variance of aggregate demand

Assume now that we wish to define the same demand function but with the linear formulation for the CES function. The demand function can be set up with the following input parameters (where **r** stands for regions, **t** for milestone years, and **bd** for the inequality bound types ('LO', 'UP')):

Table D-4. **Linear CES demand function example**

Parameters	Description
`COM_AGG(r,'0','TRT','TLPKM') = 2;`	Aggregation of TRT into TPASS with price ratios
`COM_AGG(r,'0','TRB','TLPKM') = 2;`	Aggregation of TRB into TPASS with price ratios
`COM_AGG(r,'0','TRW','TLPKM') = 2;`	Aggregation of TRW into TPASS with price ratios
`COM_AGG(r,'0','TTP','TLPKM') = 2;`	Aggregation of TTP into TPASS with price ratios
`COM_ELAST(r,t,'TLPKM','ANNUAL','FX')=0.35;`	Own-price elasticity of aggregate demand
`COM_ELAST(r,t,'TLPKM','ANNUAL','N')=1.2;`	Elasticity of substitution between components
`COM_STEP(r,'TRT','FX')=100;`	Number of steps for TRT in both directions
`COM_STEP(r,'TRB','FX')=100;`	Number of steps for TRB in both directions
`COM_STEP(r,'TRW','FX')=100;`	Number of steps for TRW in both directions
`COM_STEP(r,'TTP','FX')=100;`	Number of steps for TTP in both directions
`COM_STEP(r,'TLPKM','LO')=120;`	Number of steps for TLPKM in lower direction
`COM_STEP(r,'TLPKM','UP')=80;`	Number of steps for TLPKM in upper direction
`COM_VOC(r,t,'TRT',bd)=0.8;`	Max. variance of TRT, given in both directions
`COM_VOC(r,t,'TRB',bd)=0.8;`	Max. variance of TRB, given in both directions
`COM_VOC(r,t,'TRW',bd)=0.8;`	Max. variance of TRW, given in both directions
`COM_VOC(r,t,'TTP',bd)=0.8;`	Max. variance of TTP, given in both directions
`COM_VOC(r,t,'TLPKM','LO')=0.5;`	Max. lower variance of aggregate demand
`COM_VOC(r,t,'TLPKM','UP')=0.3;`	Max. upper variance of aggregate demand

Note that using 'FX' as a shortcut for bd={'LO','UP'} in *COM_STEP* is only supported in TIMES v4.4.0 and above, and that *COM_VOC* does not have any such shortcut.

5 Variables

The variables that are used to model the Demand Functions in TIMES are presented in Table D-5 below. The primary role of the variables and equations used to model the functions is to control the standard TIMES variable and the associated dynamic cost of these.

Table D-5. Model variables employed in demand functions

Variable (Indexes)	Variable Description
VAR_COMPRD (r,t,c,s)	Variable used for tracking the Gross production of a commodity **c** in region **r**, period **t**, and timeslice **s**.
VAR_DEM (r,t,c)	Variable used for the enodenous (elastic) demand for commodity **c** in region **r**, and period **t**, when the demand function is non-linear.
VAR_ELAST (r,t,c,s,bd)	Step variables used to linearize elastic demand curves for demand **c** in region **r**, period **t**, and timeslice **s.** The index **bd**=LO corresponds to the direction of decreasing the demand, while **bd**=UP denotes the direction for demand increase.
VAR_OBJELS (r,bd,cur)	Variable used for accounting the total discounted endogenous losses (**bd**=LO) or gains (**bd**=UP) in the utility of region **r** in currency **cur** through the demand variations of all elastic demands.

5.1 VAR_COMPRD(r,t,c,s)

Description: The amount of demand commodity **c** procured at time period **t**, timeslice **s**.

Purpose and Occurrence: This variable tracks the total amount of demand commodity produced. This variable is normally only created if a bound is imposed on total production of the demand commodity, or a cost is explicitly associated with the production level of the demand. However, when defining CES demand functions, the variable is always created both for the component demand and for the aggregate demand. The variable is defined through the equations EQE_COMPRD and/or EQ(l)_COMBAL.

Units: PJ, Bvkm, or any other unit defined by the analyst to represent the quantity of the demand.

Bounds: This variable is non-negative. It is by default not otherwise directly bounded. It can be directly bounded by the COM_BNDPRD attribute. It may be indirectly bounded by specifying a user constraint referring to it by UC_COMPRD.

5.2 VAR_DEM

Description: The total amount of demand for commodity **c** in time period **t**.

Purpose: This variable is used for tracking the endogenous amount of demand for commodity **c** in the non-linear formulation of elastic demands.

Occurrence: This variable is not created in the LP formulation. It is only created in the non-linear formulation of demand functions based on own-price and/or substitution elasticities (as well as in the Macro formulation). It is generated for each demand with a non-linear own-price elasticity function, and for all demands associated with a non-linear CES demand function.

Units: PJ, Bvkm, or any other unit in which demands are tracked.

Bounds: This variable is non-negative and is not bounded upwards.

5.3 VAR_ELAST (r,t,c,s,j,bd)

Description: Variables used to linearize elastic demand curves by step-wise variations.

Purpose: To indicate how far the demand variation extends on the elasticity curve, by step.

Occurrence: 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 and revenues of demand gains as explained in Part I, Chapter 4.

These variables are defined whenever a demand is declared to be price elastic, either to its own price or through cross-elastic substitution. These variables are indexed by j, where j runs over the number of steps used for discretizing the demand curve of demand commodity c. The jth variable stands for the portion of the demand that lies within discretization interval **j**, on side **bd** (**bd** indicates either increase or decrease of demand w.r.t. the reference case demand). In the objective function, these variables are used to represent the utility change caused by demand losses or gains, as explained in Part I, Chapter 4.

Units: Demand units: PJ, Bvkm, or any other unit in which the demand is tracked.

Bounds: This variable is non-negative. Each ELAST variable is bounded upward via virtual equation EQ_BNDELAS, of in the case of a CES function, via the equation EQL_COMCES.

5.4 VAR_OBJELS (r,bd,cur)

Description: Variables used to linearize elastic demand curves by step-wise variations.

Purpose: To indicate how far the demand variation extends on the elasticity curve.
Occurrence: The utility change caused by all demand losses and gains as explained in Part I, Chapter 4.

These variables are defined whenever any of the elastic demand formulations is used. These variables are indexed by **bd** (**bd** indicates either increase or decrease of demand w.r.t. the reference case demand). These variables are included in the objective function, to represent the total utility changes caused by demand losses or gains, as explained in Part I, Chapter 4.

Units: Demand units: PJ, Bvkm, or any other unit in which the demand is tracked.

Bounds: This variable is non-negative.

6 Equations

The equations that are used to model the demand functions in TIMES are presented in Table D-6 below. The primary role of the variables and equations used to model Demand Functions is to control the standard TIMES demand variables VAR_DEM and the associated losses or gains in consumer's utility in the regional demand utility part of the objective function (EQ_OBJELS).

Reminder: The elastic demand function formulations are activated at run time from the data handling system. The linear formulation is activated by the switch $SET TIMESED YES and the non-linear formulation by the switch $SET MICRO YES.

Table D-6. Model constraints specific to demand functions

Constraints (Indexes)	Constraint Description	GAMS Ref
EQG_COMBAL (r,t,c,s)	The commodity balance constraint associated with the demand function of commodity c, as an inequality. The constraint requires that the total production of the demand commodity is greater than or equal to the endogenous elastic demand. This constraint is normally generated for all demands modeled with own-price elasticities.	EQCOMBAL.mod
EQE_COMBAL (r,t,c,s)	The commodity balance constraint associated with the demand function of commodity c, as a strict equality. This constraint is automatically generated for the component demands of all CES demand functions, i.e. demands modeled with substitution elasticities.	EQCOMBAL.mod
EQE_COMPRD (r,t,c,s)	This equation is a strict equality and is generated in two forms for the demands included in demand functions: 1. Defining equation for the commodity production of commodity c. This constraint is automatically generated for all the component demands of CES demand functions. 2. Balance equation for the total variation of the component demands of the aggregate demands of CES demand functions. This constraint is automatically generated for all the aggregate demands of CES demand functions.	EQCOMBAL.mod
EQL_COMCES (r,t,com,c,s)	The constraint bounding the step variables of a demand commodity **c** included as a component in the CES function of demand aggregate **com**. The constraint is generated for each of the component demands whenever the aggregate demand of a CES function has been modeled with an own-price elasticity (otherwise variable bounds are sufficient).	EQOBJELS.mod
EQ_OBJELS (r,bd,cur)	The calculation of the endogenous losses or gains in utility through the demand variations of all elastic demands are discounted and summed together into the VAR_OBJELS variable representing the regional elastic demand cost part of the objective function, which is subsequently included in the total objective function (EQ_OBJ).	EQOBJELS.mod