Flexible Energy Scheduling Tool for Integrating Variable generation (FESTIV)

USER'S GUIDE

Version 1.0

February 2019

```
**********************************************************************
* Flexible Energy Scheduling Tool for Integrating Variable generation *
**********************************************************************
**    FFFFFF      EEEEEE     SSSSSS     TTTTTTTT    IIIIII     VV     VV    **
**    FF          EE         SS           TT          II       VV   VV     **
**    FFFFFF      EEEEEE     SSSSSS       TT          II        VV VV      **
**    FF          EE             SS       TT          II         VVV       **
**    FF          EEEEEE     SSSSSS       TT        IIIIII        V        **
**********************************************************************
*************** National Renewable  Energy Laboratory ****************
**********************************************************************
```


National Renewable Energy Laboratory

15013 Denver West Parkway

Golden, Colorado - 80401

Electric Power Research Institute

3420 Hillview Ave

Palo Alto, California 94304

Website : http://www.nrel.gov/electricity/transmission/festiv.html

This page is intentionally left blank.

Version	Date	Updates
0.1	June 2014	First Version
0.2	September 2014	Modified prior to external distribution
0.3	December 2015	Updated with new features and GAMS framework
1.0	February 2019	Prepared for open release

1. Table of Contents

List of Figures

List of Tables

List of Acronyms

AACEE – Absolute ACE in Energy

ACE – Area Control Error

AGC – Automatic Generation Control

CPS2 – Control Performance Standard 2

DAM – Day-Ahead Market

DASCUC – Day-Ahead Security-Constrained Unit Commitment

FESTIV – Flexible Energy Scheduling Tool for Integrating Variable generation

GUI – Graphical User Interface

ISO – Independent System Operator

LMP – Locational Marginal Price

NERC – North American Electric Reliability Corporation

PCM – Production Cost Model

RTM – Real-Time Market

RTO – Regional Transmission Organization

RTSCUC – Real-Time Security-Constrained Unit Commitment

RTSCED – Real-Time Security-Constrained Economic Dispatch

SCRPU – Security-Constrained Reserve Pick-Up

TSO – Transmission System Operator

VCR – Variable Capacity Resource

VG – Variable Generation

VOLL – Value Of Lost Load

Introduction

The Flexible Energy Scheduling Tool for Integrating Variable generation (FESTIV) is a multi-timescale steady-state power system operations simulation tool that aims at replicating the full time spectrum of scheduling resources to meet energy and reliability needs of the bulk power system. It uses a suite of five scheduling sub-models by default: the day-ahead security-constrained unit commitment (DASCUC), the real-time security-constrained unit commitment (RTSCUC), the real-time security-constrained economic dispatch (RTSCED), the automatic generation control (AGC), and the security-constrained reserve pick-up (SCRPU). Each of these sub-models is integrated within FESTIV at various timescales and trigger points that are configurable by the user. The goal of FESTIV is to test changing of inputs and changing of operational structures to understand the impacts these changes have on reliability, efficiency, and incentive structures. One of the main motivations of the FESTIV tool was to better understand the variability and uncertainty impacts of variable generation (VG), like wind and photovoltaic solar power, and the effectiveness of strategies to mitigate those impacts. The tool has since evolved to allow for testing the impacts of multiple emerging technologies, new operating strategies and scheduling applications, and changes to the electricity market design. The key feature of FESTIV is its full set of scheduling applications and its ability for users to make significant modifications while still including the full temporal spectrum or operational scheduling.

Production cost models (PCM) are powerful tools that also simulate the steady-state operations of the bulk power system. Although in recent years, these commercially available production cost models have made significant enhancements, a few limitations of these models are the motivation for the initial development of FESTIV back in 2010. These are listed below[*]:

- PCM had typically simulated operations at hourly time resolution, which lost information on the variability and uncertainty impacts that occurred within the hour.
- PCM typically run either a single- or two-stage optimization. In a single-stage optimization, there is no possible impact from making decisions that are incorrect due to imperfect knowledge. In a two-stage simulation, the tools typically modeled the commitment of resources to meet day-ahead load and variable generation forecasts, with the dispatch decision made to meet the actual load and variable generation output. In reality, forecast errors are occurring at multiple time horizons, with multiple chances of correction as operations approaches real-time.

[*] During the development stages of FESTIV, there has been substantial progress in commercial PCM as well, such that some of the listed limitations have been partially or fully eliminated.

- PCM typically focus their evaluation on total production costs and have little to no information on reliability metrics. While, load shedding instances could be an output of these models; in hourly, one- or two-stage deterministic simulations, any load-shedding would be extremely rare within the PCM simulation and therefore this metric may provide little value used in this manner. Understanding reliability impacts and the tradeoff of reliability with economic efficiency is becoming a more important study need.
- PCM typically do not simulate the automatic generation control or reserve pick-up sub-models, which are extremely important for understanding the detailed costs of controlling the system to reduce imbalance, and understanding what the true reliability impacts are.
- PCM typically do not simulate the deployment of any operating reserve. Although the tools do usually simulate the holding of these reserve, in understanding the impacts of emerging technologies and changing operating strategies and market designs, it is important to also know how these operating reserve are being deployed. Without this knowledge, any new reserve requirement methodologies are not being validated or studied fully.
- PCM typically cannot show the detailed benefits of fast responding resources, like demand response or energy storage resources.
- PCM typically have limited flexibility to model changing operational strategies or market designs. They typically model a default "common denominator" strategy, which although can give good information on high-level production costs, does not allow for an understanding of how changing operating strategies or market designs can impact results.
- PCM typically do not model operator action or response to challenging operations which loses information on realistic reliability and costs results.
- The multi-stage PCM typically are not interlinked at timescales realistic to actual scheduling operations. This also loses information on reliability, production costs, and pricing information.
- Finally, PCM are proprietary models and may require significant effort from commercial entities to make changes to model new operational strategies, market designs, or technologies. Thus being difficult to perform advanced research on new topics.

Load flow and dynamic modeling (LFDM) tools are also commonly used by the industry. These tools have tremendous capability to represent the flow of real and reactive power on the bulk power system for both steady-state snap shots in time, or dynamic events. FESTIV, in its default modes, does not have the capabilities of these tools to represent dynamics or even AC power flow. However, there are some notable limitations in these tools that motivate some of the capabilities that are present in FESTIV.

- LFDM typically do not model corrective action which can hide the performance of the system to particular events.

- LFDM typically do not include time-varying conditions. So load levels and VER production are typically constant for the length of a single simulation.
- When looking at active power balancing issues, LFDM typically look at frequency during the first ~30 seconds. They typically do not evaluate what happens during the full recovery periods following events and do not evaluate secondary frequency control and area control error.
- There is typically no possibility to make modifications to steady-state economic decisions within LFDM tools.

FESTIV is an ongoing research tool that attempts to address these limitations. A list of the primary unique features to address the above limitations is described below:

- FESTIV models the power system at multiple operational timescales. Each sub-model typically is evaluating the power system at a different time resolution and time horizon and the tool allows the user to create the appropriate timing configurations for the current study. For each sub-model, the time resolution, update frequency, time horizon, and expected model solution time are configurable by the user. The AGC and ACTUALS sub-models are at the highest resolution, typically 2-6 seconds. Any faster than this resolution would require the modeling of power system dynamics and thus it is advised not to model at resolutions higher than this.
- Depending on the time horizon and time update frequency, FESTIV simulates the notion that system conditions are being updated continuously such that decisions can continuously be made throughout a study period to correct those that were made before the accurate information was known. Forecasts of load and VG are in dimensions of sub-model, forecast time, and forecast horizon. Therefore, the forecast of VG for RTSCUC run at 10:00 for time interval 11:15, is different than the forecast of VG for RTSCED run at 10:55 for 11:15 and so on.
- The primary metrics of FESTIV are production costs that are evaluated at the finest time resolution, and reliability, in terms of imbalance, herein termed area control error (ACE), which is also evaluated at the finest time resolution for the entire study period. Line flow exceedance and control performance standards are also a part of the reliability evaluation. In addition, incentive structures in terms of existing or new pricing algorithms are also evaluated as are revenues and profits.
- FESTIV uses the AGC sub-model which directs the units that have AGC capability to be used to control the ACE to minimize the imbalance error. Numerous AGC strategies can be used as well as new ones that the user can define. Without the AGC sub-model it is impossible to simulate true steady-state reliability impacts.
- FESTIV simulates both the holding and deployment of any category of operating reserve which the user provides. In the definition given in FESTIV, operating reserve is capacity above or below the energy schedule in one sub-model, which are then deployed in a later sub-model. For example,

regulating reserve is reserve held in RTSCED and deployed in AGC. Contingency reserve is reserve held in RTSCED and deployed in the SCRPU.

- Due to the multi-timescale approach, new technologies, like fast energy storage or demand response can be simulated in FESTIV with the appropriate data and parameters and evaluated to see how these technologies can change the reliability or efficiency of a system.
- FESTIV has the flexibility to adjust the operating structures or market designs. As discussed the timing parameters are adjustable by the user. Sub-model pre and post rule modules allow for users to add unique rules that can adjust FESTIV to reflect any specific features of a particular system or study. Users may also modify or make new formulations in any of the optimization models to add unique features that are being explored in market clearing software.
- FESTIV models operator action in terms of the SCRPU. The SCRPU simulates the operator calling for reserve or changing the commitment or dispatch based on current conditions or events. While simulation of human behavior is not possible, the SCRPU allows for simulating an operator making certain decisions based on criteria that may be predefined..
- FESTIV sub-models are all interlinked such that the output of one sub-model is utilized as an input to a later sub-model. The sub-models are solved in a sequence identical to how they are solved in a typical system operation. This allows for more realistic results when one sub-model will change the system conditions such that it would have an impact on a separate sub-model.
- FESTIV is open-source and allows for further flexibility from the FESTIV User Group to either make changes themselves (by use of functional mods or formulation mods).

More information on FESTIV can also be found in Ela 2011[†], and the studying of variability and uncertainty of variable generation at multiple timescales using FESTIV can be found in Ela 2012[‡]. This User Guide focuses on how users would run typical studies using FESTIV, and does not necessarily focus on FESTIV from a developer side. This User Guide is set up as follows. First, directions on getting FESTIV set up is described. This involves set up of your *FESTIV Directory*, the needed updates to communicate between the two platforms of FESTIV, and the required files for running FESTIV successfully. Section 0 then provides a brief description of each sub-model and its functionality to give the user an idea of what is occurring during the model run. Section 4 describes the building of the system and all of its input needs. The FESTIV graphical user interface (GUI) is also described including all of the configurable parameters. A Section 5 illustrates the various output views including metrics, figures, the *Main Output File*, and the *FESTIV*

[†] E. Ela, M. Milligan, and M. O'Malley, ""A flexible power system operations simulation model for assessing wind integration," Proceedings of IEEE PES General Meeting 2011, Detroit, MI.
[‡] E. Ela, M. O'Malley, "Studying the Variability and Uncertainty of Variable Generation at Multiple Timescales," IEEE Transactions on Power Systems, vol. 27, no. 3, pp. 1324-1333, Aug. 2012.

Helper. Finally, some advanced features are briefly described in Section 6 and 7 on user-defined modifications (termed Functional or Formulation Mods) and debugging, respectively.

FESTIV Platform and Setup

FESTIV utilizes two main platforms; General Algebraic Modeling System (GAMS) and Matrix Laboratory (MATLAB). Matlab is the main home for FESTIV, whereas GAMS hosts the code for the optimization sub-models. FESTIV also uses .csv, .xls, and .txt files delivering inputs and saving outputs.

2.1 Setting up MATLAB

Once MATLAB has been installed, it is important to start it in the FESTIV directory. One way to do this is detailed below.

1) Once the MATLAB setup is complete, a shortcut is placed on the desktop.
2) Right click the MATLAB shortcut icon and select *Properties*.
3) Select the *Shortcut* tab on the top.
4) Modify the path in the *Start In* section to match the location of the documents used for FESTIV modelling. Hereon, this location will be referred to as the *FESTIV Directory*. By doing so, whenever MATLAB is started using the shortcut icon the default start location will be pointed to the FESTIV Directory.
5) The path shown below is an example. Users can define their own path depending on the working directory location.
6) Click Apply and the click ok to confirm.

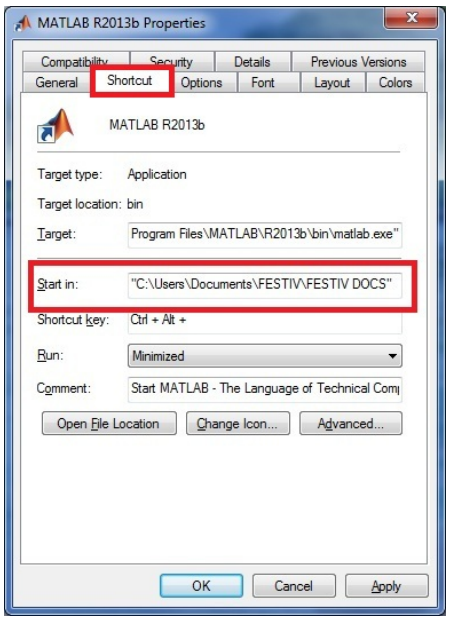


Figure 1: Configuration of the MATLAB shortcut after installation

2.2 The FESTIV Directory

The *FESTIV Directory* is the location where the FESTIV program files are located. The FESTIV Directory includes additional folders for Inputs, Outputs, Model rules, and temporary files. In addition, all the main scripts and functions that are used within FESTIV are also included in the FESTIV Directory.

The *Input Directory* folder is the location of all input case files. The *OUTPUT Directory* folder is the location where FESTIV will save all of the simulation outputs and results when requested by the user. The *TEMP Directory* folder is a location where temporary files are written during program execution. The *MODEL_RULES Directory* is a location to add any new features for specific unique rules to use within FESTIV. The remaining files in the *FESTIV Directory* are the required files to run FESTIV.

Executing the FESTIV.m script will begin the FESTIV simulation by launching the FESTIV Graphical User Interface. This can also be achieved by typing FESTIV on the Matlab command window. In addition, a FESTIV run that has been started previously, can continue by running *Start_FESTIV_From_Previous_Execution,* or typing the same on the Matlab command window. This requires the previously started simulation to already be loaded within Matlab.

The FESTIV *Input Directory* should contain folders containing the separate simulation cases to be executed. For example, the 5 bus PJM test system input directory would be FESTIV\Input\PJM_5_BUS_SYSTEM. A common directory structure that the FESTIV Development Team uses is Input\"System"\"Study Name"\"Case"\"Dates or Scenarios", which allows for different cases for the same system and different dates to be located in an organized manner. The user can configure the *Input Directory* in a manner that

suits his/her needs. As discussed in Section 4: FESTIV Inputs, the main input file can either be .xls/.xlsx or .ht format and is included somewhere on the last folder for where inputs are located. If using .xls/.xlsx, the final folder directory MUST contain a folder labeled 'TIMESERIES' which must include all of the time series data required to simulate the system along with the *Main Input File* for the system which must be located in the input file. When using .h5, the timeseries data must be included on the same .h5 file.

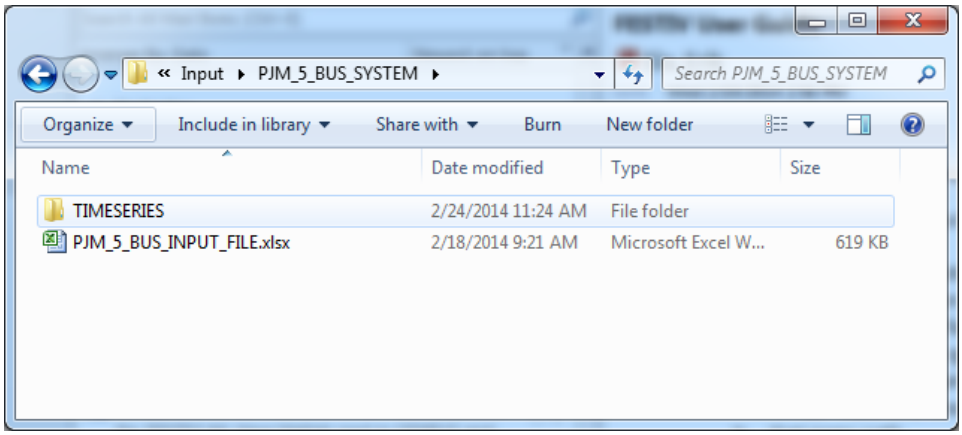


Figure 2: FESTIV\Input directory example

The details of the input files required in the *Input Directory* are discussed in Section 0.

The FESTIV *OUTPUT Directory* is the location where FESTIV will save all of the associated output data from a previously completed simulation. FESTIV will create a folder in this directory corresponding to the output name provided at the end of the simulation and all of the associated data files will be contained in this folder. For example, if the PJM 5 bus system is simulated and saved with the name 'PJM 5 Bus Example,' the following folder structure is created.

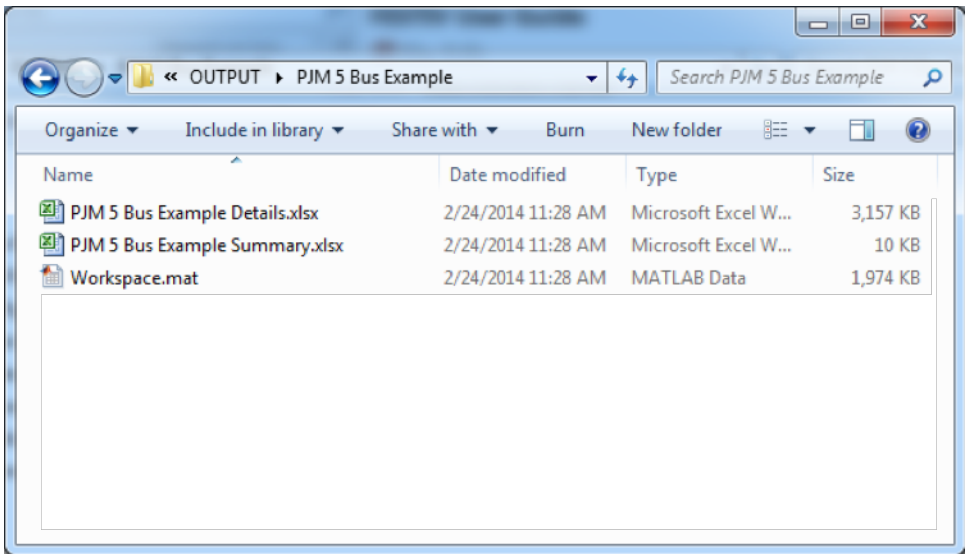


Figure 3: FESTIV\OUTPUT directory example

The details of the different information saved in the *OUTPUT Directory* will be discussed in Section 0.

The FESTIV *TEMP Directory* is used by FESTIV to write temporary files required to complete the simulation. For the most part, the *TEMP Directory* is temporary files that GAMS or MATLAB produces during the simulations. Additional user defined files that GAMS reads that are created will also be located here as well. Generally, users should not modify files within the TEMP folder, with the exception of .opt files which may contain solver specific features.

Finally, the FESTIV *MODEL_RULES Directory* will contain any additional codes that are used as unique rules within FESTIV. The details of the model rules will be discussed in Section 6.

2.3 Configuring the Interface Between GAMS and MATLAB[§]

GAMS and MATLAB need to be configured in order for FESTIV to be able to communicate between the programs.

Two files, *gams.dll* and *festiv.inp,* are provided with the FESTIV package. Copy the *gams.dll* file and paste it in the MATLAB Directory. Copy the *festiv.inp* and paste it in the *FESTIV Input* directory. The GAMS version must not be older than version 23 in order for the communication to work. Finally, both GAMS and MATLAB should both be set up on either 32-bit or 64-bit in order for communication to work.

The next step is to include GAMS in the system environment path so that GAMS can be called from within MATLAB. This can be accomplished via the following steps. Note that these steps are written with respect to Windows 10. Also note that the GAMS version may not necessarily be 24.0. The user should make sure to include the proper version number when appending the path name in step 5.

1. Right click 'My Computer' and click 'Properties'
2. Click 'Advanced system settings'
3. Click 'Environment Variables…'
4. Highlight 'Path' in the 'System variables' scrollable list and click "Edit…'
5. In the 'Variable value' edit box, append the GAMS path to the list of variables (e.g. …;C:\GAMS\win64\24.0)
6. Click 'Ok' in all of the open dialog boxes to save the changes.

The file *getgamspath.m* is required for MATLAB to call GAMS. This module should automatically locate GAMS.exe on your hard drive.

MATLAB communicates with GAMS via '.gdx' files. In order to accomplish this, two functions have been included in MATLAB, 'rgdx' and 'wgdx.' The 'wgdx' function is used to write the '.gdx' and 'rgdx' is used to read the '.gdx' file. The data sent via these files must be cell data structures from MATLAB and read from GAMS as cell data structures.

---

§ More information regarding interfacing GAMS and MATLAB can be found in this paper : GDXMRW: Interfacing GAMS and MATLAB, Michael C. Ferris, Rishabh Jainy, and Steven Dirksez; http://www.gams.com/dd/docs/tools/gdxmrw.pdf

The cell structure of a variable should include the following fields:

1. name: Name of the GAMS parameter, set, or variable.
2. form: Form of the GAMS parameter, set, or variable. This can be sparse or full. In FESTIV, all communication is performed using "full" form.
3. type: Whether the value is a parameter, set, or variable.
4. uels: The set list for which the values are for.
5. val: The values of the set (0 or 1), or the value of the parameter.

Details on these associated fields can be found in Ferris et al[3].

For example, in order to send load data to GAMS from MATLAB, the following variable should be declared in MATLAB and then sent to GAMS via the 'wgdx' command:

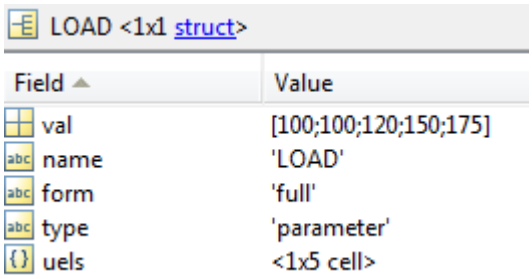


Figure 4: Structure of a variable sent to GAMS from MATLAB

The 'val' field includes the load values for all of the associated intervals being used in the next GAMS run. The 'name' field is the name that the data point uses within the GAMS model. The 'form' attribute is used to distinguish the type of data being passed by MATLAB (always full). The 'type' is used by GAMS to classify the data being passed (set, variable, or parameter). The 'uels' field is set list in which the data operates over, in this case, it would be the Interval set (e.g., {'Interval1', 'Interval2', 'Interval3', 'Interval4', 'Interval5'}).

FESTIV Functionality

3.1 FESTIV Flow Diagrams

The FESTIV model is a multi-timescale steady-state power system operations and wholesale market operations simulation tool that aims at replicating the full time spectrum of scheduling resources to meet energy and reliability needs of the bulk power system. The high-level diagram of FESTIV processes is shown in Figure 5. A new run is typically started by running FESTIV.m or typing FESTIV on the Matlab command window. Next, the simulation will detect hardware to determine whether the GUI should be launched. Additional functionality is evaluated specifically to NREL's high performance computer. If the GUI is enabled, it is opened for the user to fill out needed information for the simulation. Once complete, the user presses GO and the FESTIV simulation process begins. If the GUI is not enabled, the needed information that is typically provided by the GUI must be pre-loaded, and the FESTIV simulation process begins immediately. Users can also run a previous incomplete FESTIV run from mid-execution by running Start_FESTIV_From_Previous_Execution or typing the same on the command window. Once the FESTIV simulation process completes, it will check if multiple simulations were set to run and complete once all simulations are done. The FESTIVHelper is a tool separate from the main FESTIV process. It allows a user to load previously completed simulations and run a number of different analyses and charts on the completed simulations.

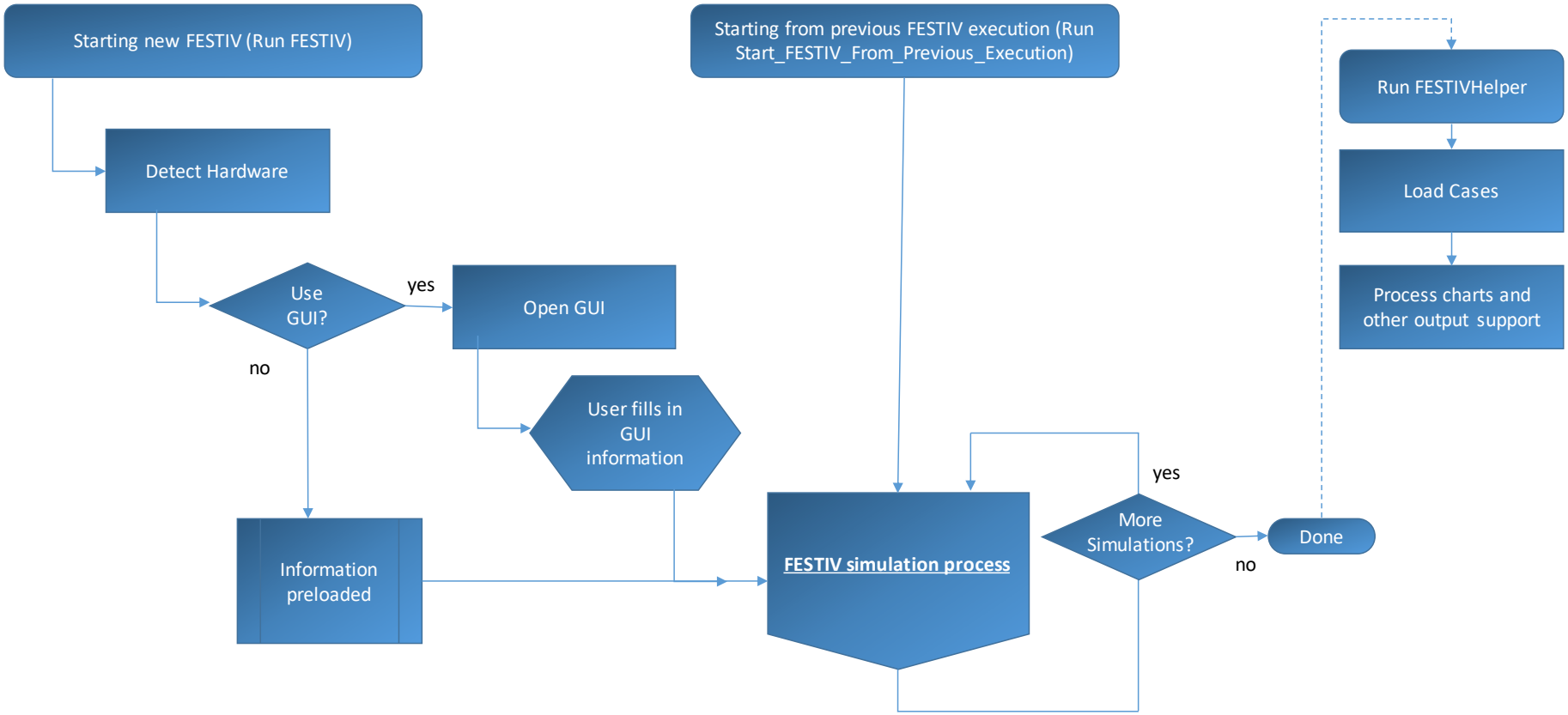


Figure 5: High-level flow diagram for FESTIV simulation processes.

The FESTIV Simulation Process is then shown in Figure 6. Once a new simulation process begins, data is loaded from the GUI, Main Input File, and the time series input files. Indices based on the input data are also declared and there is a simple data validation that occurs. Next, network calculations are completed. Then the initial scheduling models are run to set up initial conditions. Any forced outages that are enabled by the user are then set and data adjusted for the FESTIV Simulation Time Loop. Then the FESTIV Simulation Time Loop begins. Once the study period is over, the results are calculated, data saved if requested, and plots are displayed if requested. If multiple runs are requested by the user, this entire process will continue again until all simulations are complete.

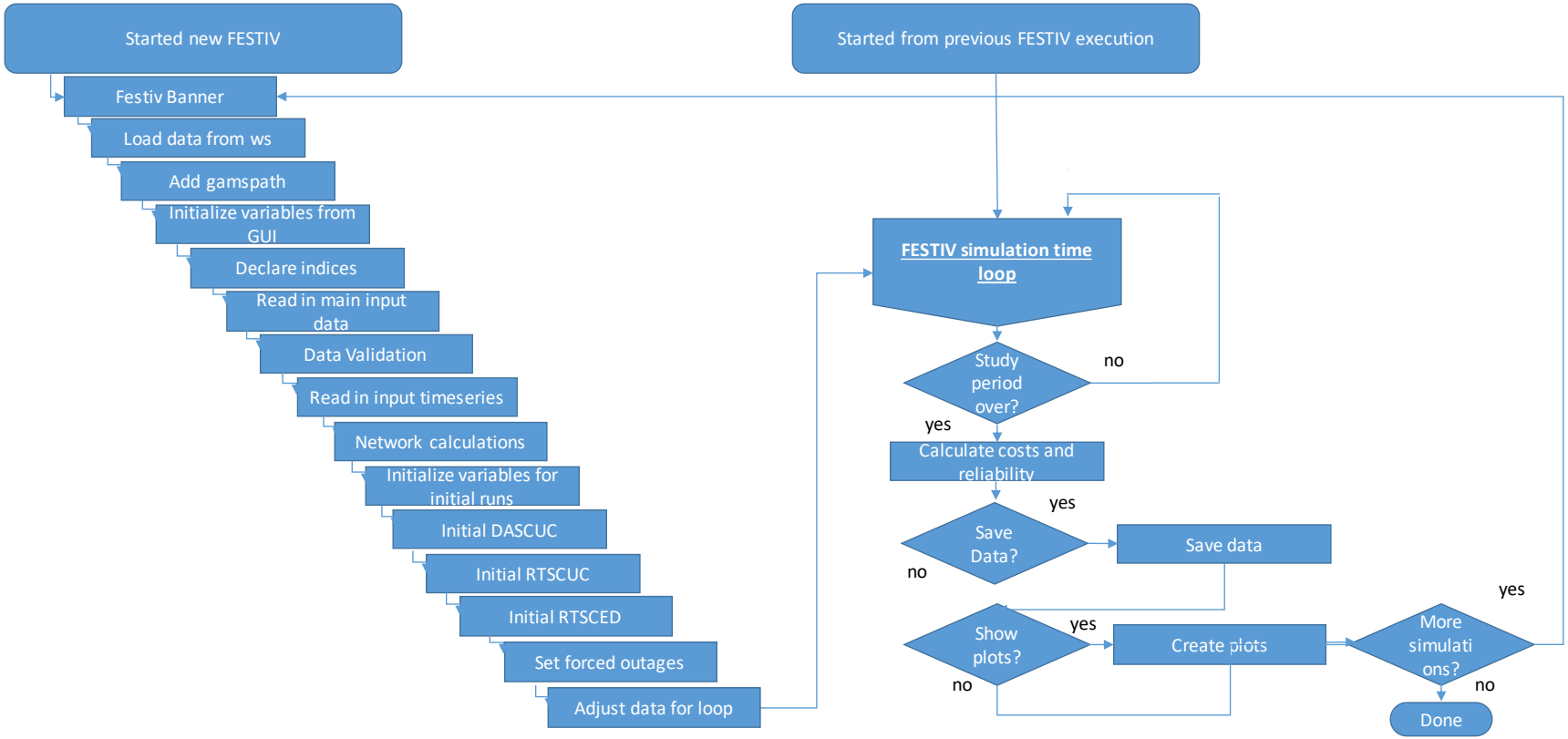


Figure 6: High-level flow diagram for FESTIV simulation processes.

FESTIV uses a suite of five default scheduling sub-models: the DASCUC, the RTSCUC, the RTSCED, the AGC, and the SCRPU. Each of these sub-models is integrated within FESTIV at various points within the FESTIV Simulation Time Loop that are configurable by the user. Figure 7 shows the high-level flow diagram of the FESTIV Simulation Time Loop. Full lines represent process flow and dashed lines represent data flow. The model attempts to replicate actual system operations at a high time resolution and allows for flexibility to accommodate the many different market and operational structures that are in existence throughout the world. Each of the sub-models is run at specific intervals as defined by the user, with each able to consider varying levels of detail of variability and uncertainty and how to accommodate it, with different binding decisions that are used as inputs in later sub-models. This is the only way to measure detailed reliability and efficiency impacts from variability and uncertainty. As will be discussed in Section 6, the decisions for each process can be modified by the user with addition of Model Rules.

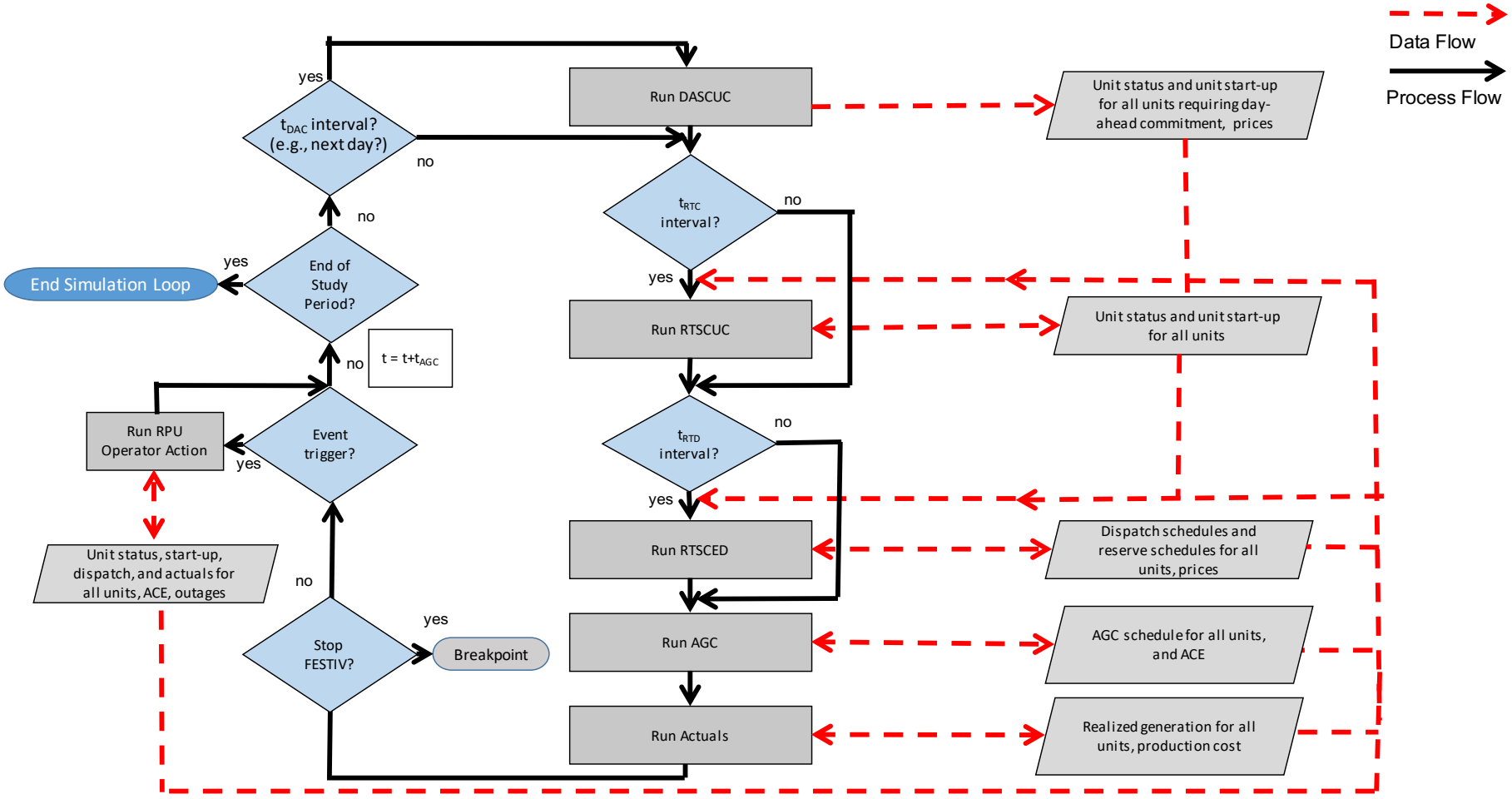


Figure 7: High-level flow diagram for daily simulation of FESTIV. Solid lines represent process flow and dashed lines represent data flow.

3.2 FESTIV Sub-Model Descriptions

In this section, we describe the various sub-models that are the primary processes within FESTIV. This includes DASCUC, RTSCUC, RTSCED, AGC, Actuals, and SCRPU. When discussing these within the simulation process we refer to them as sub-models and when referring to them outside the simulation process and refer to their application we refer to them as scheduling processes.

Two SCUC models are included within FESTIV, the DASCUC and the RTSCUC. The key difference between them being what generating units the program can turn on or off. The SCUC models are generally based on the model shown in Carrion and Arroyo (2006). The DASCUC is typically run for a longer time horizon and gives the initial commitment status and start-up decisions for all units. Only the statuses of units with a start-up time greater than $t_{RTCSTART}$ are made final decisions (see Figure 7). With these statuses as input, the RTSCUC is repeated throughout the day to continuously update unit commitment of all resources, including those with start-up times less than $t_{RTCSTART}$, based on new forecasts and system conditions. The variable $t_{RTCSTART}$ which will determine the difference in which decision variables must be made in each sub-model, is configurable by the user. In addition, the user may make additional rules that sets what resources can have their commitment set in RTSCUC by modifying the RTSCUCSTART script. To match many current market designs, the DASCUC also determines the day-ahead market energy and AS prices, while the RTSCUC prices are mostly advisory. If any part of RTSCUC pricing is needed for settlements, this can be modified using Model Rules. RTSCUC is run every t_{RTC} minutes throughout the day and DASCUC is repeated every t_{DAC} hours. DASCUC evaluates schedules and commitments at interval lengths of I_{DAC} hours over a horizon of H_{DAC} intervals. RTSCUC evaluates schedules and commitments at interval lengths of I_{RTC} minutes over a horizon of H_{RTC} intervals.

The DASCUC and RTSCUC formulations are very similar. The objective is to minimize costs including incremental energy, no-load, start-up, and ancillary service costs and include load shedding and insufficient AS penalty costs. They ensure typical generator constraints including minimum and maximum capacity, ramp rate, minimum on and off time, maximum starts per day, etc. The default DASCUC also allows for variable startup costs that depend on the offline time of the unit starting up. Due to the multiple timescale approach of FESTIV, stepwise start-up decisions where a unit goes from zero output to minimum generation are not plausible. Start-up trajectories are modeled in both programs similarly to Arroyo and Conejo (2004). Both sub-models incorporate the transmission network including contingencies via a dc load flow using power transfer distribution factors (PTDF) and line outage distribution factors (LODF) matrices. The algorithms also allow for HVDC lines and phase shifting transformers to be part of the network. Both DASCUC and RTSCUC incorporate various ancillary service constraints as directed by the user.

RTSCED is very similar to RTSCUC except that its default mode cannot change the commitment status that was provided by RTSCUC. RTSCED minimizes incremental energy costs, ancillary service costs and load shedding and insufficient AS penalty costs. It also contains all of the previously mentioned generator constraints except for commitment-based constraints, all network constraints including contingencies using PTDF and LODF, and various user-defined ancillary service constraints. RTSCED determines the energy dispatch schedules, ancillary service schedules, and real-time market prices for energy and ancillary services. RTSCED is run every t_{RTD} minutes throughout the day, at interval length of I_{RTD} for a horizon of H_{RTD} intervals.

The AGC sub-model is a rule-based algorithm that is the final line of scheduling defense to control resources to reduce the imbalance (Jaleeli et al., 1992). Unlike SCUC and SCED, it is not optimizing the scheduling of units based on costs. Instead it uses all units that are providing regulating reserve as scheduled by RTSCED to assist in correcting the ACE. ACE is the system imbalance, calculated by subtracting the total load from the total generation (this would be the same as ACE in practice assuming frequency response is equal to the frequency bias). In most cases, AGC will schedule the ACE correction needed, proportional to the units regulating schedules and ramp rates provided by the last RTSCED. AGC is run at the highest time resolution, t_{AGC}. Units not providing ACE regulation are given a schedule by AGC that interpolates one RTSCED dispatch schedule to the next. The Actuals sub-model then calculates the realized generation output. This is determined by the prior AGC schedule and the resource's behavior rate (i.e., how well it follows schedule). As seen in Figure 7, AGC uses the dispatch and reserve schedules from RTSCED as input. The RTSCED regulation schedules determine how each resource is utilized by AGC. AGC provides AGC schedules and Actuals provide realized generation as output. Also, the realized generation is used as input in all other sub-models. Production costs are also calculated using realized generation from the Actuals sub-model.

Note that if, for example, AGC is run at 4-second intervals, the AGC will be run 21,600 times for a single-day simulation. The rule-based algorithm without optimization allows for this operation to occur and is most closely aligned to AGC programs used in ISO/RTO/TSO operations. However, this still requires substantial amounts of data for simulation.

Finally, the SCRPU is an additional sub-model that is often utilized in practice by system operators. The SCRPU uses the same optimization formulation as RTSCUC, but may have different horizon (H_{RPU}) and interval length (I_{RPU}). Its primary difference is that it triggered based on condition rather than time. In addition, both commitments and dispatch can be set as binding decisions. Typical conditions to trigger SCRPU include a contingency occurrence or ACE being higher than a certain threshold. The condition that triggers SCRPU can be configured by the user by entering information into RPU_TRIGGER. Users can also disable the SCRPU which may often be the case when only evaluating default scheduling procedures.

3.2 Integration of Sub-Models

The inter-temporal coupling of sub-models is important because of the configurable timing parameters and to ensure a realistic study with meaningful results. Figure 10 through shows how the different sub-models are integrated within the FESTIV model. The blocks represent the running of the sub-models. The points ahead represent the scheduling horizon of what times each model is scheduling for. I represents interval length, t represents the time between updates, P represents sub-model completion time, and H represents scheduling horizon. For DASCUC, the interval resolution is I_{DAC}, which in today's systems would normally be one hour. DASCUC is performed every t_{DAC} which is usually once per day in today's systems (not shown in Figure 10), and would usually have an optimization horizon (H_{DAC}) of 24 hours. RTSCUC is repeated every t_{RTC} at an interval resolution of I_{RTC} and optimization horizon of H_{RTC}. RTSCED is repeated every t_{RTD} at an interval resolution of I_{RTD} and optimization horizon of H_{RTD}. RTSCUC takes P_{RTC} minutes to solve and RTSCED takes P_{RTD} minutes to solve. These P values are accounting for the fact that the system conditions can change between when sub-model inputs are received and the solution is found and represents the amount of time for the model to solve as well as for inputs and outputs to be processed and communicated. For RTSCUC, start-ups are only binding if there is no more time to change the decision (i.e., the unit must start now to be at minimum capacity at some time in the scheduling horizon due to its start-up time). For RTSCED, the first interval provides the only binding dispatch schedules and reserve schedules with all others being advisory. AGC is run every t_{AGC}. H_{AGC} and I_{AGC} are equal to t_{AGC}.

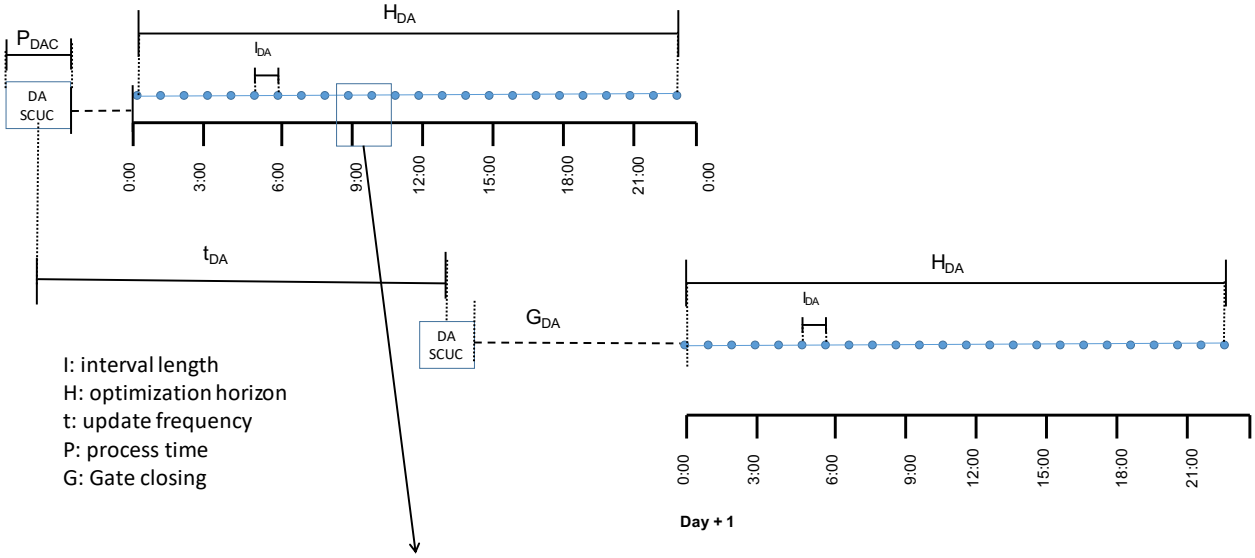


Figure 8: FESTIV time-based parameters. I: interval resolution, t: time between updates, P: model completion time, and H: horizon length.

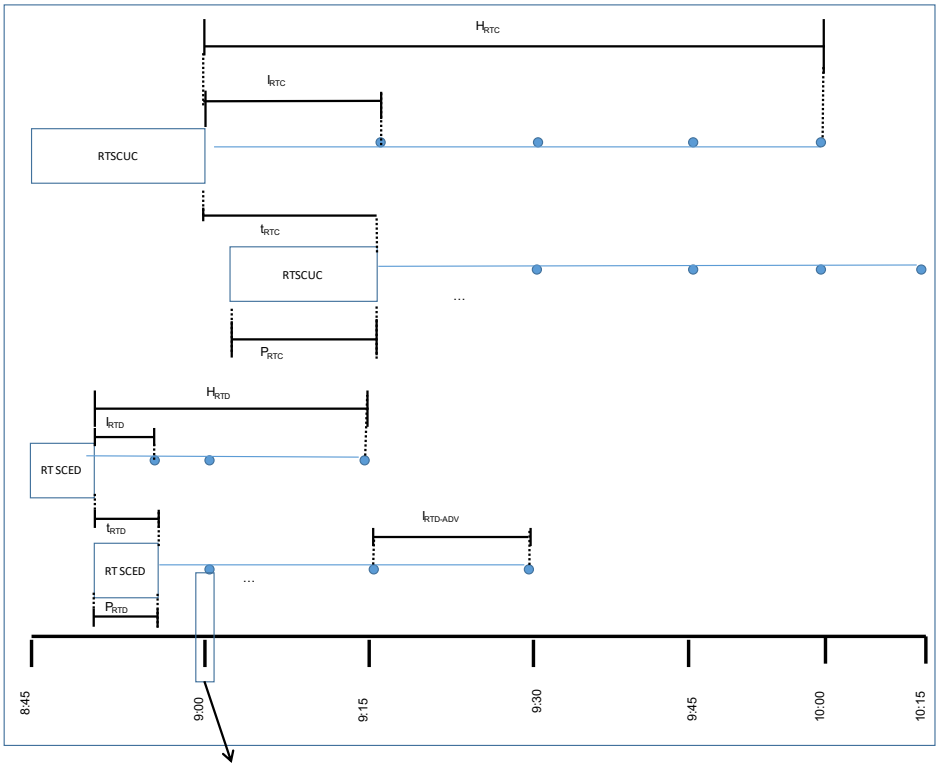


Figure 9: FESTIV time-based parameters. I: interval resolution, t: time between updates, P: model completion time, and H: horizon length.

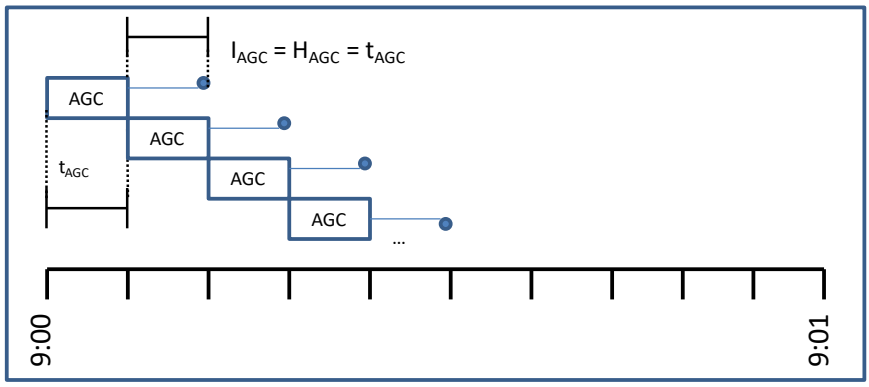


Figure 10: FESTIV time-based parameters. I: interval resolution, t: time between updates, P: model completion time, and H: horizon length.

Each of these time-based parameters has different importance in the studying of variability and uncertainty at multiple timescales. For example, the length of I will determine how much variability the sub-model will be able to prepare for. The length of t will determine how often forecasted conditions are updated and therefore impact how frequently the uncertainty impacts can be resolved. The length of P will determine what forecast inputs were used and what other decisions previously made are available to influence the new

decisions. Finally, the length of H will determine how much of the future is considered in order to make more efficient and reliable decisions in advance, affecting both variability and uncertainty impacts.

The sub-models differ in what they are attempting to accomplish (e.g., commitment vs. dispatch vs. control) and are at different time resolutions and horizons. However, all efforts should ensure that constraints affecting the ultimate outcomes must be reflected consistently in all sub-models. The previously made decisions should be used as input in current sub-models, and decisions made at different time resolutions should be considered when fine tuning at higher time resolution. For example, the different time resolutions for each sub-model are exemplified in the production of one unit in Figure 11 where a perfect forecast was assumed. This simulation used in minutes: I_{DAC} = 60, I_{RTC} = 15, and I_{RTD} = 5; and I_{AGC}= 6 seconds. Each sub-model must ensure the resources are in the correct mode (e.g., start-up, shut-down, emergency, and normal) by linking decisions of other sub-models with the correct interval resolutions and optimization horizons. With perfect forecasts assumed, the only difference in schedules should be based on the variability needs occurring at the different time resolutions. Figure 12 shows a similar chart where an LMP comparison of DAM and RTM are shown for when models are not harmonized between the different interval resolutions (left), and where they have been harmonized (right). With perfect forecasts, the prices between DAM and RTM should be on average converged in a multi-timescale simulation model, if correct harmonization between the models has been accomplished.

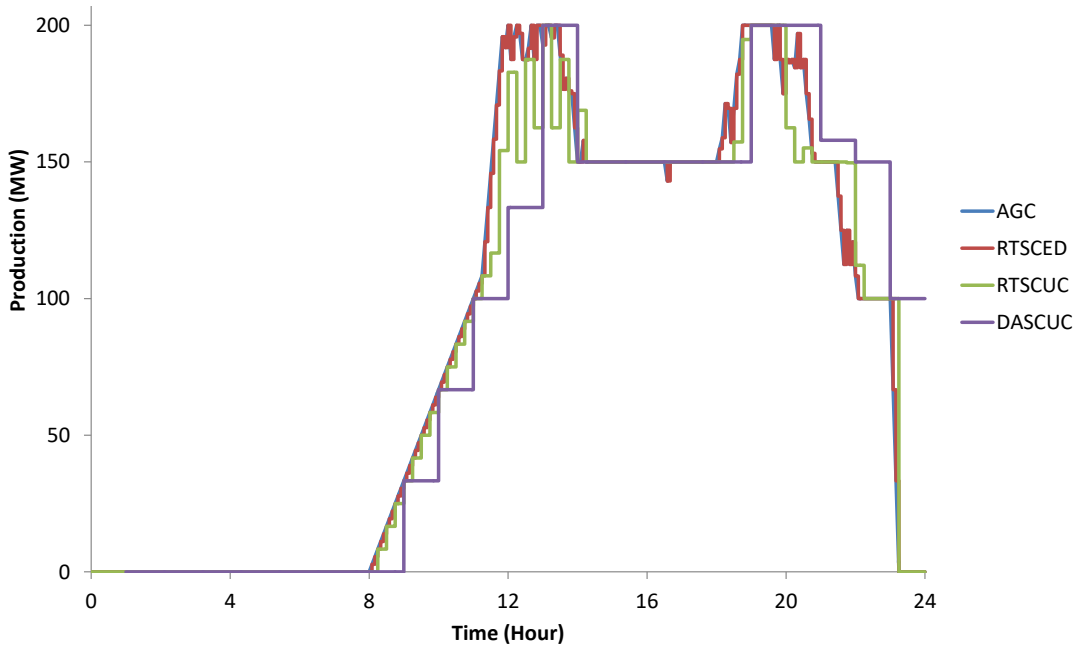


Figure 11: Example generation production from all sub-models

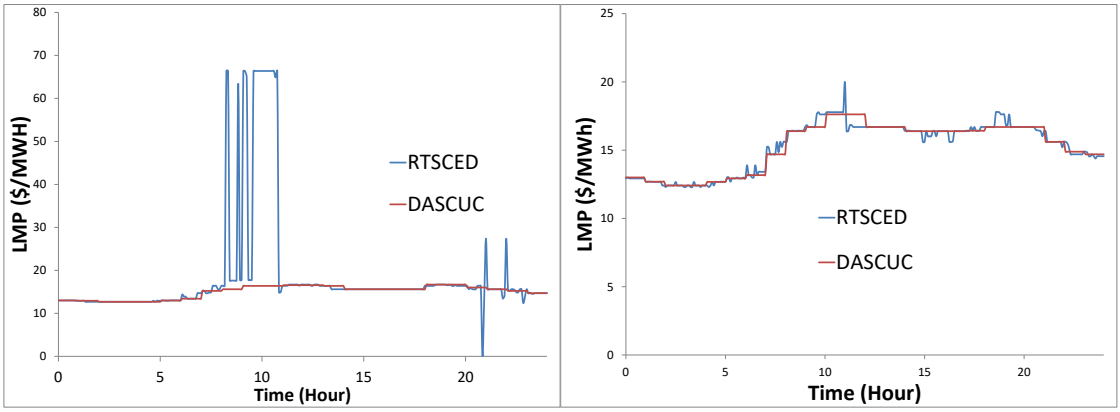


Figure 12: Comparison of DAM and RTM LMPs using perfect forecasts for non-harmonized (left) and harmonized (right) sub-models.

The results of sub-models are continuously communicated to other sub-models throughout the simulation. The model ensures that the results of one sub-model are incorporated into the inputs and constraints of others. The realized generation of units should be known for the RTSCUC, RTSCED, and AGC sub-models so that they do not give infeasible schedules. For example, RTSCED must know both the realized generation from AGC of the units it is scheduling as well as their prior RTSCED dispatch schedule. In practice, the RTSCED and RTSCUC sub-models would require time to solve (P) and they are solving for points ahead (I). With this information, schedules will be feasible based on where the unit is operating at the time the sub-model starts and based on the predicted direction that it is moving toward while the sub-model is solving (i.e., based on its last update, t) considering its ramp rate. For example, Figure 13 shows the operating range (shaded region) that the next RTSCED can schedule a unit based on its realized output at the start of the RTSCED initialization and the prior RTSCED dispatch schedule. With this implementation, units that are given AGC schedules that oppose the direction of RTSCED dispatch schedules (due to unforeseen ACE), or have poor schedule behavior, are given feasible dispatch schedules. This procedure is very important for realistic integration of scheduling sub-models at multiple timescales. Without this communication between sub-models, infeasible outputs (outside of the shaded region) would result.

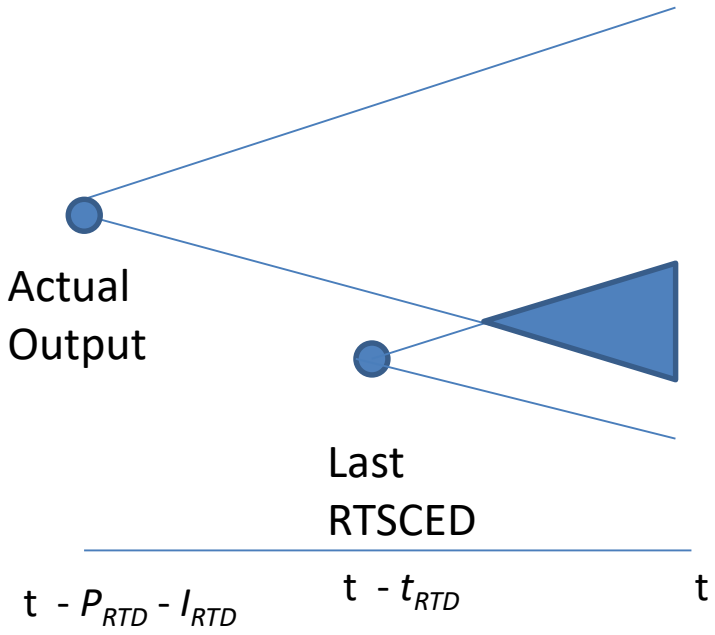


Figure 13: RTSCED dispatch range considering communication of realized output and last schedule from prior RTSCED.

FESTIV Inputs

There are a number of inputs that must be stored before the start of a FESTIV run. These inputs can either be stored in Microsoft Excel format or H5 format. This section mainly deals with creating/building the inputs, and uses the PJM 5 Bus System as an example. A spreadsheet, referred to as the *Main Input File,* holds the majority of the static input data and references used for the study. In addition to the *Main Input File*, there are additional spreadsheets that contain specific timeseries data including those for load, variable capacity or variable generation data, interchanges, and reserve requirement levels. When using H5 format, the Main Input File and time series data are all included in the same file.

4.1 Main Input File

The *Main Input File* contains detailed static information such as system information, bus data, load connections, generator parameters, generator connections, generator costs, branch data, operating reserve parameters, energy storage data (if applicable), generator start-up data (if applicable), and references for the *timeseries inputs*. The following tables itemize the required data in order to create the *Main Input File*.

System Information:

The following data is inserted into the “SYSTEM” Tab on the *Main Input File.*

Table 1: System Information details for FESTIV Model (SYSTEM Tab)

Name	Description	Format
SLACK_BUS	Identification of the slack bus	[bus index]
MVA_PERUNIT	MVA Base for Per Unit Calculations	[MVA]
VOLL	Value of lost load. This is the penalty for involuntary load shedding.	[$/MWh]

FREQUENCY	Nominal Frequency of system (for frequency response studies)	[Hz]
VOIRAMP	Value of insufficient ramp for individual units. This is a penalty variable for unit ramp violations.	[$/MW]

Example SYSTEM tab for the PJM 5 BUS System is shown below. The SLACK_BUS must be an integer less than or equal to the number of buses on the system, but greater than or equal to 1. MVA_PERUNIT is used when performing the DC Power flow and reflects the MVA base of the provided network data. Note that all the reactances in BRANCHDATA should be expressed in this base (100 MVA is typically used for most systems). VOLL should be a large number to ensure load is not shed. This number is also used for penalties involving too much generation that cannot be reduced and is higher than the load. VOIRAMP should also be a very high number to reflect that ramping penalties should not occur.

Table 2: SYSTEM Sheet Information for PJM 5 BUS System

	A	B
1	SLACK_BUS	1
2	MVA_PERUNIT	100
3	VOLL	10000
4	FREQUENCY	60

5	VOIRAMP	10000

Bus Information:

The following data is inserted into the "BUS" Tab on the *Main Input File.*

Name	Description	Format
BUSES	A list of all buses that are part of the system	[bus names]

Example BUS Tab is shown below for the PJM 5 BUS System. Note that all bus names must start with a letter and cannot start with a number (e.g., 5A cannot be a bus name, A5 can be used). Each bus name should also be unique.

Table 3: BUS Sheet Information for PJM 5 BUS System

	A
1	BUSES
2	A
3	B
4	C
5	D
6	E

Load Distribution Information:

The following data is inserted into the "LOAD_DIST" Tab on the *Main Input File.*

Name	Description	Format

BUS	A list of all buses that have load withdrawals	[bus names]
LOAD	Percentage of total load located at each bus	[number between 0 and 1]

Example LOAD_DIST Tab is shown below for the PJM 5 BUS System. Note that all bus names on this tab must be included in the BUS tab. The sum of all distributions should equal 1.

Table 4: LOAD_DIST Sheet Information for PJM 5 BUS System

	A	B
1	BUS	LOAD
2	B	0.33
3	C	0.33
4	D	0.34

Generator Information:

The following data is inserted into the "GEN" Tab on the *Main Input File.*

Table 5: Generator Information details for FESTIV Model (GEN Tab)

Information	Description	Format
CAPACITY	Maximum capacity level of generator	[MW]
NO_LOAD_COST	Cost at no load	[$/h]
STARTUP_COST	Cost associated with starting up a generator (see Table 12 for variable startup)	[$/start]

4.1 – Main Input File

Information	Description	Format
MIN_RUN_TIME	Minimum amount of time required for the generator to remain online once it is turned on	[hours]
MIN_DOWN_TIME	Minimum amount of time required for the generator to remain off once it is turned off	[hours]
RAMP_RATE	Amount of power a generator can change its output within one minute	[MW/min]
MIN_GEN	Minimum stable operating level of generator	[MW]
GEN_TYPE	Type of generator	[#]*
STARTUP_TIME	Amount of time required for a generator to reach its minimum stable operating level from offline	[hours]
SHUTDOWN_TIME	Amount of time required for a generator to turn off completely	[hours]
AGC Capability	Generator capability of responding to an AGC signal	[1 if capable/0 if not]
MAX_STARTS	Maximum number of times a generator can start-up in one day	[#start-ups / day]
INITIAL_STATUS	The initial online status of a generator one interval before the simulation begins	[1/0]
INITIAL_HOUR	The number of intervals a generator has been in its initial state before the simulation begins	[hours]
INITIAL_MW	The initial output of a generator one interval before the simulation begins	[MW]
FORCED_OUTAGE_RATE	The probability that a unit will be forced out.	[percentage]
MTTR	Mean Time to Repair. The average time it takes for a unit to repair from a forced outage	[hours]

Information	Description	Format
VARIABLE_STARTUP	Whether a unit uses variable start-up costs based on how long it has been off-line	[1 if yes, 0 if no]
BEHAVIOR_RATE	How well a generator follows control signals (1 – perfect, 0 – completely random)	[0-1]
GEN_AGC_MODE	Individual AGC mode of generator (see 4.3 FESTIV User Interface)	[integers 1-4]

Example GEN Tab is shown in Table 6 for the PJM 5 BUS System. If a generator has VARIABLE_STARTUP of 1, STARTUP_COST in this tab will be ignored, and VARIABLE_STARTUP tab will be used (Table 12). MIN_RUN_TIME should be greater than or equal to the unit's STARTUP_TIME and SHUTDOWN_TIME since the start-up and shut-down processes are part of its online time. The INITIAL_STATUS, INITIAL_HOUR, and INITIAL_MW parameters are only used in the initial DASCUC and must be updated depending on what resolution of DASCUC is used (I_{DAC}). FORCED_OUTAGE_RATE and MTTR are only used when simulating contingencies.

Table 6: GEN Sheet Information for PJM 5 BUS System (NOTE: this table is transposed)

	1	2	3	4	5	6	7
A		ALTA	PARKCITY	SOLITUDE	SUNDANCE	BRIGHTON	WINDY_HILL
B	CAPACITY	110	100	520	200	600	125

4.1 – Main Input File

	1	2	3	4	5	6	7
C	NO_LOAD_COST	100	100	100	100	200	0
D	STARTUP_COST	450	900	300	150	1200	0
E	MIN_RUN_TIME	4	4	6	1	8	0
F	MIN_DOWN_TIME	4	4	6	1	8	0
G	RAMP_RATE	2	2	5	5	0.5	250
H	MIN_GEN	40	40	100	50	200	0
I	GEN_TYPE	1	1	1	2	1	7
J	STARTUP_TIME	3	3	3	0.5	6	0.01
K	SHUTDOWN_TIME	0.25	0.25	0.5	0.5	0.25	0.01
L	AGC_QUALIFIED	1	1	1	1	1	0
M	MAX_STARTS	3	3	3	4	2	24
N	INITIAL_STATUS	1	1	1	0	1	1
O	INITIAL_HOUR	3	5	4	2	24	24
P	INITIAL_MW	110	100	190	0	400	25
Q	FORCED_OUTAGE_RATE	0.15	0.04	0.04	0.04	0.04	0
R	MTTR	8	8	8	8	12	1
S	VARIABLE_STARTUP	0	0	1	0	0	0
T	BEHAVIOR_RATE	1	1	1	1	1	1

4.1 – Main Input File

	1	2	3	4	5	6	7

	1	2	3	4	5	6	7
Z	GEN_AGC_MODE	1	1	2	1	1	1

Table 7 shows the different generator types currently used in FESTIV. The GEN_TYPE can be used for display purposes after the simulation is completed. Additionally, different GEN_TYPEs may have different treatment within the different sub-models. We provide high-level description of the different treatment in the table. Note that additional GEN_TYPEs can be made by the user and that the user can use the GEN_TYPE to add functional mods or formulation mods.

Table 7: Generator Type details for FESTIV Model

#	GEN_TYPE	Description
1	STEAM	Thermal steam plant, All gen costs, commitment, and ramp rates constrained
2	CT	Currently treated same as GEN_TYPE 1.

#	GEN_TYPE	Description
3	COMBINED CYCLE	Currently treated same as GEN_TYPE 1.
4	HYDRO	Currently treated same as GEN_TYPE 1.
5	NUCLEAR	Currently treated same as GEN_TYPE 1.
6	PUMPED STORAGE	Storage constraints are modeled with commitments needed for gen and pump modes
7	WIND	Variable maximum output for all sub-models.
8	ESR	Storage constraints are modeled without commitment constraints
9	LESR	Not fully functional in default models
10	PV	Generally modeled same as GEN_TYPE 7
11	CSP	Not fully functional in default models
12	Demand Response	Currently modeled same as GEN_TYPE 1
13	VIRTUAL GEN	Not fully functional in default models
14	INTERCHANGE	Interchange schedules that can be positive or negative. The input timeseries is enforced exactly as provided.
15	OUTAGED	Unit is out for entire study period.

#	GEN_TYPE	Description

16	VARIABLE CAPACITY GENERATORS	Variable maximum output for all sub-models except for AGC. Resources that can use changing maximum levels for dispatch but not at an AGC level (e.g., Hydro and storage with pre-determined schedules from separate models).

Genbus Information:

The following data is inserted into the "GENBUS" Tab on the *Main Input File.*

Table 8: Generator Connection Information details for FESTIV Model (GENBUS tab)

Information	Description	Format
GENBUS	The generator and bus connection information	["Bus name"."Gen Name"]
INJECTION_FACTOR	The contribution of generator to bus. Most of the time this should read 1, unless the generator is connected to multiple buses (e.g., DER Aggregations)	[number between 0 and 1]

Example GENBUS Tab is shown below for the PJM 5 BUS System. The Sum of participation factors for any generator should be equal to 1.

Table 9: GENBUS Sheet Information for PJM 5 BUS System

	A	B
1	GENBUS	PARTICIPATION_FACTOR
2	A.ALTA	1
3	E.BRIGHTON	1
4	A.PARKCITY	1

5	C.SOLITUDE	1
6	D.SUNDANCE	1
7	E.WINDY_HILL	1

Generator Cost Information:

The following data is inserted into the "COST" Tab on the *Main Input File.*

Table 10: Generator Cost Information details for FESTIV Model (COST tab)

Information	Description	Format
COSTn	The linear incremental cost of the energy in block n	[$/MWh]
MWn	The cumulative capacity limit of block n.	[MW]

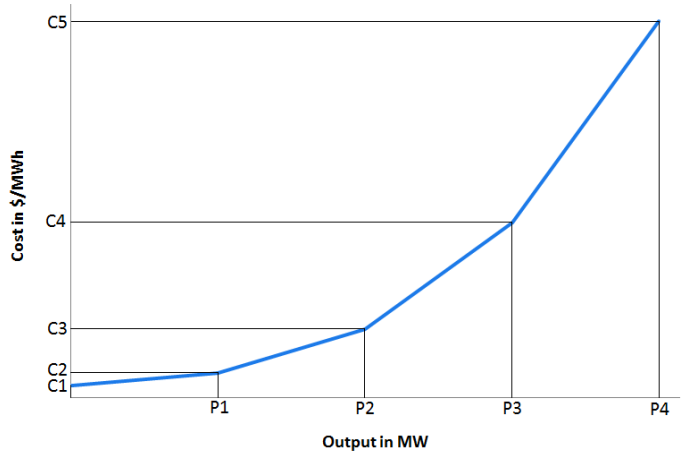


Figure 14: Linearized Generator Cost Curve

Example COST Tab is shown below for the PJM 5 BUS System. FESTIV typically uses four piecewise linear segments for costs and segment capacity, but the user can include as many as he/she would like (note more segments will impact computation time). If less than the number is used, then all subsequent segments should be left blank. The MWn value is cumulative so that MWn must be greater than or equal to MWn-1. FESTIV currently only allows monotonically increasing piecewise linear cost curves.

Table 11: COST Sheet Information for PJM 5 BUS System

	A	B	C	D	E	F	G	…	…

1		COST1	MW1	COST2	MW2	COST3	MW3	COST4	MW4
2	ALTA	14	110						
3	PARKCITY	15	100						
4	SOLITUDE	28	300	30	400	32	500	75	520
5	SUNDANCE	40	200						
6	BRIGHTON	10	600						
7	WINDY_HILL	0	125						

Variable Start-up Cost Information:

The following data is inserted into the "STARTUP" Tab on the *Main Input File.*

Table 12: Variable Start-up Cost Information details for FESTIV Model (STARTUP Tab)

Information	Description	Format
OFF_HOT	The minimum off-line time for a hot start-up	Hours
COST_HOT	The cost associated with a hot start-up	$/Start
OFF_WARM	The minimum off-line time for a warm start-up	Hours
COST_WARM	The cost associated with a warm start-up	$/Start
OFF_COLD	The minimum off-line time for a cold start-up	Hours
COST_COLD	The cost associated with a cold start-up	$/Start

Example COST Tab is shown below for the PJM 5 BUS System.

Table 13: STARTUP Sheet Information for PJM 5 BUS System

	A	B	C	D	E	F	G

1		OFF_HOT	COST_HOT	OFF_WARM	COST_WARM	OFF_COLD	COST_COLD
2	SOLITUDE	1	200	3	300	6	800

Storage Information:

The following data is inserted into the "STORAGE" Tab on the *Main Input File.*

Table 14: Storage Parameters in the FESTIV Model (STORAGE Tab)

Information	Description	Format
MAX_PUMP	Maximum pumping limit for an energy storage unit	[MW]
MIN_PUMP	The minimum stable pumping limit for an energy storage unit	[MW]
MIN_PUMP_TIME	The minimum required time an energy storage unit must pump when it begins pumping	[hours]
PUMP_STARTUP_TIME	The amount of time it takes for an energy storage unit to reach the minimum stable pumping limit	[hours]
PUMP_SHUTDOWN_TIME	The amount of time it takes for an energy storage unit to completely stop pumping	[hours]
PUMP_RAMP_RATE	The amount of power an energy storage unit can change its output within one minute	[MW/min]
INITIAL_SOC	The initial amount of storage available one interval before the optimization begins	[MWh]
FINAL_SOC	If enforcing final storage, what the reservoir level at the end of the first DASCUC Optimization window will be required to end with.	[MWh]
SOC_MAX	The maximum amount of SOC in the reservoir possible for an energy storage unit	[MWh]

Information	Description	Format
EFFICIENCY	The round trip efficiency of an energy storage unit	[%]
RESERVOIR_VALUE	The dollar value associated with any storage value still available at the end of the optimization	[$/MWh]
INITIAL_PUMP_STATUS	The initial status of an energy storage unit one interval before the optimization begins	[1 pumping/0 off]
INITIAL_PUMP_MW	The amount of time an energy storage unit has been in its initial state before the optimization begins	[hours]
INITIAL_PUMP_HOUR	The initial output of an energy storage unit one interval before the optimization begins	[MW]
VARIABLE_EFFICIENCY	Whether the model should incorporate efficiency that is a function of pumping and generator levels or is constant. If VARIABLE_EFFICIENCY = 1, see GEN_EFFICIENCY and PUMP_EFFICIENCY tabs.	[1 variable efficiency, 0 constant efficiency]
ENFORCE_FINAL_SOC	Whether the final storage is enforced in the DASCUC Optimization.	[1 enforce final storage, 0 do not enforce]

Example STORAGE Tab is shown below for an example system. A storage unit cannot have INITIAL_PUMP_STATUS of 1, if it also has INITIAL_STATUS of 1 (i.e., it cannot be generating and pumping at the same time). MIN_PUMP_TIME should be greater than or equal to the unit's PUMP_STARTUP_TIME and PUMP_SHUTDOWN_TIME since the start-up and shut-down processes are part of its online time. If ENFORCE_FINAL_STORAGE = 0, then FINAL_STORAGE is ignored and RESERVOIR_VALUE is the only control of how much storage will be utilized. If VARIABLE_EFFICIENCY = 1, then EFFICIENCY is ignored, and GEN_EFFICIENCY and PUMP_EFFICCIENCY tabs are used for efficiency instead (**Error! Reference source not found.** – **Error! Reference source not found.**).

Table 15: STORAGE Sheet Information for PJM 5 BUS System (NOTE: this table is transposed)

	1	2	3	…	999	1000
A		STORAGE_1	[STORAGE Name]	…	[STORAGE Name]	[STORAGE Name]
B	MAX_PUMP	200				
C	MIN_PUMP	200				
D	MIN_PUMP_TIME	1				
E	PUMP_STARTUP_TIME	0.5				
F	PUMP_SHUTDOWN_TIME	0.25				
G	PUMP_RAMP_RATE	7				
H	INITIAL_STORAGE	2000				
I	FINAL_STORAGE	2000				
J	STORAGE_MAX	3200				
K	EFFICIENCY	0.8				
L	RESERVOIR_VALUE	25				
M	INITIAL_PUMP_STATUS	1				
N	INITIAL_PUMP_MW	200				
O	INITIAL_PUMP_HOUR	3				
P	VARIABLE_EFFICIENCY	0				
Q	ENFORCE_FINAL_STORAGE	1				

Variable efficiency information for energy storage resources:

Branch Information:

The following data is inserted into the “BRANCHDATA” Tab on the *Main Input File.*

Table 16: Branch Information details for FESTIV Model (BRANCHDATA Tab)

Information	Description	Format
Names of Branches in the System	Unique IDs of each branch in the system	[branch name]
BRANCHBUS	Branch connection information including “from” bus and “to” bus.	[“branch name.frombus.tobus”]
REACTANCE	Reactance of each branch	[pu]
RESISTANCE	Resistance of each branch	[pu]
LINE_RATING	Loading limit of a branch under normal operating conditions	[MW]
STE_RATING	Loading limit of a branch under contingency operating conditions	[MW]
PHASE_SHIFTER_ANGLE_LOW	Minimum phase shifter angle	[Degrees]
PHASE_SHIFTER_ANGLE_HIGH	Maximum phase shifter angle	[Degrees]
CTGC_MONITOR	Whether necessary to monitor this branch as a contingency constraint	[1 monitor/0 do not]
BRANCH_TYPE	What type of branch	[branch_type identifier]

Example BRANCH Tab is shown below for the PJM 5 BUS System. Bus names must be part of BUS tab. The flows on lines will always be described in terms of flowing in the direction of “from bus” to “to bus”. LINE_RATING is used for network constraints and STE_RATING is used for contingency constraints. CTGC_MONITOR is whether the failure of that branch should be modeled as a contingency constraint.

4.1 – Main Input File

Table 17: BRACHDATA Sheet Information for PJM 5 BUS System

	A	B	C	D	E	F	G	H	I	J	K
1		BRANCHBUS		REACTANCE	RESISTANCE	LINE_RATING	STE_RATING	PHASE_SHIFTER_ANGLE_LOW	PHASE_SHIFTER_ANGLE_HIGH	CTGC_MONITOR	BRANCH_TYPE
2	BRANCH1	BRANCH1.A.B	BRANCH1	0.0281	0.00281	1000	1000			0	1
3	BRANCH2	BRANCH2.A.D	BRANCH2	0.0304	0.00304	1000	1000			0	1
4	BRANCH3	BRANCH3.A.E	BRANCH3	0.0064	0.00064	1000	1000			0	1
5	BRANCH4	BRANCH4.B.C	BRANCH4	0.0108	0.00108	1000	1000			0	1
6	BRANCH5	BRANCH5.C.D	BRANCH5	0.0297	0.00297	1000	1000			0	1
7	BRANCH6	BRANCH6.D.E	BRANCH6	0.0297	0.00297	230	250			0	1

The table below displays the current BRANCH_TYPE available for FESTIV input.

Table 18: Branch Type details for FESTIV Model

#	BRANCH_TYPE	Description
1	TRANSMISSION LINE	A normal AC transmission line.
2	PAR_FIXED or Unidirectional	A phase angle regulator transformer which has a fixed angle or can only have a positive angle.

3	PAR_ADJ	A phase angle regulator transformer which has an angle that can be optimized between a negative minimum and positive maximum
4	HVDC	High Voltage Direct Current line that acts as a sink at from bus and source at to bus.

Ancillary Service Information:

The following data is inserted into the "RESERVEPARAM" Tab on the *Main Input File.*

Table 19: Ancillary Service Information details for FESTIV Model (RESERVEPARAM tab)

Information	Description	Format
RESERVE_TYPE	Type of reserve product	[reserve name]
RESERVE_ON	Whether the units providing this reserve must be online (or pumping) in order to provide	[1 needs to be online, 0 is offline]
RESERVE_TIME	Time horizon of reserve product being considered	[minutes]
RESERVE_DIR	Operating direction of the reserve product (1=up, 2=down, 3=both directions)	[1/2/3]
RESERVE_AGC	Whether the reserve product requires AGC capability	[1 needs to be agc/0 does not]
RESERVE_GOV	Whether the reserve product requires governor action	[1/0]

RESERVE_INERTIA	Whether the reserve is for the inertia ancillary service	[1/0]
RESERVE_INCLUSIVE	Which other reserve type can contribute to this requirement	[index of other reserve type, 0 if none]
RESERVE_VG	Whether VG will be allowed to provide the reserve type	[1/0]
VOIR	The penalty associated with insufficient reserve to meet the reserve requirement	[$/MW-h]

Example RESERVEPARAM Tab is shown on the following page for the PJM 5 BUS System. Note that any reserve type that includes other reserves should have a requirement that is greater than or equal to the reserve type that it includes.

Table 20: RESERVEPARAM Sheet Information for PJM 5 BUS System

	A	B	C	D	E	F	G	H	I	J
1		RESERVE_ON	RESERVE_TIME	RESERVE_DIR	RESERVE_AGC	RESERVE_GOV	RESERVE_INERTIA	RESERVE_INCLUSIVE	RESERVE_VG	VOIR
2	SPIN	1	10	1	0	0	0	0	0	5000
3	NON_SPIN	0	10	1	0	0	0	1	0	1500
4	REGULATION	1	5	3	1	0	0	0	0	2500

5	REPLACEMENT 1	1	30	1	0	0	0	2	0	500
6	REPLACEMENT 2	0	30	1	0	0	0	4	0	500

Ancillary Service Costs Information:

The following data is inserted into the "ASC" Tab on the *Main Input File.*

Information	Description	Format
Ancillary Service Costs	Costs by gen to provide each reserve_type.	$/MW-h

Example ASC Tab is shown below for the PJM 5 BUS System.

Table 21: ASC Sheet Information for PJM 5 BUS System

	A	B	C	D	E	F
1		SPIN	NON_SPIN	REGULATION	REPLACEMENT 1	REPLACEMENT 2
2	ALTA			5		
3	PARKCITY			10		
4	SOLITUDE			4		
5	SUNDANCE			1		
..	BRIGHTON			8		
..	WINDY_HILL					

References to Timeseries data:

A number of additional sheets are part of the *Main Input File* which contains the references to the timeseries data to be used in the study. The following reference tabs are included:

1. ACTUAL_LOAD_REF: Files that contain the actual load data
2. ACTUAL_VG_REF: Files that contain the actual VG data
3. RTD_LOAD_REF: Files that contain the load data used in the RTSCED
4. RTD_VG_REG: Files that contain the VG data used in the RTSCED
5. RTD_RESERVE: Files that contain the reserve data used in the RTSCED
6. RTC_LOAD_REF: Files that contain the load data used in the RTSCUC
7. RTC_VG_REF: Files that contain the VG data used in the RTSCUC
8. RTC_RESERVE: Files that contain the reserve data used in the RTSCUC
9. DA_LOAD_REF: Files that contain the load data used in the DASCUC
10. DA_VG_REF: Files that contain the VG data used in the DASCUC
11. DA_RESERVE_REF: Files that contain the reserve data used in the DASCUC

An example of DA_LOAD_REF is shown below. Each day requires a new input for all reference sheets. These files must be in the same directory as the input file, one level below under the directory "TIMESERIES". Each of the other references follow the same general pattern with the associated files listed under the reference tab. Note that ACTUAL_LOAD_REF must be included in any study in the *Main Input File.* ACTUAL_VG_REF must be included only if there is VG on the system. DAC_LOAD_REF, DAC_VG_REF, RTD_LOAD_REF, RTD_VG_REF, RTC_LOAD_REF, and RTC_VG_REF must have files included if the data_create is chosen as 1 for the associated type (see 4.3 FESTIV User Interface). DAC_RESERVE_REF must be included unless there is no reserve chosen for the simulation. If RESERVEPARAM does include RESERVE_TYPE that is not blank, then each RESERVE_TYPE will have a requirement of 0 for all intervals if there is no files listed under DAC_RESERVE_REF. RTD_RESERVE and RTC_RESERVE must have files included if the reserve_data_create is chosen as 2 (see 4.3 FESTIV User Interface).

Table 22: DA_LOAD_REF tab example for PJM 5 bus example system

	A
1	
2	PJM5BUS_ DASCUC_LOAD_DAY1
3	PJM5BUS_ DASCUC_LOAD_DAY2
4	PJM5BUS_ DASCUC_LOAD_DAY3
5	PJM5BUS_ DASCUC_LOAD_DAY4
6	PJM5BUS_ DASCUC_LOAD_DAY5
7	PJM5BUS_ DASCUC_LOAD_DAY6
8	PJM5BUS_ DASCUC_LOAD_DAY7

4.2 Time series Inputs

In addition to the *Main Input File*, several files are used to represent the timeseries information. In particular, this includes load, variable generation or variable capacity resources, interchange schedules, and reserve requirement levels. Each of the timeseries inputs is now discussed.

Actual Load Data:

Of all the timeseries files the actual load data file(s) are the only that are absolutely required in order for a simulation to be run. The actual load data includes the load that occurs at the final sub-model, AGC. The actual load data includes the TIME (timestamp) in column A and the LOAD in column B. The timestamp MUST be at intervals that are equal to t_{AGC}. Separate files are required for each day of study. An example of the PJM 5 BUS system is shown below. In this example, the AGC is run every 6 seconds, and therefore the intervals in the actual load file should be at 6 second intervals as shown.

Table 23: Actual load data used for PJM 5 bus example

	A	B
1		LOAD
2	0:00:00	853.6
3	0:00:06	854
4	0:00:12	853.8
5	0:00:18	853.4
6	0:00:24	854.2
7	0:00:30	854.4
8	0:00:36	854.5
9	0:00:42	854.6
10	0:00:48	854.2
11	0:00:54	854.4

12	0:01:00	851
…	…	…
14397	23:59:30	971.6
14398	23:59:36	972
14399	23:59:42	971.7
14400	23:59:48	971.5
14401	23:59:54	970.6

Actual VG Data:

If there is VG on the system, then the actual VG data file(s) are required. The actual VG data includes the realized VG output that occurs at the final sub-model, AGC. The actual VG data includes the TIME (timestamp) in column A and the VG realized output in columns B and forward, depending on the number of VG resources. The timestamp MUST be at intervals that are equal to t_{AGC}. Separate files are required for each day of study. An example of the PJM 5 BUS system is shown below. In this example, the AGC is run every 6 seconds, and therefore the intervals in the actual VG file should be at 6 second intervals as shown. In this example, there is only one VG resource. If two VG resources were present, the second VG resource would be on column C and so forth. It is essential that the names of the VG resources in row 1 identically match those of the VG resources in the GEN tab in the *Main Input File.* If there is a mismatch, FESTIV will not recognize the actual VG output of the resource.

Table 24: Actual VG data used for PJM 5 bus example

	A	B
1		WINDY_HILL
2	0:00:00	111.8
3	0:00:06	112.5
4	0:00:12	113
5	0:00:18	112.7

6	0:00:24	112.7
7	0:00:30	112.6
8	0:00:36	112.5
9	0:00:42	112.8
10	0:00:48	112.5
11	0:00:54	112.9
12	0:01:00	112.8
…	…	…
14397	23:59:30	47.9
14398	23:59:36	48.1
14399	23:59:42	48.1
14400	23:59:48	47.3
14401	23:59:54	46.2

RTSCED Load Data:

The RTSCED load data file(s) are required if the *rtd_load_data_create* is equal to 1. The RTSCED load data includes the load that would be forecast for the RTSCED sub-model. The RTSCED load data includes the START_TIME (timestamp) in column A, the TIME_INTERVAL (timestamp) in column B, and the RTSCED Load Forecast in column C. The timestamps in the RTSCED load data must follow the requirements of t_{RTD}, I_{RTD}, $I_{RTD\text{-}ADV}$, and H_{RTD}. An example of the PJM 5 BUS system is shown below.

Table 25: Load forecast timeseries used for RTSCED in the PJM 5 bus example

	A	B	C
1	START_TIME	TIME_INTERVAL	Load
2	23:55:00	0:00:00	675.16

3	23:55:00	0:15:00	674.87
4	23:55:00	0:30:00	675.10
5	23:55:00	0:45:00	673.76
6	23:55:00	1:00:00	669.20
7	0:00:00	0:05:00	674.87
8	0:00:00	0:15:00	675.10
9	0:00:00	0:30:00	673.76
10	0:00:00	0:45:00	669.20
11	0:00:00	1:00:00	664.48
12	0:05:00	0:10:00	675.10
13	0:05:00	0:15:00	673.76
14	0:05:00	0:30:00	669.20
15	0:05:00	0:45:00	664.48
16	0:05:00	1:00:00	659.74
17	0:10:00	0:15:00	673.76
18	0:10:00	0:30:00	669.20
19	0:10:00	0:45:00	664.48
20	0:10:00	1:00:00	659.74
21	0:10:00	1:15:00	656.14
…	…	…	…
1442	23:55	0:00	694.28
1443	23:55	0:15	694.28

1444	23:55	0:30	694.28
1445	23:55	0:45	694.28
1446	23:55	1:00	694.28

In this example, the RTSCED is run every 5 minutes (t_{RTD}=5) with an interval length of 5 minutes (I_{RTD}=5), advisory interval length of 15 minutes ($I_{RTD\text{-}ADV}$=15), and horizon of 5 intervals (H_{RTD}=5). The timestamp rules are described next. For one START_TIME, there should be H_{RTD} number of TIME_INTERVALs. The first TIME_INTERVAL should be I_{RTD} minutes after the START_TIME. The next TIME_INTERVAL will land on the next point that is divisible by $I_{RTD\text{-}ADV}$, and the following intervals will be exactly $I_{RTD\text{-}ADV}$ minutes after the previous. Finally, the next START_TIME in column A should be t_{RTD} minutes after the prior START_TIME. Further explanations of the time intervals are given in Section 3.3. For best results, it is recommended to put in the appropriate time parameters into the FESTIV GUI, run the rtd_load_data_create of 2, and copy columns A and B of the output that is created, insert them into a new spreadsheet, and then insert the appropriate RTSCED Load forecasts. Finally, it is important that the **first day** of the study has one RTSCED that reflects the hour 0 with a start time that is prior to that day. For studies greater than one day, the subsequent days' files do not require this first RTSCED interval, and these would start with START_TIME of 0:00.

RTSCED VG Data:

The RTSCED VG data file(s) are required if the *rtd_vg_data_create* is equal to 1. The RTSCED VG data includes the VG, or Variable Capacity Resource (VCR) that would be forecast for the RTSCED sub-model. The RTSCED VG data includes the START_TIME (timestamp) in column A, the TIME_INTERVAL (timestamp) in column B, and the RTSCED VG Forecast in column C and forward, depending on the number of VG resources. The timestamps in the RTSCED VG data must follow the requirements of t_{RTD}, I_{RTD}, $I_{RTD\text{-}ADV}$, and H_{RTD}. An example of the PJM 5 BUS system is shown on the following page.

Table 26: VG forecast timeseries used for RTSCED in the PJM 5 bus example

	A	B	C

1	START_TIME	TIME_INTERVAL	WINDY
2	23:55:00	0:00:00	239.4
3	23:55:00	0:15:00	239.4
4	23:55:00	0:30:00	239.4
5	23:55:00	0:45:00	239.4
6	23:55:00	1:00:00	239.4
7	0:00:00	0:05:00	239.4
8	0:00:00	0:15:00	239.4
9	0:00:00	0:30:00	239.4
10	0:00:00	0:45:00	239.4
11	0:00:00	1:00:00	239.4
12	0:05:00	0:10:00	239.4
13	0:05:00	0:15:00	239.4
14	0:05:00	0:30:00	239.4
15	0:05:00	0:45:00	239.4
16	0:05:00	1:00:00	239.4
17	0:10:00	0:15:00	226.1
18	0:10:00	0:30:00	226.1
19	0:10:00	0:45:00	226.1
20	0:10:00	1:00:00	226.1
21	0:10:00	1:15:00	226.1
…	…	…	…

1442	23:55	0:00	131.8
1443	23:55	0:15	131.8
1444	23:55	0:30	131.8
1445	23:55	0:45	131.8
1446	23:55	1:00	131.8

In this example, the RTSCED is run every 5 minutes (t_{RTD}=5) with an interval length of 5 minutes (I_{RTD}=5), advisory interval length of 15 minutes ($I_{RTD\text{-}ADV}$=15), and horizon of 5 intervals (H_{RTD}=5). The timestamp rules are described next. For one START_TIME, there should be H_{RTD} number of TIME_INTERVALs. The first TIME_INTERVAL should be I_{RTD} minutes after the START_TIME. The next TIME_INTERVAL will land on the next point that is divisible by $I_{RTD\text{-}ADV}$, and the following intervals will be exactly $I_{RTD\text{-}ADV}$ minutes after the previous. Finally, the next START_TIME in column A should be t_{RTD} minutes after the prior START_TIME. Further explanations of the time intervals are given in Section 3.3. For best results, it is recommended to put in the appropriate time parameters into the FESTIV GUI, run the rtd_vg_data_create of 2, and copy columns A and B of the output that is created, insert them into a new spreadsheet, and then insert the appropriate RTSCED VG forecasts in column C and forward. Finally, it is important that the **first day** of the study has one RTSCED that reflects the hour 0 with a start time that is prior to that day. For studies greater than one day, the subsequent days' files do not require this first RTSCED interval, and these would start with START_TIME of 0:00.

RTSCED Reserve Data:

The RTSCED Reserve data file(s) are required if the *RTD_RESERVE_FORECAST_MODE*

is equal to 1. The RTSCED Reserve data includes the Reserve levels that would be required for the RTSCED sub-model. The RTSCED VG data includes the START_TIME (timestamp) in column A, the TIME_INTERVAL (timestamp) in column B, and the RTSCED Reserve levels in column C and forward, depending on the number of Reserve Types. The timestamps in the RTSCED VG data must follow the requirements of t_{RTD}, I_{RTD}, $I_{RTD\text{-}ADV}$, and H_{RTD}. An example of the PJM 5 BUS system is shown below where the timestamps are in excel time format.

Table 27: Reserve level timeseries used for RTSCED in the PJM 5 bus example

	A	B	C	D	E

1	START_TIME	TIME_INTERVAL	SPIN	NON_SPIN	REGULATION
2	23:55:00	0:00:00	30	60	25
3	23:55:00	0:15:00	30	60	25
4	23:55:00	0:30:00	30	60	25
5	23:55:00	0:45:00	30	60	25
6	23:55:00	1:00:00	30	60	25
7	0:00:00	0:05:00	30	60	25
8	0:00:00	0:15:00	30	60	25
9	0:00:00	0:30:00	30	60	25
10	0:00:00	0:45:00	30	60	25
11	0:00:00	1:00:00	30	60	25
12	0:05:00	0:10:00	30	60	25
13	0:05:00	0:15:00	30	60	25
14	0:05:00	0:30:00	30	60	25
15	0:05:00	0:45:00	30	60	25
16	0:05:00	1:00:00	30	60	25
17	0:10:00	0:15:00	30	60	25
18	0:10:00	0:30:00	30	60	25
19	0:10:00	0:45:00	30	60	25
20	0:10:00	1:00:00	30	60	25
21	0:10:00	1:15:00	30	60	25
…	…	…	…	…	…

1442	23:55	0:00	30	60	25
1443	23:55	0:15	30	60	25
1444	23:55	0:30	30	60	25
1445	23:55	0:45	30	60	25
1446	23:55	1:00	30	60	25

In this example, the RTSCED is run every 5 minutes (t_{RTD}=5) with an interval length of 5 minutes (I_{RTD}=5), advisory interval length of 15 minutes ($I_{RTD\text{-}ADV}$=15), and horizon of 5 intervals (H_{RTD}=5). The timestamp rules are described next. For one START_TIME, there should be H_{RTD} number of TIME_INTERVALs. The first TIME_INTERVAL should be I_{RTD} minutes after the START_TIME. The next TIME_INTERVAL will land on the next point that is divisible by $I_{RTD\text{-}ADV}$, and the following intervals will be exactly $I_{RTD\text{-}ADV}$ minutes after the previous. Finally, the next START_TIME in column A should be t_{RTD} minutes after the prior START_TIME. Further explanations of the time intervals are given in Section 3.3. For best results, it is recommended to put in the appropriate time parameters into the FESTIV GUI, run the rtd_reserve_data_create of 1, and copy columns A and B of the output that is created, insert them into a new spreadsheet, and then insert the appropriate RTSCED Reserve Levels in column C and forward. Finally, it is important that the **first day** of the study has one RTSCED that reflects the hour 0 with a start time that is prior to that day. For studies greater than one day, the subsequent days' files do not require this first RTSCED interval, and these would start with START_TIME of 0:00.

RTSCUC Load Data:

The RTSCUC load data file(s) are required if the *rtc_load_data_create* is equal to 1. The RTSCUC load data includes the load that would be forecast for the RTSCUC sub-model. The RTSCUC load data includes the START_TIME (timestamp) in column A, the TIME_INTERVAL (timestamp) in column B, and the RTSCUC Load Forecast in column C. The timestamps in the RTSCUC load data must follow the requirements of t_{RTC}, I_{RTC}, and H_{RTC}. An example of the PJM 5 BUS system is shown below.

Table 28: Load forecast timeseries used for RTSCUC in the PJM 5 bus example

	A	B	C
1	START_TIME	TIME_INTERVAL	Load

2	23:45	0:00	674.578
3	23:45	0:15	664.475
4	23:45	0:30	651.223
5	23:45	0:45	636.221
6	23:45	1:00	618.903
7	23:45	1:15	607.005
8	23:45	1:30	596.319
9	23:45	1:45	585.272
10	23:45	2:00	575.972
11	23:45	2:15	568.530
12	23:45	2:30	560.790
13	23:45	2:45	554.109
14	0:00	0:15	664.475
15	0:00	0:30	651.223
16	0:00	0:45	636.221
17	0:00	1:00	618.903
18	0:00	1:15	607.005
19	0:00	1:30	596.319
20	0:00	1:45	585.272
21	0:00	2:00	575.972
22	0:00	2:15	568.530
23	0:00	2:30	560.790

24	0:00	2:45	554.109
25	0:00	3:00	547.907
26	0:15	0:30	651.223
27	0:15	0:45	636.221
28	0:15	1:00	618.903
29	0:15	1:15	607.005
30	0:15	1:30	596.319
31	0:15	1:45	585.272
32	0:15	2:00	575.972
33	0:15	2:15	568.530
34	0:15	2:30	560.790
35	0:15	2:45	554.109
36	0:15	3:00	547.907
37	0:15	3:15	545.320
…	…	…	…
1154	23:45	0:00	695.273
1155	23:45	0:15	695.273
1156	23:45	0:30	695.273
1157	23:45	0:45	695.273
1158	23:45	1:00	695.273
1159	23:45	1:15	695.273
1160	23:45	1:30	695.273

1161	23:45	1:45	695.273
1162	23:45	2:00	695.273
1163	23:45	2:15	695.273
1164	23:45	2:30	695.273
1165	23:45	2:45	695.273

In this example, the RTSCUC is run every 15 minutes (t_{RTC}=15) with an interval length of 15 minutes (I_{RTC}=15), and horizon of 12 intervals (H_{RTC}=12). The timestamp rules are described next. For one START_TIME, there should be H_{RTC} number of TIME_INTERVALs. The first TIME_INTERVAL should be I_{RTC} minutes after the START_TIME. The following TIME_INTERVALs should be I_{RTC} minutes after the last. Finally, the next START_TIME in column A should be t_{RTC} minutes after the prior START_TIME. Further explanations of the time intervals are given in Section 3.3. For best results, it is recommended to put in the appropriate time parameters into the FESTIV GUI, run the rtc_load_data_create of 2, and copy columns A and B of the output that is created, insert them into a new spreadsheet, and then insert the appropriate RTSCUC Load forecasts. Finally, it is important that the **first day** of the study has one RTSCUC that reflects the hour 0 with a start time that is prior to that day. For studies greater than one day, the subsequent days' files do not require this first RTSCUC interval, and these would start with START_TIME of 0:00.

RTSCUC VG Data:

The RTSCUC VG data file(s) are required if the *rtc_vg_data_create* is equal to 1. The RTSCUC VG data includes the VG or VCR that would be forecast for the RTSCUC sub-model. The RTSCUC VG data includes the START_TIME (timestamp) in column A, the TIME_INTERVAL (timestamp) in column B, and the RTSCUC VG Forecast in column C and forward, depending on the number of VG resources. The timestamps in the RTSCUC VG data must follow the requirements of t_{RTC}, I_{RTC}, and H_{RTC}. An example of the PJM 5 BUS system is shown below.

Table 29: VG forecast timeseries used for RTSCUC in the PJM 5 bus example

	A	B	C

1	START_TIME	TIME_INTERVAL	WINDY_HILL
2	23:45	0:00	180.254
3	23:45	0:15	180.254
4	23:45	0:30	180.254
5	23:45	0:45	180.254
6	23:45	1:00	180.254
7	23:45	1:15	180.254
8	23:45	1:30	180.254
9	23:45	1:45	180.254
10	23:45	2:00	180.254
11	23:45	2:15	180.254
12	23:45	2:30	180.254
13	23:45	2:45	180.254
14	0:00	0:15	180.254
15	0:00	0:30	180.254
16	0:00	0:45	180.254
17	0:00	1:00	180.254
18	0:00	1:15	180.254
19	0:00	1:30	180.254
20	0:00	1:45	180.254
21	0:00	2:00	180.254
22	0:00	2:15	180.254

23	0:00	2:30	180.254
24	0:00	2:45	180.254
25	0:00	3:00	180.254
26	0:15	0:30	176.782
27	0:15	0:45	176.782
28	0:15	1:00	176.782
29	0:15	1:15	176.782
30	0:15	1:30	176.782
31	0:15	1:45	176.782
32	0:15	2:00	176.782
33	0:15	2:15	176.782
34	0:15	2:30	176.782
35	0:15	2:45	176.782
36	0:15	3:00	176.782
37	0:15	3:15	176.782
…	…	…	…
1154	23:45	0:00	143.8
1155	23:45	0:15	143.8
1156	23:45	0:30	143.8
1157	23:45	0:45	143.8
1158	23:45	1:00	143.8
1159	23:45	1:15	143.8

1160	23:45	1:30	143.8
1161	23:45	1:45	143.8
1162	23:45	2:00	143.8
1163	23:45	2:15	143.8
1164	23:45	2:30	143.8
1165	23:45	2:45	143.8

In this example, the RTSCUC is run every 15 minutes (t_{RTC}=15) with an interval length of 15 minutes (I_{RTC}=15), and horizon of 12 intervals (H_{RTC}=12). The timestamp rules are described next. For one START_TIME, there should be H_{RTC} number of TIME_INTERVALs. The first TIME_INTERVAL should be I_{RTC} minutes after the START_TIME. The following TIME_INTERVALs should be I_{RTC} minutes after the last. Finally, the next START_TIME in column A should be t_{RTC} minutes after the prior START_TIME. Further explanations of the time intervals are given in Section 3.3. For best results, it is recommended to put in the appropriate time parameters into the FESTIV GUI, run the rtc_vg_data_create of 2, and copy columns A and B of the output that is created, insert them into a new spreadsheet, and then insert the appropriate RTSCUC VG forecasts. Finally, it is important that the **first day** of the study has one RTSCUC that reflects the hour 0 with a start time that is prior to that day. For studies greater than one day, the subsequent days' files do not require this first RTSCUC interval, and these would start with START_TIME of 0:00.

RTSCUC Reserve Data:

The RTSCUC Reserve data file(s) are required if the *RTC_RESERVE_FORECAST_MODE* is equal to 1. The RTSCUC Reserve data includes the Reserve levels that would be required for the RTSCUC sub-model. The RTSCED VG data includes the START_TIME (timestamp) in column A, the TIME_INTERVAL (timestamp) in column B, and the RTSCUC Reserve levels in column C and forward, depending on the number of Reserve Types. The timestamps in the RTSCUC VG data must follow the requirements of t_{RTC}, I_{RTC}, and H_{RTC}. An example of the PJM 5 BUS system is shown below where the timestamps are in excel time format.

Table 30: Reserve levels timeseries used for RTSCUC in the PJM 5 bus example

	A	B	C	D	E
1	START_TIME	TIME_INTERVAL	SPIN	NON_SPIN	REGULATION
2	23:45	0:00	30	60	25
3	23:45	0:15	30	60	25
4	23:45	0:30	30	60	25
5	23:45	0:45	30	60	25
6	23:45	1:00	30	60	25
7	23:45	1:15	30	60	25
8	23:45	1:30	30	60	25
9	23:45	1:45	30	60	25
10	23:45	2:00	30	60	25
11	23:45	2:15	30	60	25
12	23:45	2:30	30	60	25
13	23:45	2:45	30	60	25
14	0:00	0:15	30	60	25
15	0:00	0:30	30	60	25
16	0:00	0:45	30	60	25
17	0:00	1:00	30	60	25
18	0:00	1:15	30	60	25
19	0:00	1:30	30	60	25
20	0:00	1:45	30	60	25

21	0:00	2:00	30	60	25
22	0:00	2:15	30	60	25
23	0:00	2:30	30	60	25
24	0:00	2:45	30	60	25
25	0:00	3:00	30	60	25
26	0:15	0:30	30	60	25
27	0:15	0:45	30	60	25
28	0:15	1:00	30	60	25
29	0:15	1:15	30	60	25
30	0:15	1:30	30	60	25
31	0:15	1:45	30	60	25
32	0:15	2:00	30	60	25
33	0:15	2:15	30	60	25
34	0:15	2:30	30	60	25
35	0:15	2:45	30	60	25
36	0:15	3:00	30	60	25
37	0:15	3:15	30	60	25
…	…	…	…	…	…
1154	23:45	0:00	30	60	25
1155	23:45	0:15	30	60	25
1156	23:45	0:30	30	60	25
1157	23:45	0:45	30	60	25

1158	23:45	1:00	30	60	25
1159	23:45	1:15	30	60	25
1160	23:45	1:30	30	60	25
1161	23:45	1:45	30	60	25
1162	23:45	2:00	30	60	25
1163	23:45	2:15	30	60	25
1164	23:45	2:30	30	60	25
1165	23:45	2:45	30	60	25

In this example, the RTSCUC is run every 15 minutes (t_{RTC}=15) with an interval length of 15 minutes (I_{RTC}=15), and horizon of 12 intervals (H_{RTC}=12). The timestamp rules are described next. For one START_TIME, there should be H_{RTC} number of TIME_INTERVALs. The first TIME_INTERVAL should be I_{RTC} minutes after the START_TIME. The following TIME_INTERVALs should be I_{RTC} minutes after the last. Finally, the next START_TIME in column A should be t_{RTC} minutes after the prior START_TIME. Further explanations of the time intervals are given in Section 3.3. For best results, it is recommended to put in the appropriate time parameters into the FESTIV GUI, run the rtc_reserve_data_create of 1, and copy columns A and B of the output that is created, insert them into a new spreadsheet, and then insert the appropriate RTSCUC Reserve Levels in column C and forward. Finally, it is important that the **first day** of the study has one RTSCUC that reflects the hour 0 with a start time that is prior to that day. For studies greater than one day, the subsequent days' files do not require this first RTSCUCD interval, and these would start with START_TIME of 0:00.

DASCUC Load Data:

The DASCUC load data file(s) are required if the *dac_load_data_create* is equal to 1. The DASCUC load data includes the load that would be forecast for the DASCUC sub-model. The DASCUC load data includes the DASCUC index (e.g., the number of the DASCUC run) in column A, the INTERVAL in units of hours in column B, and the DASCUC Load Forecast in column C. The timestamps in the DASCUC load data must follow the requirements of t_{DAC}, I_{DAC}, and H_{DAC}. An example of the PJM 5 BUS system is shown in Table 31.

Table 31: DASCUC LOAD FORECAST TIMESERIES Data

	A	B	C
1	DASCUC	INTERVAL	LOAD
2	1	0	849.6
3	1	1	848.9
4	1	2	849.9
5	1	3	859.2
6	1	4	876.1
…	…	…	…
24	1	22	1025.6
25	1	23	986.3

In this example, the DASCUC is run once a day (t_{DAC}=24) with an interval length of 1 hour (I_{DAC}=1), and horizon of 24 intervals (H_{DAC}=24). For one DASCUC index, there should be H_{DAC} number of INTERVALs. Each INTERVAL should be I_{DAC} hours apart. Further explanations of the time intervals are given in Section 3.3.

DASCUC VG Data:

The DASCUC load data file(s) are required if the *dac_vg_data_create* is equal to 1. The DASCUC load data includes the VG or VCR that would be forecast for the DASCUC sub-model. The DASCUC load data includes the DASCUC index (e.g., the number of the DASCUC run) in column A, the INTERVAL in units of hours in column B, and the DASCUC VG or VCR Forecast in column C, and forward depending on the number of VG or VCR resources. The timestamps in the DASCUC VG data must follow the requirements of t_{DAC}, I_{DAC}, and H_{DAC}. An example of the PJM 5 BUS system is shown in Table 32.

Table 32: DASCUC VG FORECAST TIMESERIES Data

	A	B	C
1	DASCUC	INTERVAL	WINDY_HILL

2	1	0	111.6
3	1	1	112.8
4	1	2	110.6
5	1	3	112.4
6	1	4	109.4
…	…	…	…
24	1	22	43.1
25	1	23	52.1

In this example, the DASCUC is run once a day (t_{DAC}=24) with an interval length of 1 hour (I_{DAC}=1), and horizon of 24 intervals (H_{DAC}=24). For one DASCUC index, there should be H_{DAC} number of INTERVALs. Each INTERVAL should be I_{DAC} hours apart. Further explanations of the time intervals are given in Section 3.3.

DASCUC Reserve Data:

The DASCUC load data file(s) are required unless there are no reserve requirements. The DASCUC Reserve Level data includes the Reserve Levels that would be required for the DASCUC sub-model. The DASCUC Reserve Level data includes the DASCUC index (e.g., the number of the DASCUC run) in column A, the INTERVAL in units of hours in column B, and the DASCUC Reserve Levels in column C, and forward depending on the number of Reserve Types. The timestamps in the DASCUC Reserve data must follow the requirements of t_{DAC}, I_{DAC}, and H_{DAC}. An example of the PJM 5 BUS system is shown in Table 33.

Table 33: DASCUC Reserve Levels TIMESERIES Data

	A	B	C	D	E
1	DASCUC	INTERVAL	SPIN	NON_SPIN	REGULATION
2	1	0	30	60	25
3	1	1	30	60	25

4	1	2	30	60	25
5	1	3	30	60	25
6	1	4	30	60	25
…	…	…	…	…	…
24	1	22	30	60	25
25	1	23	30	60	25

In this example, the DASCUC is run once a day (t_{DAC}=24) with an interval length of 1 hour (I_{DAC}=1), and horizon of 24 intervals (H_{DAC}=24). For one DASCUC index, there should be H_{DAC} number of INTERVALs. Each INTERVAL should be I_{DAC} hours apart. Further explanations of the time intervals are given in Section 3.3.

4.3 FESTIV User Interface

Once all the necessary inputs are gathered and the input file has been created, FESTIV is ready to run. Before beginning the simulation, the input parameters need to be entered into the FESTIV GUI (Graphical User Interface). The following are the list of things that need to be entered. All the fields in the initial FESTIV window are necessary. Inputs to each parameter are explained in detail below (in sequential order).

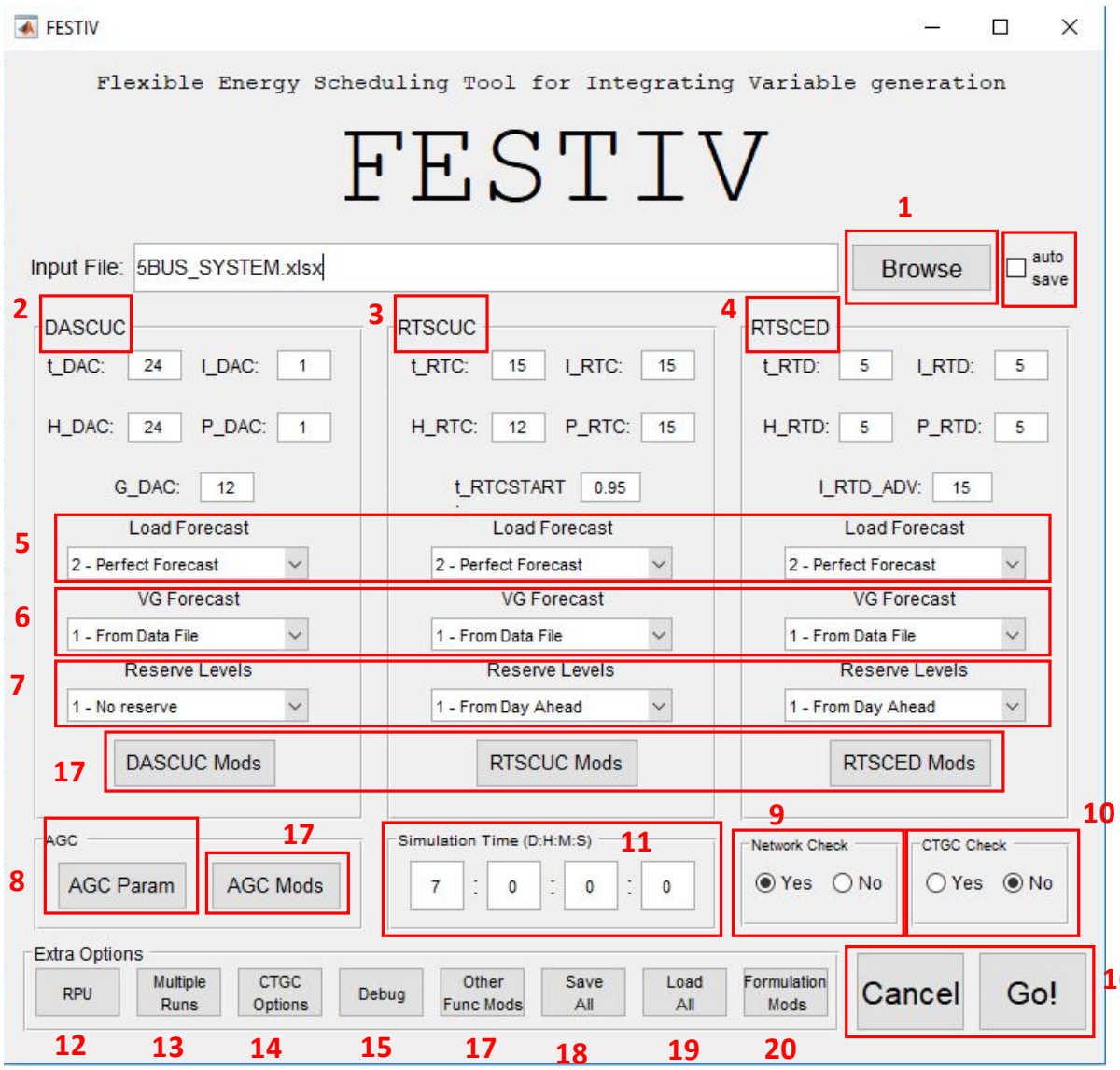


Figure 15: The FESTIV GUI

1) **Browse** : This button is used to access the input file built in the previous section. Click the browse button and the following window will appear. Select the appropriate input file from the list of files.

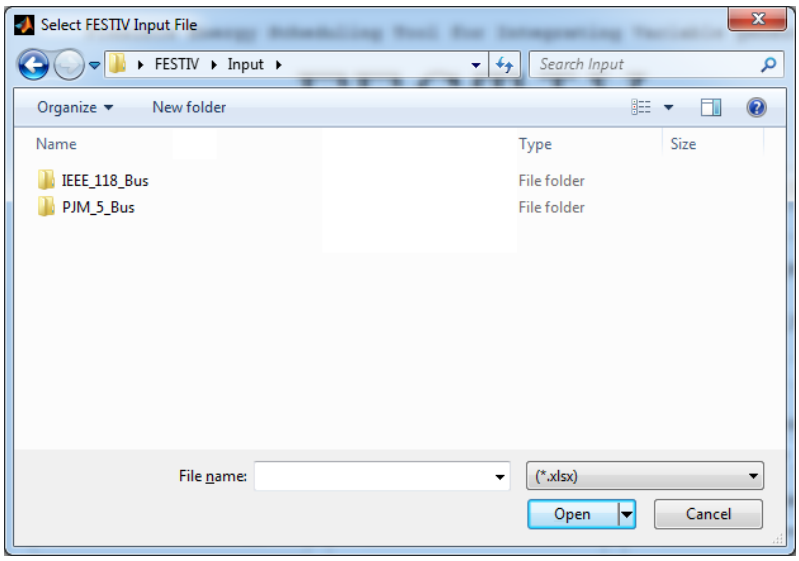


Figure 16: FESTIV Input file browser window

Tip: Try to keep any additional excel files that are not *Main Input Files*, out of the *Input Directory*. This will help all users know which files are inputs and which are not.

2) DASCUC : Day Ahead Security Constrained Unit Commitment Parameters

Table 34: Parameters used in Day Ahead Security Constrained Unit Commitment

Parameter Name	Parameter Description	Parameter Symbol	Format	Typical Values
Optimization Horizon	DASCUC Scheduling Horizon	H_{DAC}	Intervals (#)	1-2 days (24 if 1 I_{DAC}=1)
Optimization Interval Resolution	DASCUC Interval length	I_{DAC}	Hours	1
Update Horizon	Time between DASCUC updates	t_{DAC}	Hours	24 (i.e., once per day)
Time to Solve	Model processing time	P_{DAC}	Hours	

Parameter Name	Parameter Description	Parameter Symbol	Format	Typical Values
1st Market Gate	Market Gate for DASCUC. When the solution for the DASCUC first completes	G_{DAC}	Time of day (hours)	12 (noon)

Notes: For realistic results, I_{DAC} should be divisible by 60 minutes. To avoid error, the DASCUC must cover the intervals before RTSCUC gets to them.

3) RTSCUC : Real Time Security Constrained Unit Commitment Parameters

Table 35: Parameters used in Real Time Security Constrained Unit Commitment

Parameter Name	Parameter Details	Parameter Symbol	Format	Typical Values
Optimization Horizon	RTSCUC Scheduling Horizon	H_{RTC}	Intervals (#)	3 hours (12 if I_{RTC}=15)
Optimization Interval Resolution	RTSCUC Interval length	I_{RTC}	Minutes	15 Minutes
Update Horizon	Time between RTSCUC updates	t_{RTC}	Minutes	15 Minutes
Time to Solve	Model processing time	P_{RTC}	Minutes	15 Minutes

Notes: For realistic results I_{RTC} should be less than or equal to I_{DAC} and I_{DAC} should be divisible by I_{RTC}. Only simulations where t_{RTC}=I_{RTC} have been thoroughly tested in the current version.

4) RTSCED : Real Time Security Constrained Economic Dispatch Parameters

Table 36: Parameters used in Real Time Security Constrained Economic Dispatch

Parameter Name	Parameter Details	Parameter Symbol	Format	Typical Values
Optimization Horizon	Scheduling Horizon	H_{RTD}	Intervals (#)	5 Minutes – 1 Hour (value depends on I_{RTD})
Optimization Interval Resolution	Interval length	I_{RTD}	Minutes	5 Minutes
Advisory Interval	Interval length of advisory lookahead intervals	$I_{RTD-ADV}$	Minutes	5-15 Minutes
Update Horizon	Time between updates	t_{RTD}	Minutes	5 Minutes
Time to Solve	Model processing time	P_{RTD}	Minutes	5 Minutes

Notes: For realistic results I_{RTD} should be less than or equal to I_{RTC} and I_{RTC} should be divisible by I_{RTD}. Only simulations where t_{RTD}=I_{RTD} have been thoroughly tested. $I_{RTD-ADV}$ should be divisible by 60 and greater than or equal to I_{RTD}. Often, if H_{RTD} is greater than 1, $I_{RTD-ADV}$ can be set to I_{RTC} to ensure changes in commitment are considered in the RTSCED. The second RTSCED interval will land on the next time point that is divisible by $I_{RTD-ADV}$. The following RTSCED intervals will be $I_{RTD-ADV}$ minutes after the previous intervals.

5) Load Forecasts

For DASCUC, RTSCUC, and RTSCED, an option is available for the user to select how the Load Forecasts are calculated. This drop down box has four options and each option defines how the load forecast is calculated for each sub-model. The four options are listed below:

- 1-From Data File: This will read in the data that is given as a reference in the *Main Input File*, and lies in the TIMESERIES directory (See Section 4.1 Main Input File and Section 4.2 Time series Inputs).
- 2-Perfect Forecast: This will create Load Forecasts that are equal to the average *Actual Load*, based on averaging over the interval resolution, *I.*
- 3-Persistance Forecast: This will create Load Forecasts that are equal to the load before the sub-model would start (i.e., time – *P*). Note that this method will not work for DASCUC Load Forecasts, as persistence forecasts would not be plausible.
- 4-Predefined with NDE: This will create Load Forecasts that have a predefined normally distributed error randomly added to the A*verage Load* for each interval in the sub-model. Values for the error must be entered manually either through a FESTIV Mod placed prior to Forecast Creation or in the FESTIV_ADDL_OPTIONS script. There must be the same number of errors as the horizon count for each sub-model.

6) Variable Generation Forecasts

For DASCUC, RTSCUC, and RTSCED, an option is available for the user to select how the VG Forecasts are calculated. This drop down box has four options and each option defines how the VG Forecast is calculated for each sub-model. The four options are listed below:

- 1-From Data File: This will read in the data that is given as a reference in the *Main Input File*, and lies in the TIMESERIES directory (See Section 4.1 Main Input File).
- 2-Perfect Forecast: This will create VG Forecasts that are equal to the average *Actual VG*, based on averaging over the interval resolution, *I.* Note that if any VCR exists, an input file would still be needed.
- 3-Persistance Forecast: This will create VG Forecasts that are equal to the VG output before the sub-model would start (i.e., time – *P*). Note that this method will not work for DASCUC VG Forecasts, as persistence forecasts would not be plausible. Note that if any VCR exists, an input file would still be needed.
- 4-Predefined with NDE: This will create VG Forecasts that have a predefined normally distributed error randomly added to the A*verage VG* for each interval in the sub-model. Note that if any VCR exists, an input file would still be needed. Values for the error must be entered manually either through a FESTIV Mod placed prior to Forecast Creation or in the FESTIV_ADDL_OPTIONS script. There must be the same number of errors as the horizon count for each sub-model.

7) Reserve Levels

For RTSCUC and RTSCED, an option is available for the user to select how the Reserve Level requirements are determined.. By choosing option "1-From Day Ahead", the values are taken from Day Ahead Reserve Level requirements. By choosing option "2-From Data File", the values are taken from a separate input file, which would be referenced from the *Main Input File* and would lie in the TIMESERIES directory.

- 1-No reserve: This will set the reserve requirements at 0MW for all services (Day-ahead only. Use "From Day-Ahead" for RTSCUC and RTSCED along with this for DASCUC to set requirements to 0 MW for all services and all sub-models)
- 1-From Day Ahead: This will set the reserve requirements in RTSCUC and RTSCED at the day-ahead level for the corresponding time interval (RTSCUC and RTSCED only).
- 2-From Data File: This will read in the data that is given as a reference in the *Main Input File*, and lies in the TIMESERIES directory (See Section 4.1 Main Input File).

8) AGC: Automatic Generation Control Parameters

The AGC section has an *AGC Param* button. This provides more options for the user to configure how the ACG is modeled in FESTIV. Options include CPS2 Options, Smoothed ACE Options, AGC Deadband, and AGC modes.

The CPS2 options provide the user with options to change the CPS2 interval length (in minutes) and the L10 (ACE Limit in MW) values. CPS2 is a reliability standard that is based on the NERC Reliability Standards.[5] Predefined L10 values based on the U.S. NERC Balancing Areas are preloaded into the model for reference. The "Other L10 Values" button will provide the L10 values for some common operating regions. If individual electric utility values are needed, click the complete list button on the L10 Values window and a NERC page with all the published values will be displayed. Select the appropriate values and then enter them in the CPS2 Options section. The NERC CPS2 interval length is 10 minutes, but this can be changed if a different metric is sought to be evaluated.

[5] North American Electric Reliability Corporation, http://www.nerc.com/files/Reliability_Standards_Complete_Set.pdf

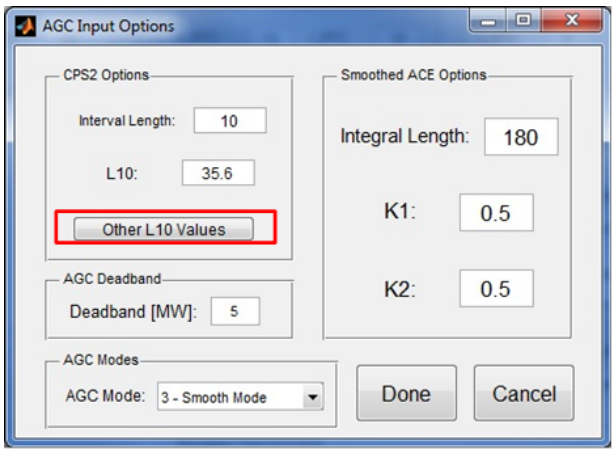


Figure 17: AGC input options dialog box

FESTIV has several different options regarding how generators perform automatic generation control. Mode 1 is the 'Blind Mode.' In this mode, generators ignore the ACE signal and do not participate in AGC. Mode 2 is the 'Fast Mode.' In this mode, generators follow the unfiltered, high frequency ACE signal. This mode essentially tries to correct the real time, raw imbalance. Mode 3 is the 'Smooth Mode.' In this mode, generators follow a filtered, low frequency ACE signal. This low frequency signal is produced by passing the raw ACE signal through a PI filter whose parameters are defined in the right hand side of the 'AGC Input Options' dialog box shown in Figure 17. Mode 4 is the 'Lazy Mode.' In this mode, regulation only occurs if there is an expected CPS2 violation. If there is an expected violation, the generators will participate in mode 3 regulation; otherwise they participate in mode 1 regulation. Mode 5 is the 'Individual Mode.' In this mode, each generator receives its own AGC signal as defined by the 'GEN_AGC_MODE' column on the GEN input sheet. Note that the update frequency of the AGC model, t_{AGC}, is determined by the temporal resolution of the realized load timeseries.

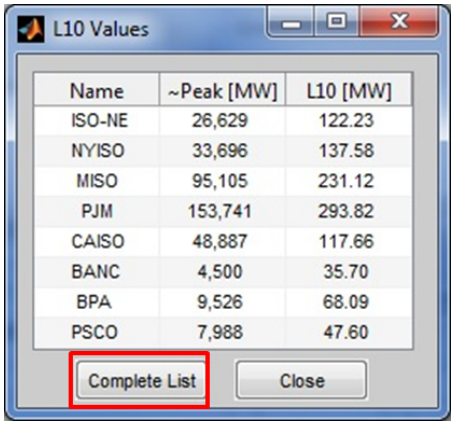


Figure 18: L10 values dialog box

9) Network Check

This radio button is used to select if the model will incorporate the normal network constraints in the DASCUC, RTSCUC, and RTSCED sub-models. By checking *Yes*, Network Check is activated. By checking no, the system is modeled as a single bus without transmission flow constraints.

10) CTGC Check

This radio button is used to select if the model will incorporate network contingency constraints. By checking *Yes*, the Contingency Check is activated and DASCUC, RTSCUC, and RTSCED will ensure monitored network contingencies are enforced. By checking no, these models will not enforce any contingency constraints. Note that Network Check must be checked in order for Contingency Check to be enabled.

11) Simulation Time

The user needs to enter the simulation time frame in this section. The syntax for this is DD:HH:MM:SS. By selecting 01:00:00:00, the simulation will be run as a one-day simulation. Selecting 07:00:00:00 will run a one-week simulation.

12) RPU: Reserve Pick Up

This button pop-ups a new window, *Reserve Pick Up Parameter Input Dialog.* Using this option, the user can configure if and how the RPU model would be used during the simulation. The RPU, also called real-time contingency dispatch (RTCD) in practice, is a RTSCUC model that runs based on an event being triggered, rather than based on time. The basis of the event trigger can be configured by the user.

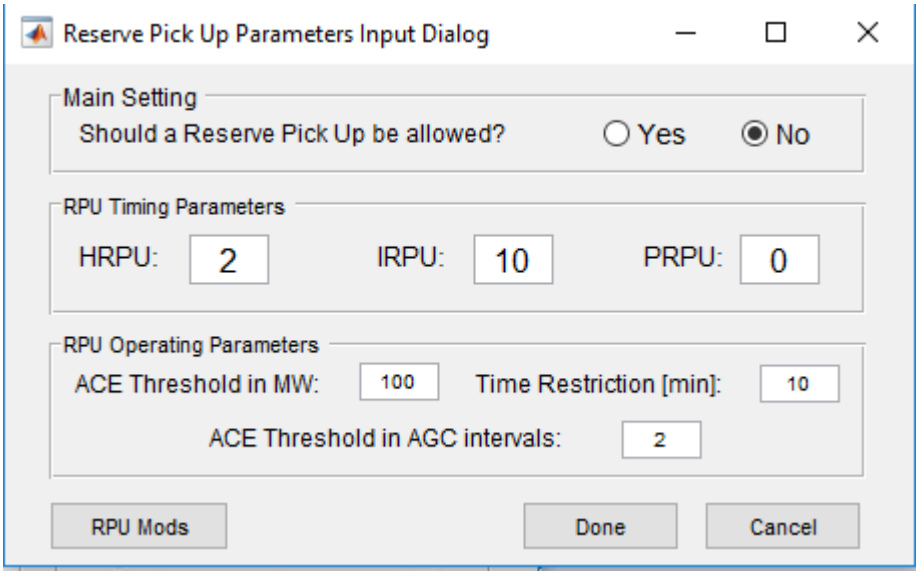


Figure 19: Reserve Pick Up Parameter Input Dialog Window

The first Section will provide the user with option if the RPU should be used or not. By default this is *NO*. The section contains the information about RPU timing parameters.

Table 37: Parameters used in the Reserve Pick Up

Parameter Name	Parameter Details	Parameter Symbol	Format	Typical Values
Optimization Horizon	Scheduling Horizon	H_{RPU}	Intervals (#)	20 minutes
Optimization Interval	Interval length	I_{RPU}	Minutes	10
Time to Solve	Model processing time	P_{RPU}	Minutes	0

ACE Threshold	The threshold of ACE where if it exceeds an operator would run an RPU	-	MW	
Time Restriction	The time requirement before which a subsequent RPU is run to avoid overuse	-	Minutes	10
Threshold in AGC intervals	The number of AGC intervals that the ACE Threshold is required to be exceeded before RPU is run.	-	# intervals	

The ACE Threshold is the required threshold where if the absolute value of ACE exceeds then RPU would be initiated. RPU is also run if a generator contingency occurs. In order to run RPU only for contingencies, the ACE Threshold should be set very high. Time Restriction is used so to avoid over running of RPU. It is common to set this at least greater than I_{RPU} so that the RPU would only run again if the directions from the previous RPU have been completed. Finally, the Threshold in AGC Intervals is the amount of AGC Intervals that the ACE Threshold must be exceeded for (length of time where ACE is significant) before the RPU is run. This can avoid running RPU due to ACE noise, or ACE for extremely short duration.

13) Multiple Runs

This button provides the user with an option to run multiple scenarios back-to-back without the need to re-enter all the information for each new run. This option is helpful for running multiple simulations overnight. Once this button is clicked, *Multiple Runs Input Dialog* window is opened. In order to activate Multiple Runs:

a. Click the check box on the top-left hand side of the window.
b. Use the *Browse* button to select a new input file for a different scenario and then enter the output file path in the *Output File Name* section.
c. Click the add button to add this to FESTIV run.
d. Repeat steps b and c to add more runs.
e. Click *Done* to finish adding the scenarios.

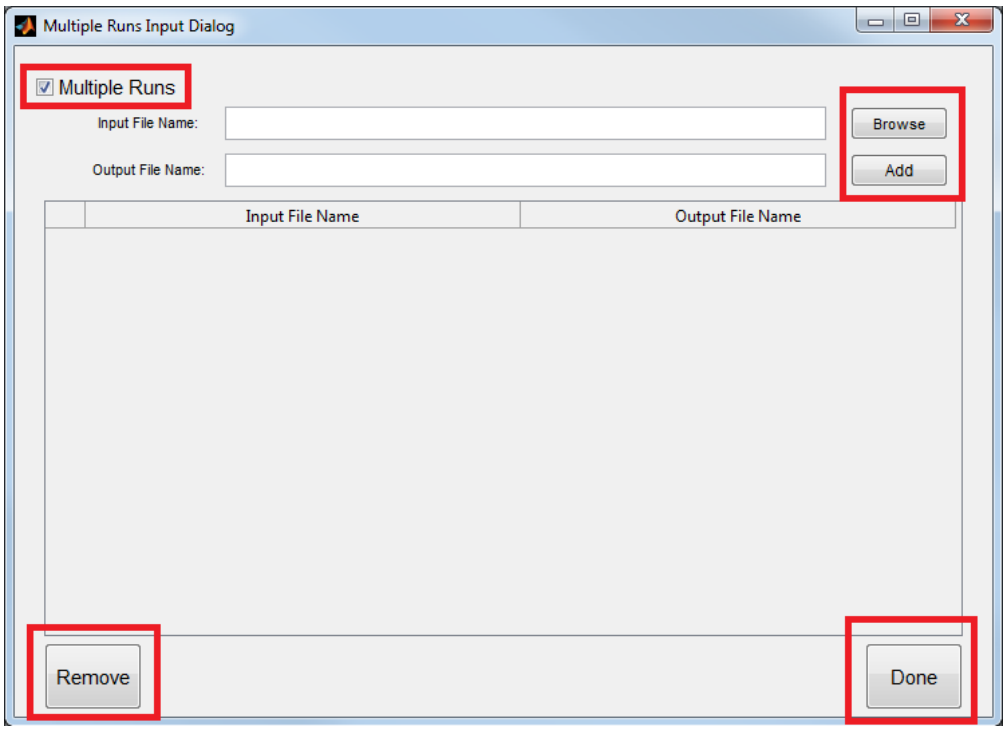


Figure 20: Multiple Runs Input Dialog Window

Note that multiple simulations can be run where other parameters within the FESTIV GUI can be applied as well. If this is desired, add the first case with the desired parameters and input and output file names, click done, then change the parameters for the next simulation case, and click *Multiple Runs* and repeat until the desired number of simulations are entered. If a case is added to the list by mistake, it can be removed by highlighting it and pressing *Remove*.

14) Contingencies

This option is used to select if the model should simulate generator contingencies. By clicking this button, a new window pops up, *Contingency Parameters Input Dialog.*

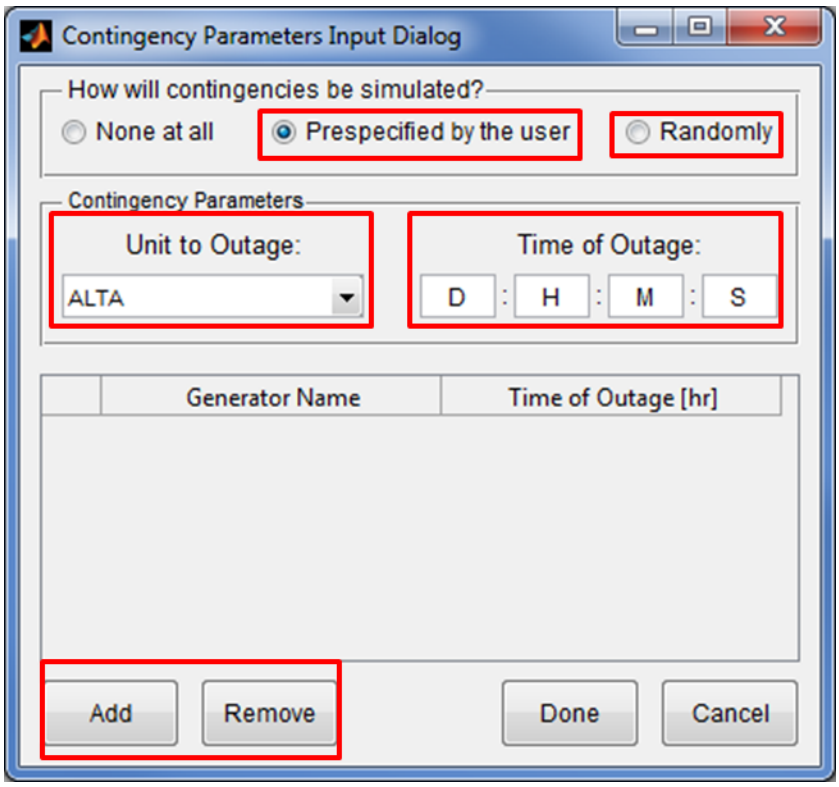


Figure 21: Contingency Parameters Input Dialog Window

This window contains various options that the user can select on how to apply the generator contingency. By selecting *Randomly* button in the top section, the contingencies occur randomly. If the user needs to enter the contingencies for a particular generator, then select *Prespecified* by the user option. This will activate the *Contingency Parameter* sections. Select the unit from the dropdown menu, *Unit to Outage*, and select the time of the outage by inputting the values in *Time of Outage* section. Once the required contingency is selected, click *Add*. All the required contingencies should be added in the same manner. Once all the required contingencies are added, click *Done*.

15) Debug

This button is used to change the debugging parameters to identify any problems that may be encountered during program execution or to stop the simulation during the run to analyze a particular time frame. Clicking this button will pop-up the *Debugging Parameters Input Dialog* window. The user can suppress output plots if figures following the simulation are not desired (user can always create these afterward if suppressed). The user can also disallow integers to be used for the unit commitment models in FESTIV. Finally, the user can require FESTIV to stop in the middle

of execution at a specific time frame to debug or analyze the current conditions for that timeframe. Users can also use STOP_FESTIV script to do this for reasons other than timeframes (see Section 7.1)

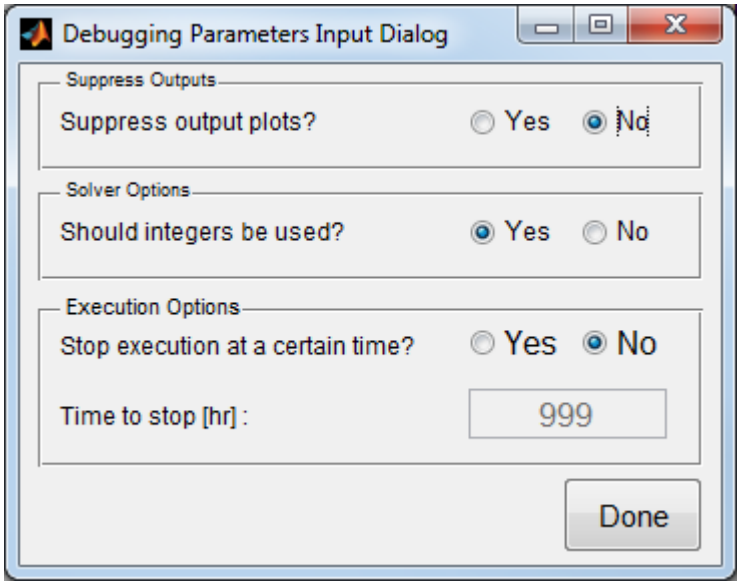


Figure 22: Debugging Parameters Input Dialog window

16) GO! And Cancel

Once all the necessary parameters are entered, use the *Go!* Button to run the simulation. If the user does not want to start the simulation, the *Cancel* button can be used to exit FESTIV.

17) Functional Mods Buttons

FESTIV has the ability to simulate specific sub-model modifications (Functional Mods) for specific applications. Clicking one of these buttons will open a dialog box similar to that shown in Figure 23.

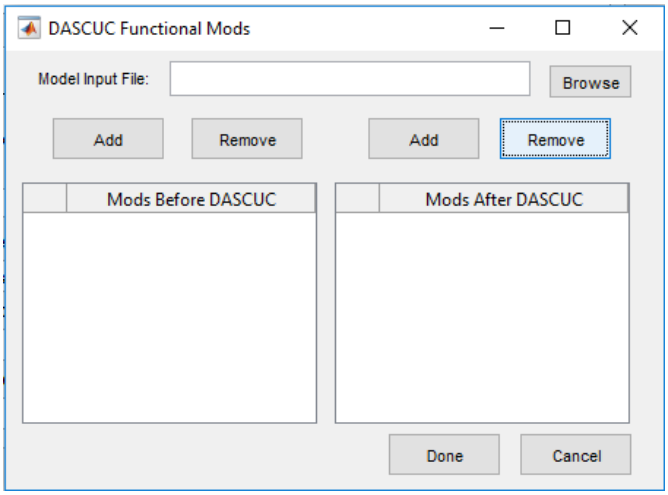


Figure 23: Sample sub-model rules dialog box

Clicking the 'Browse' button will open a windows file browser in the 'MODEL_RULES' directory in the FESTIV directory. After selecting a specific rule file, the rule can then be added either before or after the appropriate sub-model. In this example, a rule may be added either before or after the DASCUC sub-model. More information on Model Rules can be found in Section 6. FESTIV.

18) Save All

Save the current selection of functional modifications being used and other data from the FESTIV GUI (timing parameters for DASCUC, RTSCUC, and RTSCED, load Forecast/VG Forecast/Reserve levels types, Network Check, CTGC Check, AGC parameters, Simulation time, and RPU parameters.

19) Load All

Load a saved configuration of functional modifications and other data from the FESTIV GUI (timing parameters for DASCUC, RTSCUC, and RTSCED, load Forecast/VG Forecast/Reserve levels types, Network Check, CTGC Check, AGC parameters, Simulation time, and RPU parameters.

20) Formulation Modification Options

This button opens the *Modify Optimization Formulation* dialog box. This dialog box is used to modify the formulation of the optimizations being performed by FESTIV and allow users to add or delete Formulation Modifications to the formulation of different optimization-based sub-models. The user can load, save, and produce a summary for the full set of functional mods for all optimization sub-models. The user can also clear or move to the default formulation for an individual sub-model. Individual formulation mods can be added or deleted to the appropriate section and appropriate sub-model using the browse button and middle arrows. More details on how to use this feature are given in Section 6. FESTIV.

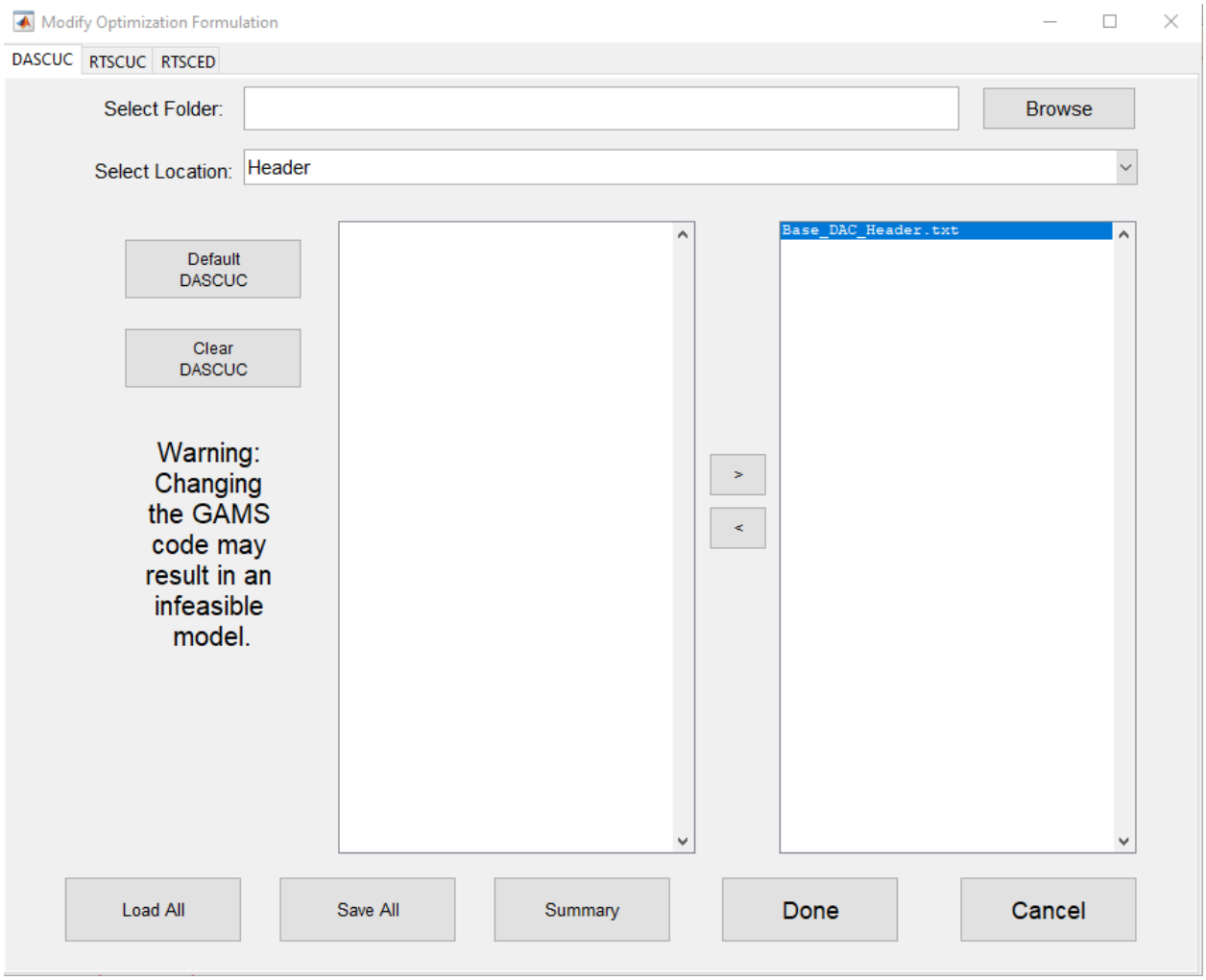


Figure 24: GAMS options dialog box

21) Autosave checkbox

Checking this box will create a dialog box prompting the user to input a file name to save the current FESTIV run at the end automatically without launching the user normal user prompt at the end of the run.

FESTIV GUI Example

In the aforementioned sub-section, details about the GUI parameters are described. This section extends the PJM 5 Bus system example case to illustrate what / how the FESTIV GUI parameters are entered to run the PJM 5 Bus system example case.

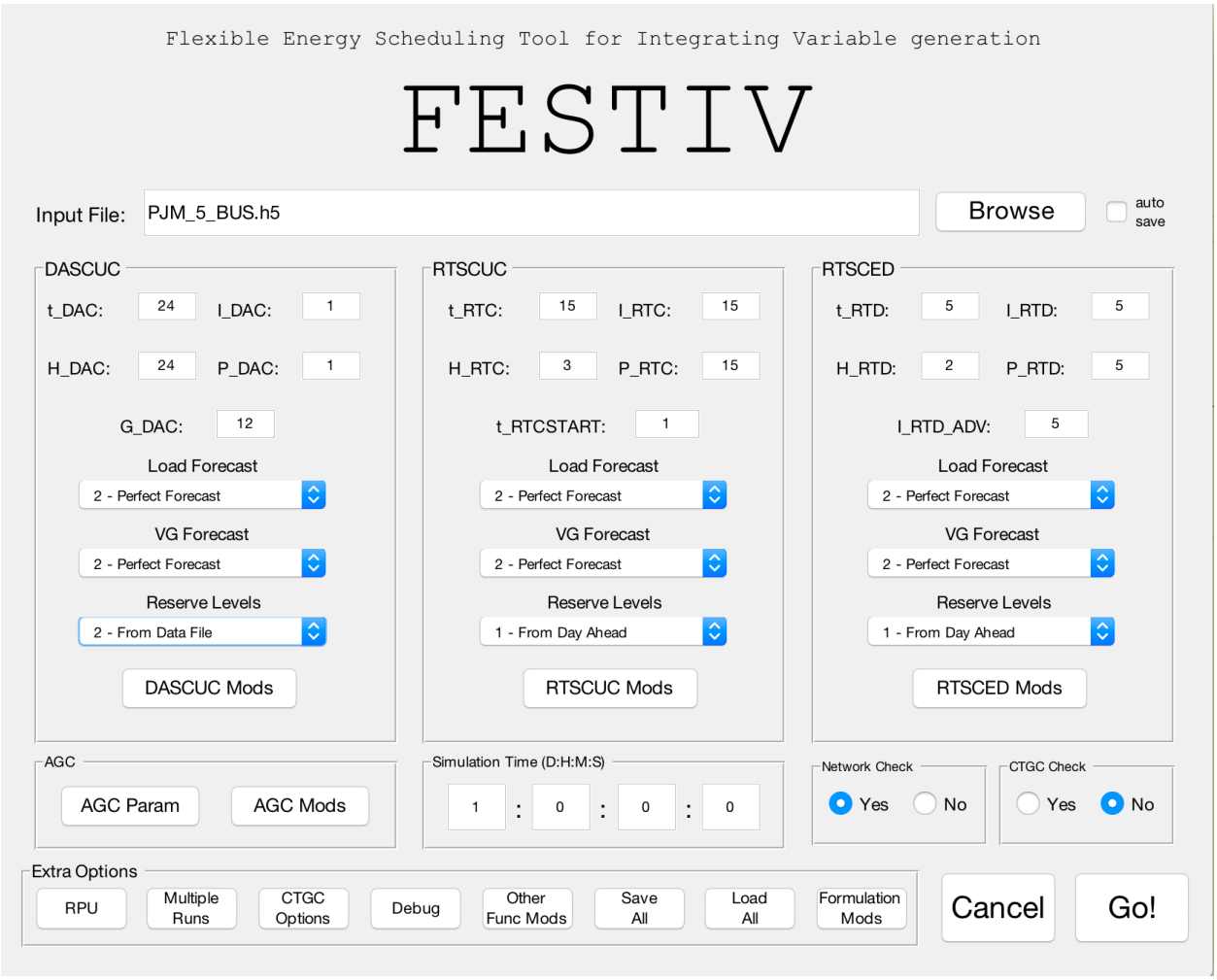


Figure 25: PJM 5 Bus System FESTIV GUI Example

As the figure above illustrates, the Network Check is selected while the CTCG Check option is not. This system is simulated for one day only as can be seen in the Simulation Time section. *Perfect Forecast* and *Persistence Forecast* are selected for Load Forecast and VG Forecast respectively (for both RTSCUC and RTSCED). For Reserve forecast, *From Day Ahead* option is selected for both RTSCUC and RTSCED. The day-ahead load and vg forecasts are provided by separate timeseries. The following parameters are entered in the window shown above:

Table 38: FESTIV GUI Input Parameters for PJM 5 BUS System

Parameter Name	Parameter Details	Parameter Symbol	Values Entered
DASCUC Parameters			
Optimization Horizon	Scheduling Horizon	H_{DAC}	24 hours
Optimization Interval	Interval length	I_{DAC}	1 hour
Update Horizon	Time between updates	t_{DAC}	24 hours
Time to Solve	Model processing time	P_{DAC}	1
1st Market Gate	Hours before solution for actual day	G_{DAC}	12
RTSCUC Parameters			
Optimization Horizon	Scheduling Horizon	H_{RTC}	3
Optimization Interval	Interval length	I_{RTC}	15 (minutes)
Update Horizon	Time between updates	t_{RTC}	15 (minutes)
Time to Solve	Model processing time	P_{RTC}	15 (minutes)
RTSCED Parameters			
Optimization Horizon	Scheduling Horizon	H_{RTD}	2
Optimization Interval	Interval length	I_{RTD}	5 (minutes)
Advisory Interval		$I_{RTD-ADV}$	5
Update Horizon	Time between updates	t_{RTD}	5 (minutes)
Time to Solve	Model processing time	P_{RTD}	5 (minutes)

Using the parameters mentioned above, the PJM 5 Bus System model can be simulated. See the output section for sample outputs produced by FESTIV regarding this example.

Outputs

During the FESTIV simulation, the MATLAB Command Window would show the progress of the current simulation. Information such as Reading the input file, modelling the DASCUC, RTCUC, Study Period clock, and end of simulation are shown. Once the simulation is completed, the *Command Window* displays the output summary and generates seven graphs. The user will be prompted with an option to save the workspace, output, and figures. When prompted, the user needs to enter the output file name to save the results or hit enter to continue without saving. The results are stored in a sub-directory with the user specified name within the FESTIV\OUTPUT directory. The following figure shows the details from the aforementioned discussion:

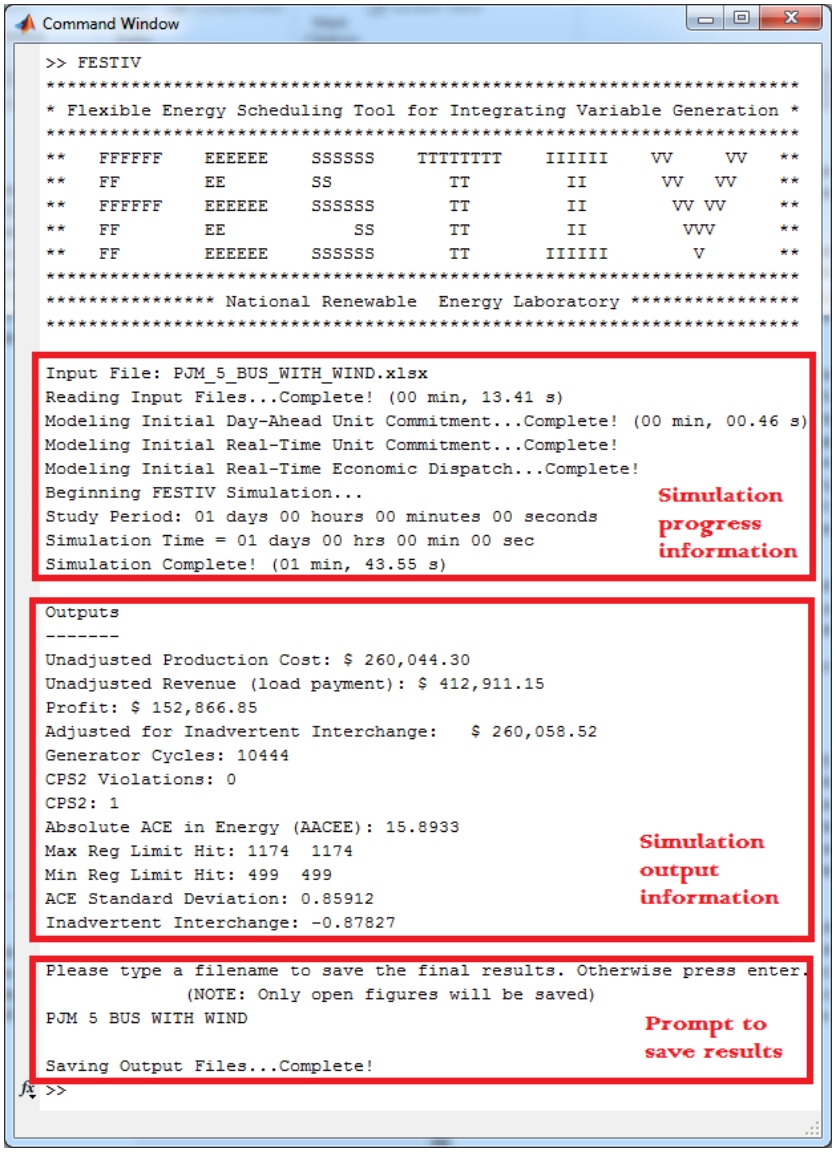


Figure 26: FESTIV Simulation Command Window

The MATLAB Workspace, the summary spreadsheet, model details spreadsheet, and the seven graphs will be stored in the sub-directory if the user selects to save the results. The summary spreadsheet contains case summary and the high level case results. The model detail spreadsheet contains the summary information as well as detailed information regarding the ACE, output, schedules, reserves, and prices for the run

5.1 FESTIV Metrics

The amount of data that output from a normal FESTIV simulation run is quite large. However, there are a few key metrics that are presented and are of most importance for comparing different simulations with each other.

Unadjusted Production Cost: The Unadjusted Production Cost is the total system production costs as directly computed at the end of the simulation. The Unadjusted Production Cost calculates the cost of each generator to supply energy at every t_{AGC} interval for the entire study period. This includes start-up, no-load, and incremental energy cost as well as any bid in ancillary service costs. These costs can also be shown by unit and by time period.

Adjusted Production Cost: The Adjusted Production Cost uses the Unadjusted Production Cost with some post-processing. Since a primary FESTIV application is the testing of the exact same system input data with differing timing parameters, scheduling strategies, market designs, or new market entries, the production costs comparisons must be “apples-to-apples”. Since some cases will have more ACE than others, they may not be providing the same amount of energy and thus making the cost comparisons unfair. The Adjusted Production Cost will pay (or sell) any negative (positive) inadvertent interchange it has at the end of the study period at a high (or low) price. The Adjusted Production Cost will also make adjustments when energy storage resources are present to make sure accurate comparisons can be made.

Revenue: Revenue is the total revenue of all resources on the system after getting paid LMP for energy, and any ancillary service price for ancillary service provisions. The revenue can also be seen by unit, time period, by day-ahead or real-time, and by energy or ancillary services.

Profit: Profit is the total revenue minus the total cost of all resources on the system. Profit can also be seen by unit and by time period.

Area Control Error: Area Control Error (ACE) is the difference between the sum of total generation and the total load at any given time period. Although, ACE in balancing areas that are part of large synchronous interconnections is a bit more complex, FESTIV, modeling a single balancing area without electrical frequency, will model ACE as simply the total generation minus the total load. This is similar to ACE in practical systems if frequency is at nominal level, or if governor responsive units are providing frequency

response that is equal to the frequency bias of the ACE equation. ACE is the main driver of all imbalance metrics.

CPS2: Control Performance Standard 2 is a NERC reliability standard that measures the amount of intervals where the absolute value of ACE exceeds a predefined threshold. An interval that exceeds the ACE Limit (known as L10) in a CPS2 interval (10 minutes as defined by NERC) will cause one violation. The percent of intervals without violations is the CPS2 score. This gives an indication of how often the system encounters severe imbalance errors.

Absolute Area Control Error in Energy (AACEE): AACEE is the absolute value of ACE at every t_{AGC} interval, summed up for the study period in units of MWh. This gives an indication of the total imbalance occurring for the study period in either direction.

Standard Deviation of ACE: The standard deviation of ACE gives an indication of the distribution of ACE for the study period.

Generator Cycles: The times where a generator had to change direction from increasing power output to decreasing, or decreasing to increasing.

Solve Time: How long the simulation took to solve.

5.2 Output Figures

As mentioned earlier, seven graphs will be generated as an output of the FESTIV simulation. The details of the seven graphs and their respective plots are shown below. All of the graphs are based on the PJM 5 Bus System simulation. All of the information contained in the plots can be printed to an excel spreadsheet including additional information such as individual reserve schedules.

1) ***Ace Levels*** *:* contains a plot of raw ACE, Continuous Integrated ACE (i.e., inadvertent interchange), and CPS2 ACE for the period of simulation.
2) ***Actual Generation***: Contains the plot of actual generation outputs for each generator for the period of simulation.
3) ***Day Ahead Prices****:* Contains a plot of day-ahead locational marginal prices for each bus for the period of simulation.
4) ***Real Time Prices***: Contains a plot of real-time locational marginal prices for each bus for the period of simulation.
5) ***RTSCED Schedules****:* Contains a plot of RTSCED schedule for all the generators in the system for the period of simulation.
6) ***Day Ahead Schedules***: Contains a plot of day-ahead schedules for all the generators in the system for the period of simulation.
7) ***Generation and Load****:* Contains a plot of load on top of total generation for the period of simulation.

The MATLAB Figure window has several tools that are useful in understanding the results of the simulation visually. If the graph contains multiple plots, for instance, Actual Generation Graph or Day-Ahead Schedule Graph, the unnecessary plots within that graph can be removed for much better visual inspection of the results. More information regarding how to use the MATLAB Figure Toolbar can be found here: http://www.mathworks.com/help/matlab/creating_plots/figures-plots-and-graphs.html

The seven figures are shown below for the 5-bus system for a 1-day simulation with 6-second wind and load data, hourly DASCUC, 15-minutely RTSCUC, and five-minute RTSCED.

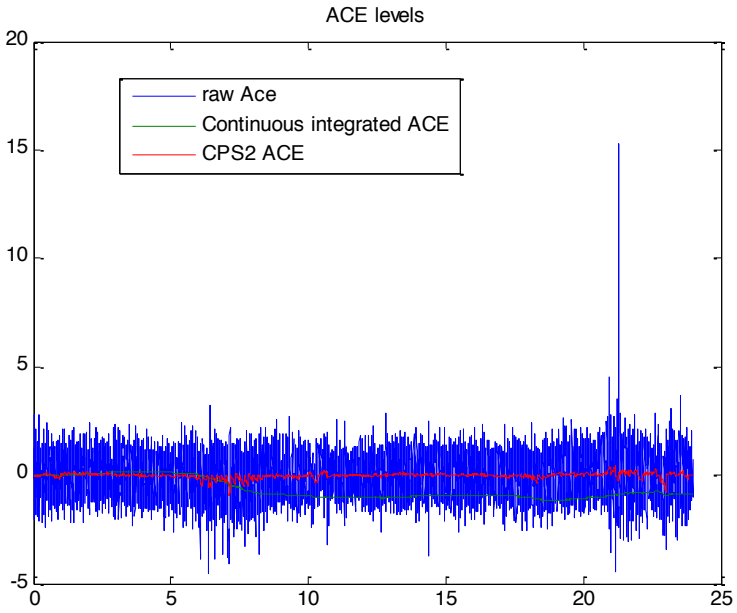


Figure 27: ACE Levels

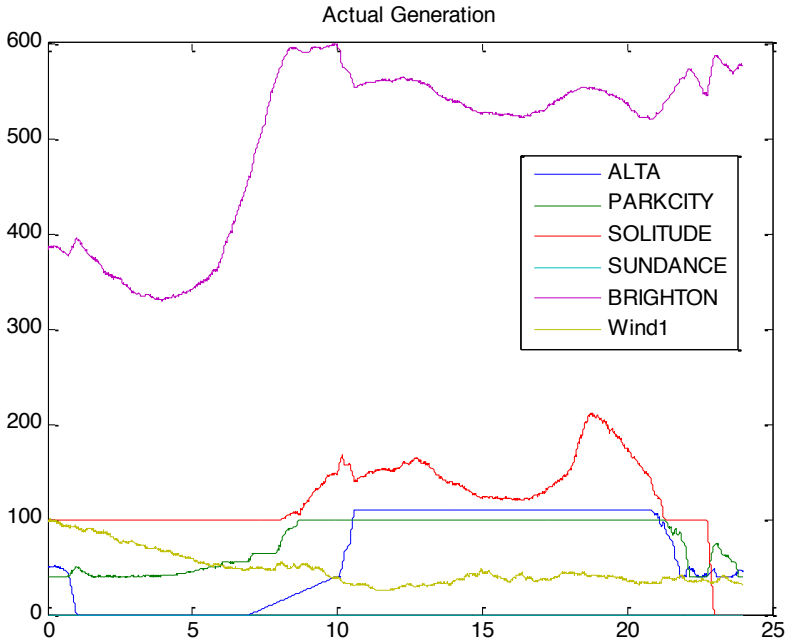


Figure 28: Actual Generation

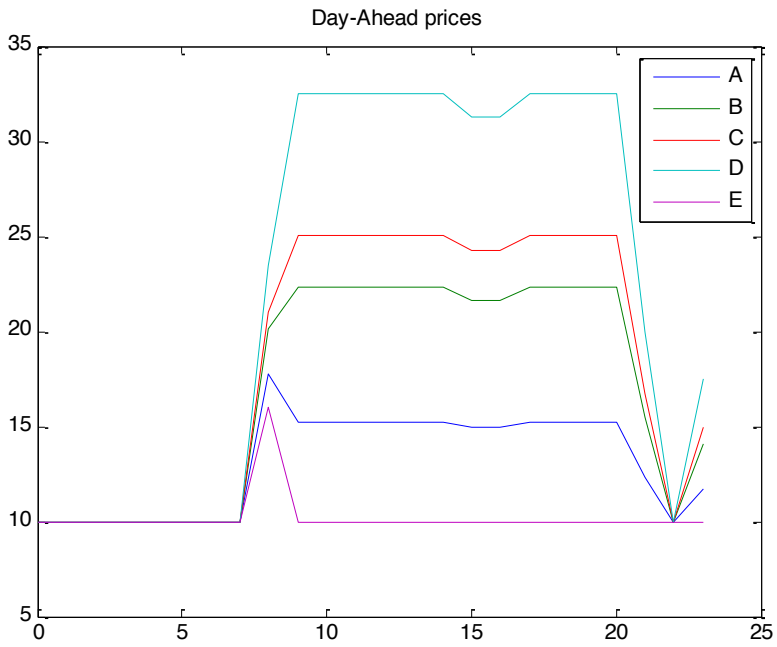


Figure 29: Day-Ahead Locational Marginal Prices

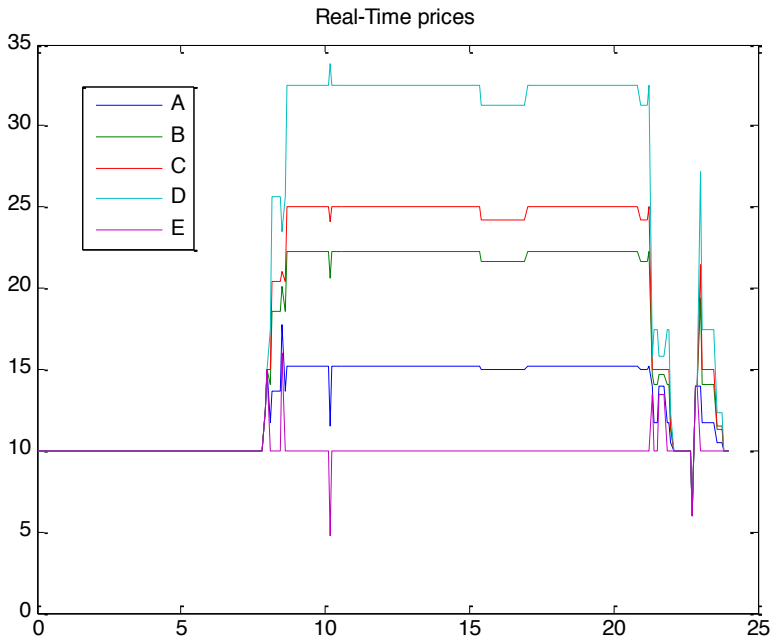


Figure 30: Real-Time Locational Marginal Prices

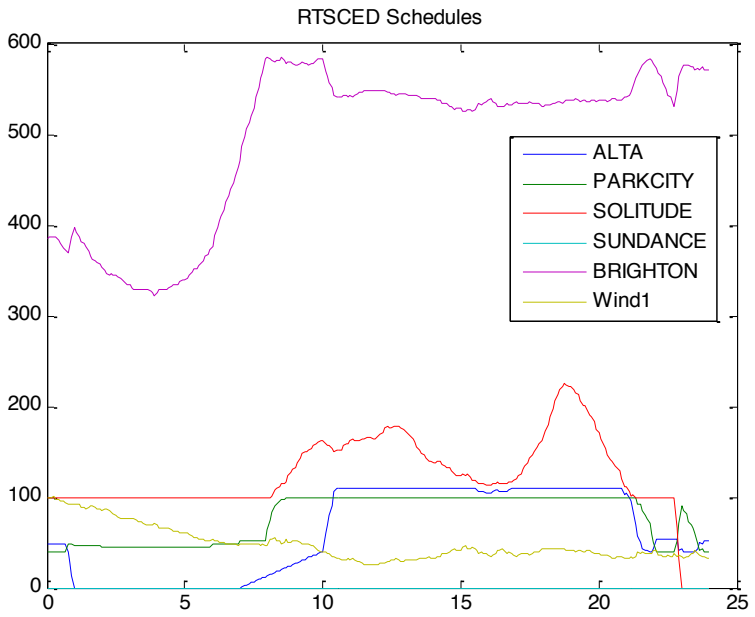


Figure 31: RTSCED Schedules

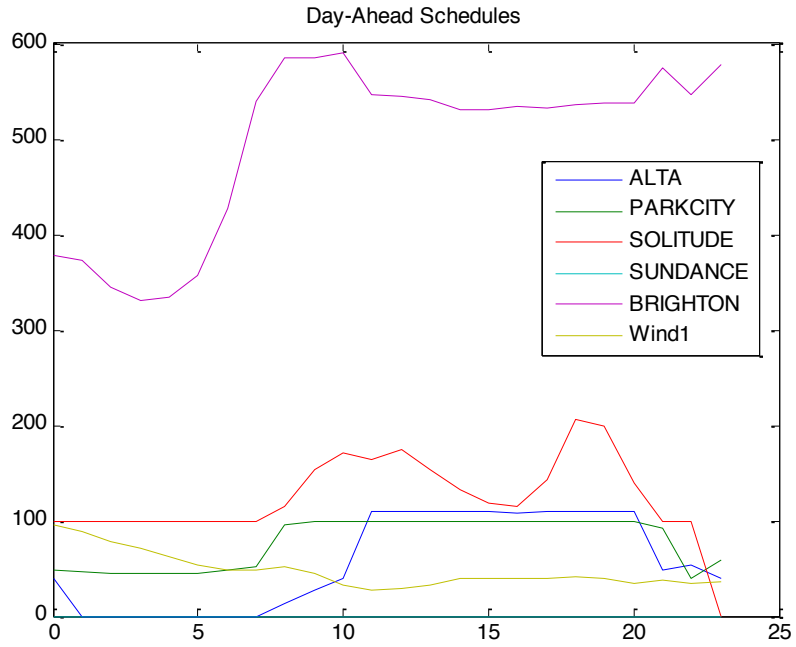


Figure 32: Day-Ahead Schedule

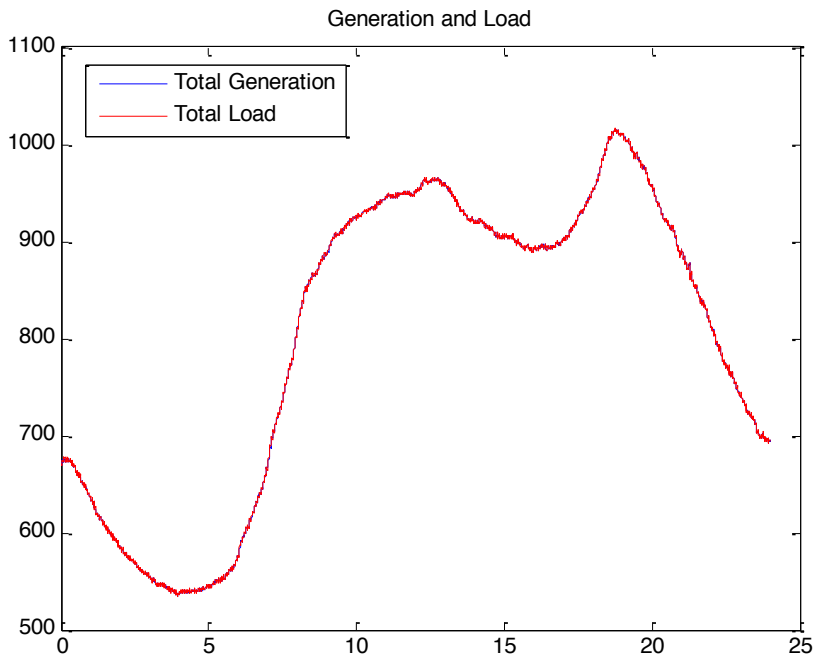


Figure 33: Total Generation versus Load

5.3 Main Output File

If an output name is provided at the end of a simulation, FESTIV will create a directory inside the 'FESTIV\OUTPUT' directory in which to save all appropriate output information. Along with the MATLAB workspace and the figures, FESTIV will also create 2 spreadsheets. These spreadsheets will be named with the provided name and the words 'Summary' or 'Details' appended to it. The 'Summary' spreadsheet includes all of the optimization parameters along with the outputs printed to the command window at the end of a simulation. The 'Details' spreadsheet includes detailed information regarding the simulation. This spreadsheet includes the following worksheets:

1. case_summary
2. results_summary
3. ACE
4. realized_generation
5. AGC_schedules
6. dascuc_schedules
7. rtscuc_schedules
8. rtsced_schedules
9. dascuc_lmps
10. rtsced_lmps
11. rtsced_reserveN

The first 2 worksheets are the same worksheets found in the 'Summary' spreadsheet. The 'ACE' worksheet includes the ACE timeseries calculated by FESTIV during that simulation. The first column on this worksheet is the time stamp. The second column is the raw ACE at each time interval. The third column is the integrated ACE up to that point in time. The fourth column is the CPS2 averaged ACE. The fifth column is the filtered, low frequency ACE signal that is used with AGC mode 3 (i.e. smooth mode). The sixth column is the current absolute ACE in energy.

The 'realized_generation' worksheet includes the realized output of all generators at the AGC temporal resolution. The 'AGC_schedules' worksheet includes the schedules given to all generators after the AGC is deployed. The 'dascuc_schedules' worksheet includes the day-ahead schedules of all generators. The 'rtscuc_schedules' worksheet includes the real-time security constrained unit commitment schedules. The 'rtsced_schedules' worksheet includes the real time dispatch of all generators.

The 'dascuc_lmps' worksheet includes the day ahead locational marginal prices at each bus. The 'rtsced_lmps' worksheet includes the real-time locational marginal prices at each bus.

The ‘rtsced_reserveN’ worksheets will include the real time reserve schedules for each generator where ‘N’ is the number of the reserve product being simulated. There will be ‘N’ number of worksheets corresponding to each reserve product. For example, if the simulation includes 3 reserve products, then there will be 3 additional worksheets in the ‘Details’ output spreadsheet labeled ‘rtsced_reserve1,’ ‘rtsced_reserve2,’ and ‘rtsced_reserve3.’

5.3 Workspace

In addition to the Main Output File, the entire workspace is saved as a .mat data file. This contains all the data that is part of each FESTIV Simulation Loop at the time of simulation completion. Note that when running multiple simulations, each simulation will have its own separate workspace. These files tend to be large, but are useful for finding information not part of the Main Output File and are required in order to run the FESTIV helper.

5.4 FESTIV Helper

The FESTIV helper is a separate tool that can be used to quickly alternate between analyzing previously completed and saved cases and create additional plots not included at the end of a simulation. An image of the FESTIV helper is shown below.

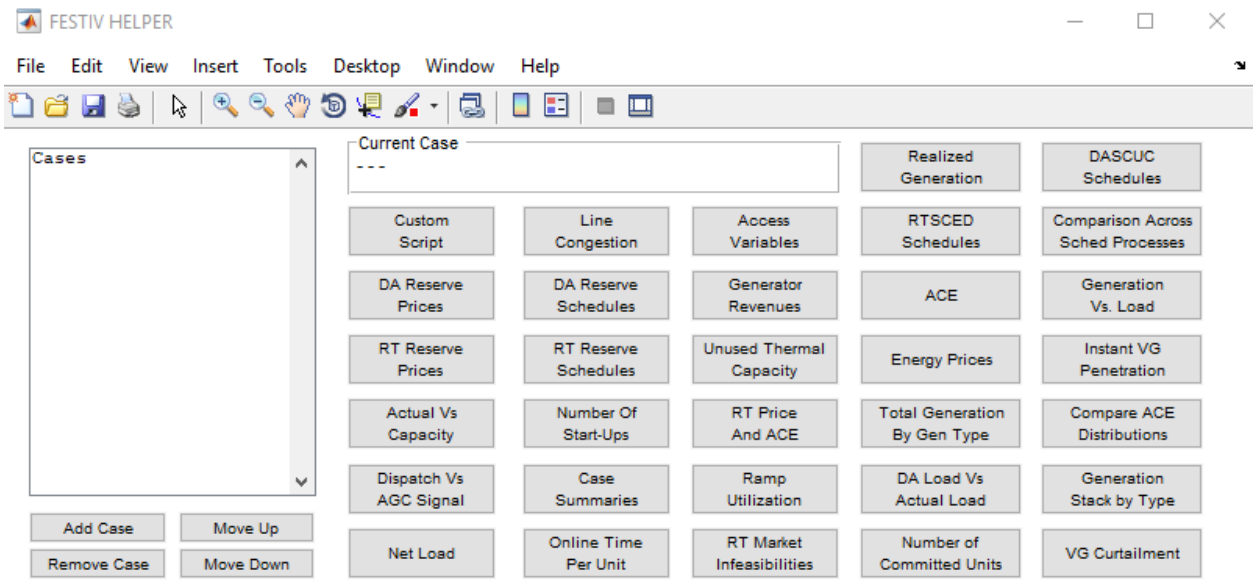


Figure 34: The FESTIV helper tool

In order to add a case to the helper tool, click ‘Add Case’ and select the directory where the output workspace(s) is saved. The helper will automatically load a list of all workspaces in that directory and all subdirectories. If a case is added by mistake, it can be removed by clicking on the case name and pressing the ‘Remove Case’ button. In order to load the case, double click the case name in list of names on the left. The case name should load automatically in the *Current Case* panel in the top-center. The cases can be rearranged by using the ‘Move Up*’* and ‘Move Down*’* buttons. This can be useful to control the way the cases are read by the helper and displayed in the plots. Then by clicking one of the buttons on the right, additional plots can be generated. A brief description of each button in the FESTIV helper is given below.

1) Realized Generation: A plot of the actual generation schedules of each generator
2) DASCUC Schedules: A plot of the day-ahead generation schedules of each generator
3) Custom Script: This will allow a user to choose a previously saved script which creates a plot for the case(s) selected (no more than two cases can be selected) These scripts are also called Helper Mods.
4) Line Congestion: A heat map of the congestion in the system based on 5-minute intervals.
5) Access Variables: A feature to compare the same variable across several cases. Clicking this button will open a dialog box where the user can select a list of cases and a list of variables to compare. The helper will then save all the variables into a single structure specified by the user for easy access to each case.
6) RTSCED Schedules: A plot of the real-time schedules of each generator.
7) Comparison Across Scheduling Processes: A plot comparing two scheduling processes (sub-models) for different data types (e.g., VG, Load, Net Load, variable capacity resource, conventional generation).
8) DA Reserve Prices: A plot of the day-ahead reserve clearing prices.
9) DA Reserve Schedules: A plot of the day-ahead reserve requirements and how much reserve was actually scheduled in the day-ahead commitment.
10) Generator revenues: A plot of the revenues for each generator.
11) ACE: A plot of the area control error.
12) Generation vs Load: A plot of the total, aggregate generation versus the actual load profile.
13) RT Reserve Prices: A plot of the real-time reserve clearing prices.
14) RT Reserve Schedules: A plot of the real-time reserve requirements and how much reserve was actually scheduled in the real-time dispatch.
15) Unused Thermal Capacity: A plot of the unused thermal capacity in the system per generator.
16) Energy Prices: A plot of the day-ahead and real-time locational marginal prices.
17) Instant VG Penetration: A plot of the instantaneous VG penetration for the study period.

18) Actual vs Capacity: A plot of the total online capacity per generation type and the actual dispatched capacity.
19) Number of Start-Ups: A plot of when each generation type started-up during the simulaton.
20) RT Price and ACE: A plot of the real-time energy price and area control error
21) Total Generation by Gen Type: A bar plot of the total energy delivered by each generation type.
22) Compare ACE Distributions: A plot of the ACE distribution of each case selected in the list of cases on the left side of the helper.
23) Dispatch Vs AGC Signal: A plot of the dispatch signal sent to each generator and the actual output of each generator due to the AGC signal.
24) Case Summaries: Provides a summary overview of each case in the list of cases
25) Ramp Utilization: A plot of unused thermal ramping capacity
26) DA Load Vs Actual Load: A plot of the day-ahead load profile and the actual AGC timeframe load profile.
27) Generation Stack by Type: A generation stack separated by technology for the simulation period.
28) Net Load: A plot of the net load profile (demand minus VG).
29) Online Time Per Unit: A table of the number of hours each generator was online.
30) RT Market Infeasibilities: A plot of the magnitude and occurrence of real-time market infeasibilities.
31) Number of Committed Units: A plot of the number of units committed throughout the study period by generation type.
32) VG Curtailment: A plot of the amount of VG generation being curtailed throughout the simulation.

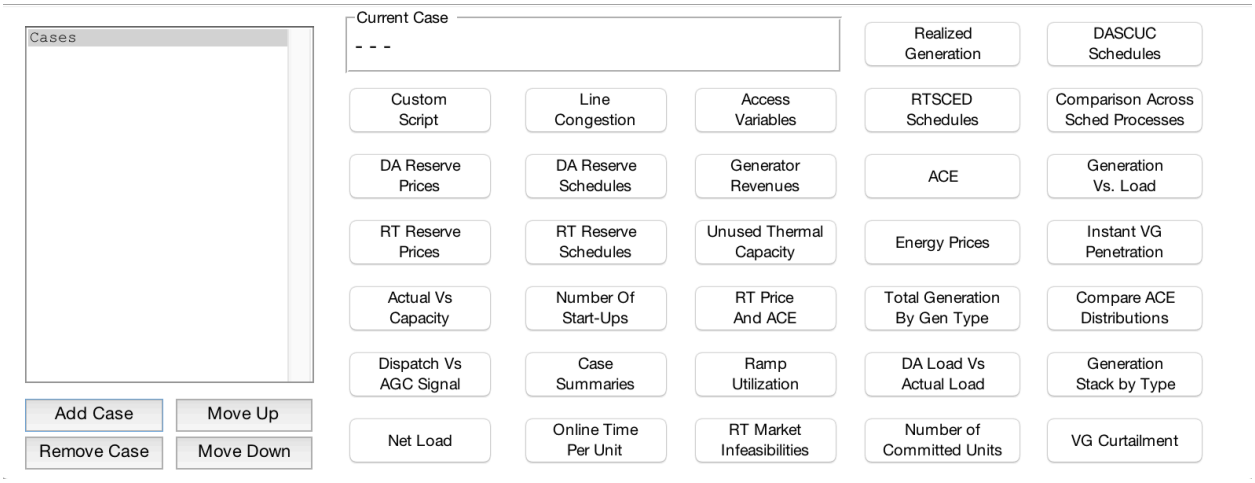


Figure 35: Screenshot of the FESTIV helper

Some of the additional plots that can be generated, besides those shown at the end of the simulation run, are shown here for the PJM 5 Bus Example.

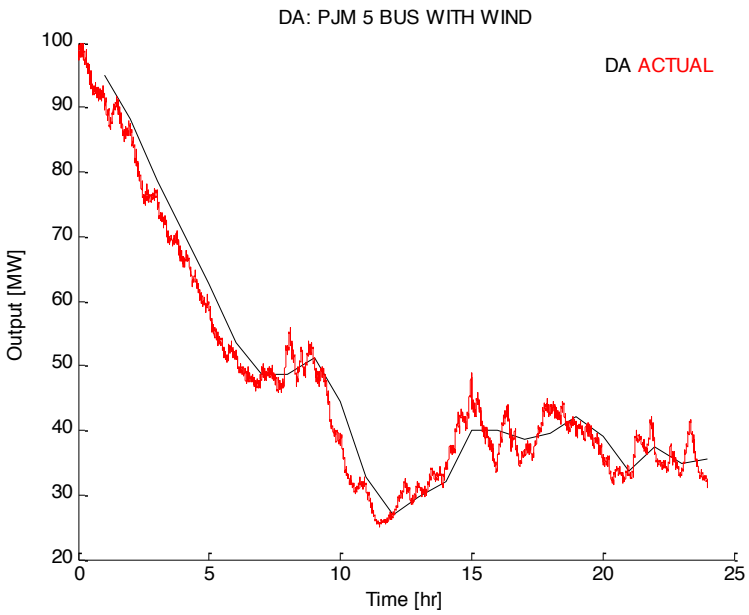


Figure 36: Day-Ahead VG vs actual VG data

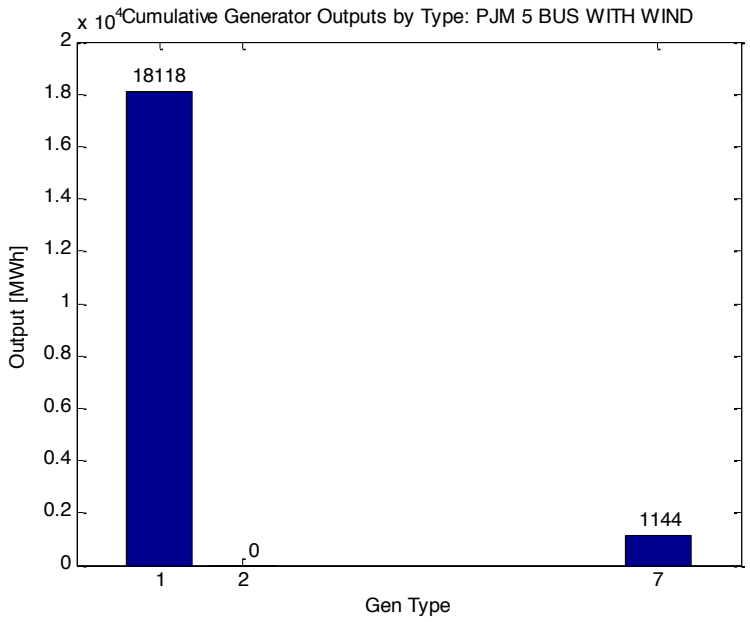


Figure 37: The total generation by generator type

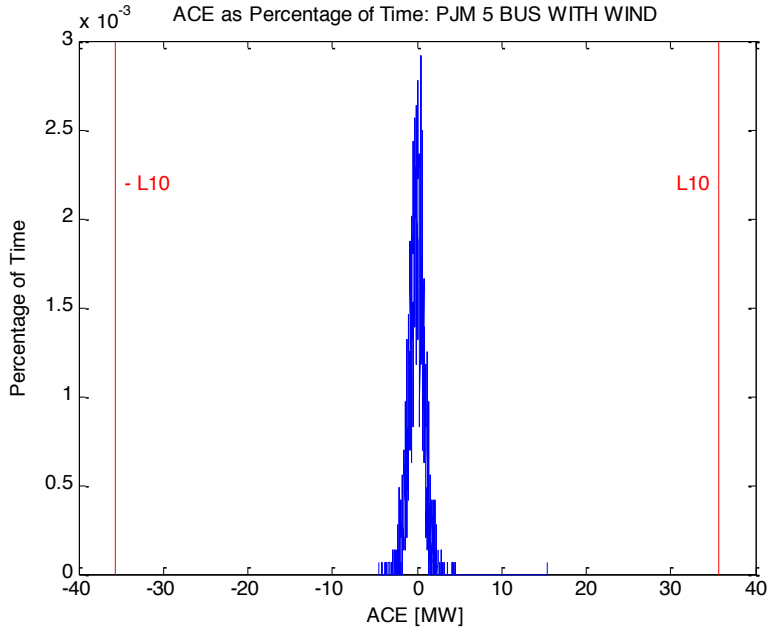


Figure 38: ACE as a distribution. L10 represents the L10 limit specified by the user for this case

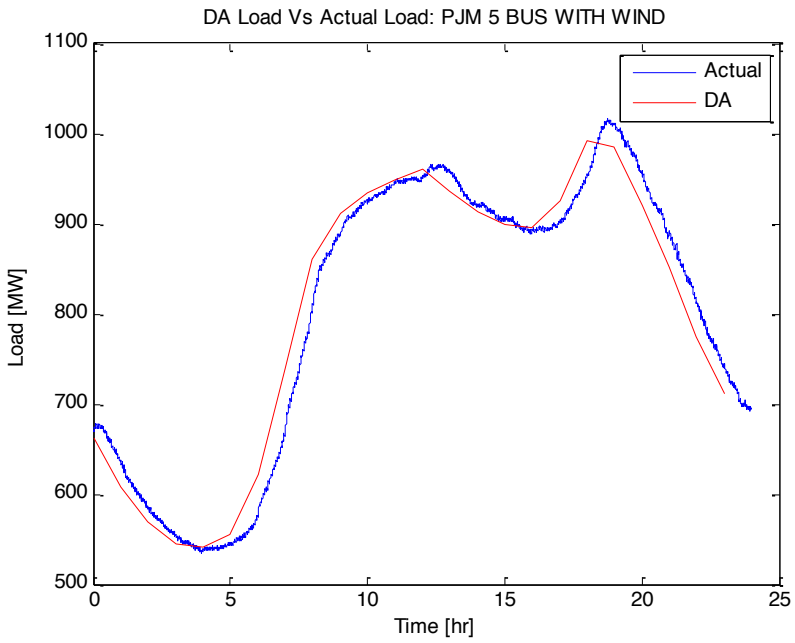


Figure 39: Day-Ahead load vs actual load data

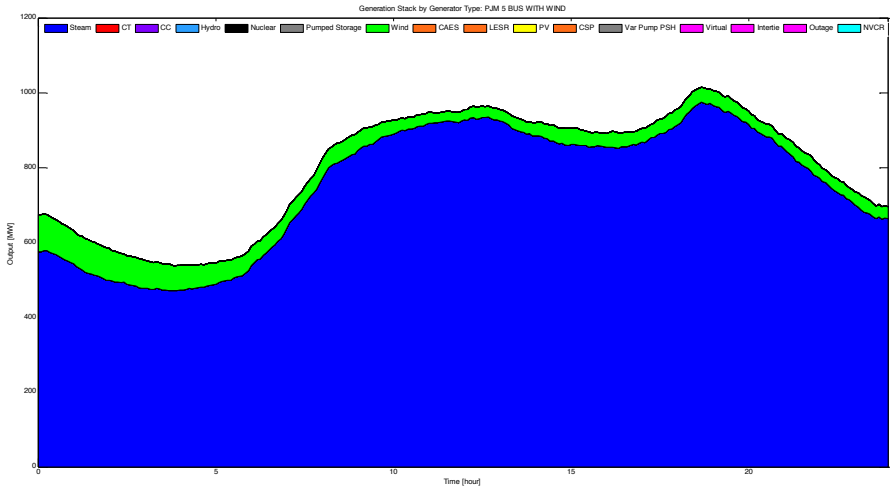


Figure 40: Generation stack over the complete study period

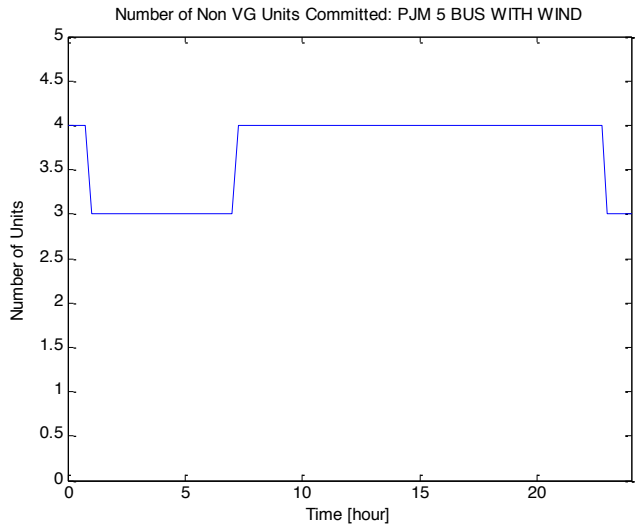


Figure 41: Number of non VG units committed per RTSCUC interval

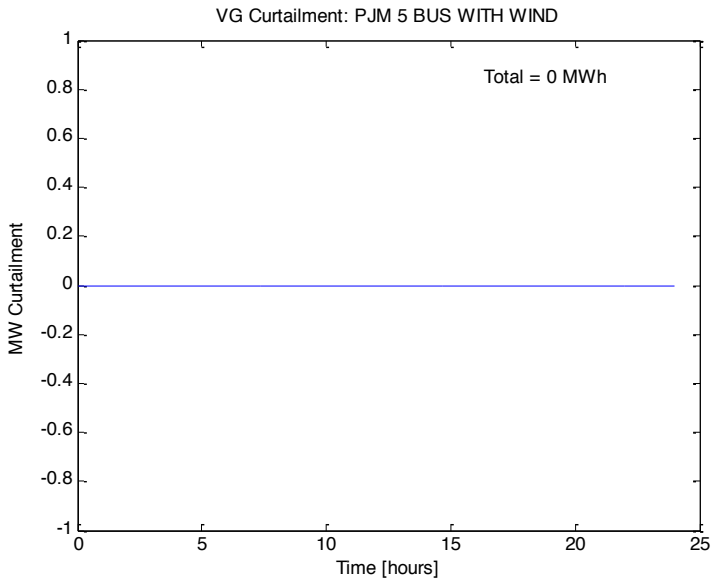


Figure 42: Actual VG curtailment and total in MWh

If the user would like access to other variables or plots that are not immediately available after a FESTIV simulation, the user can press “Access Variables” or “Custom Script”. Access Variables allows the user to compare variables/parameters across multiple cases. Custom Script (Development Team calls this a “Plot Mod”) allows for user-specified plot functions that may be useful for different studies. An example is provided with the FESTIV package to show how these can be created. The data that gets passed over is

through a struct format and must be used in a careful way to make sure the plot works. Alternatively, users can load the workspace from a saved simulation and run the custom script separately to create a new plot.

6. FESTIV Mods

One of the main benefits of FESTIV is the ability for users to add the ways in which it can be used and continue to evaluate new technologies, new operating strategies and new market designs. FESTIV has the ability to incorporate specific modifications seamlessly with relative ease once a user becomes familiar with the code. There are several different opportunities to incorporate these modifications. This includes functional mods and formulation mods. The use of mods is recommended for more advanced FESTIV users and requires more detailed knowledge of the existing default FESTIV code. It is very possible for mods to be created with significant errors that will either lead to unusable results or will not allow FESTIV to complete due to errors that stop the code. The ability to create mods takes time and carefulness, and it may require several simulations before the mod is working the way as intended. In this section, we provide some very brief introduction to the use of FESTIV mods, but for more advanced use, we recommend reaching out to the FESTIV Development Team and requesting a training session.

6.1 Configurable FESTIV Scripts

The majority of FESTIV scripts, we recommend that the user not modify, as these can have significant impacts on other features and make the probability of error in FESTIV high. However, there are a few FESTIV scripts that are configurable, and that we do recommend the user modify, if so desired. These are explained below:

1. FESTIV_ADDL_OPTIONS – This file contains a few options for the user to modify if they so wish. Typically, the modification of options here would be done with other Functional or Formulation Mods. The options are all explained within the comments of the script.
2. RTSCUCSTART – This script is run just prior to the RTSCUC and selects the set of resources that can be given commitment directions by the RTSCUC that are different from DASCUC. This can include start-up decisions, shut-down decisions, decisions to keep a DASCUC unit online even when DASCUC had it turning off, or keeping a DASCUC unit offline when DASCCUC had it turned it on. User would add logic at the bottom of the script and then modify the RTSCUCSTART_MODE_RTC and/or RTSCUCSTART_MOD_RPU in FESTIV_ADDL_OPTIONS to a number other than 1 or 2. User can also modify the commitment selection of storage units within this script.

3. RPU_TRIGGER – This script is run at each time interval if RPU is enabled within the FESTIV GUI. It triggers whether RPU should be run. By default, RPU_TRIGGER_MODE of 1, RPU is run when there is a contingency or when ACE is higher than the threshold given in the FESTIV GUI. The user can modify the script to trigger RPU based on different criteria. The user must modify RPU_TRIGGER_MODE within FESTIV_ADDL_OPTIONS
4. STOP_FESTIV – The user can stop FESTIV with a breakpoint based on time or logic. If the user wishes to stop FESTIV based on logic, the Stop_FESTIV script can be modified. For example, if the user would like to view intermediary data when a price spike happens, the user can use an "if" statement to make stop = 1 if the conditional is true. This is described more in Section 7.

6.2 Functional Mods

Beyond these given scripts, the user can create a series of additional modifications on their own, and the options are virtually limitless. The first set of Mods are called Functional Mods. These generally refer to changing inputs or outputs before or after a FESTIV routine or sub-model is making computations or decisions. A Functional Mod can be added to each sub-model as was shown in the GUI buttons in Figure 15. The set of different placements for Functional Mods are shown below.

1. Before/after the input data is initialized
2. Before/after the load, VG, and reserve forecasts are created
3. Before the FESTIV simulation Time Loop begins and at the end of a single loop
4. Before/after the DASCUC is solved
5. Before/after the RTSCUC is solved
6. Before/after the RTSCED is solved
7. Before/after actual outputs are determined
8. Before/after the ACE calculation
9. Before/after the AGC is solved
10. Before/after the SCRPU is solved
11. Before/after the post processing of the results
12. Before/after forced outages are determined
13. Before/after initial network calculations (e.g., shift factors) are calculated
14. Before/after the reliability metrics are calculated
15. Before/after the economic metrics are calculated
16. Before/after the current case is saved

Functional Mods allow for creating dynamic definitions in the code that can change between scenarios and simulations. The mods can be built as separate Matlab scripts that can be saved to the *Model Rules*

Directory. This is used to change inputs or outputs from how they may be determined from default FESTIV. For example, in the AGC sub model, the raw area control error is fed through a PI filter in order to produce a low frequency control signal to be sent to the generators. The parameters of this filter are fully configurable by the user by opening the 'AGC Param' button on the FESTIV GUI. Once these parameters are set, they are held constant throughout the entirety of the simulation. However, by utilizing Functional Mods , these filter parameters can continually be updated throughout the simulation based on the state of the system in the simulation. For example, we can create a Mod to change the length of time over which to integrate depending on the last calculated area control error: if the area control error is less than 5 MW, then the filter integrates over the last three minutes; If the area control error is between 5 MW and 10 MW, then the filter should integrate over the last five minutes; and if the area control error is greater than 10 MW, the filter should integrate over the last 10 minutes. In order to accomplish this task, a new script should be created as a Functional Mod. Create a MATLAB script called 'Update_Integral.m' and save it in the MODEL_RULES directory within the FESTIV directory.

Within this script, copy the following code:

```
if ACE(end-1,raw_ACE_index) < 5
   Type3_integral=60*3;
elseif ACE(end-1,raw_ACE_index) >= 5 && ACE(end-1,raw_ACE_index) < 10
   Type3_integral=60*5;
else
   Type3_integral=60*10;
end
```

After the Mod has been created, it must be added within the proper location via the FESTIV GUI. In this case, clicking the 'AGC Mods' button on the FESTIV GUI will open the AGC Functional Mods dialog box. Click the 'Browse' button and locate the 'Update_Integral.m' script. Once the rule has been located, click the 'Add' button that is located above the 'Mods Before AGC' list (on the left hand side of the dialog box). Clicking this button will execute the Mod before the AGC is performed. When finished, the AGC Functional Mods dialog box should look similar to Figure 43.

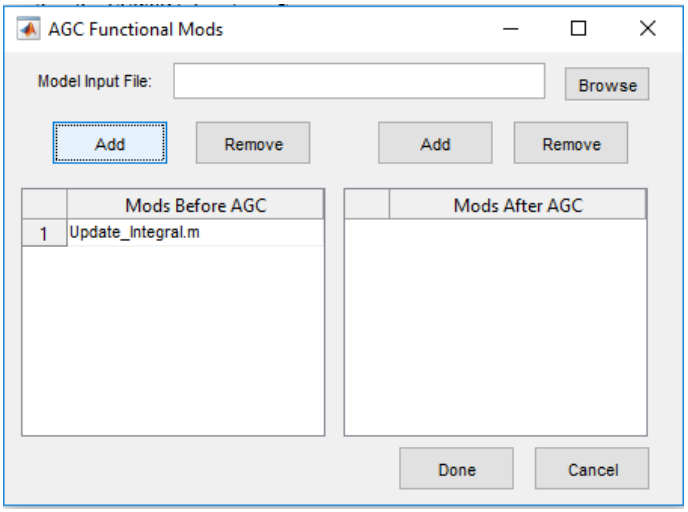


Figure 43: AGC Model Rules dialog box

Once the Mod has been successfully added, click done and run the simulation. Now, without modifying the Main FESTIV code, the ACE PI filter integral length can be dynamically modified throughout the simulation. As can be seen in Figure 44, the integral length changes between three, five, and ten minutes as defined by our model rule.

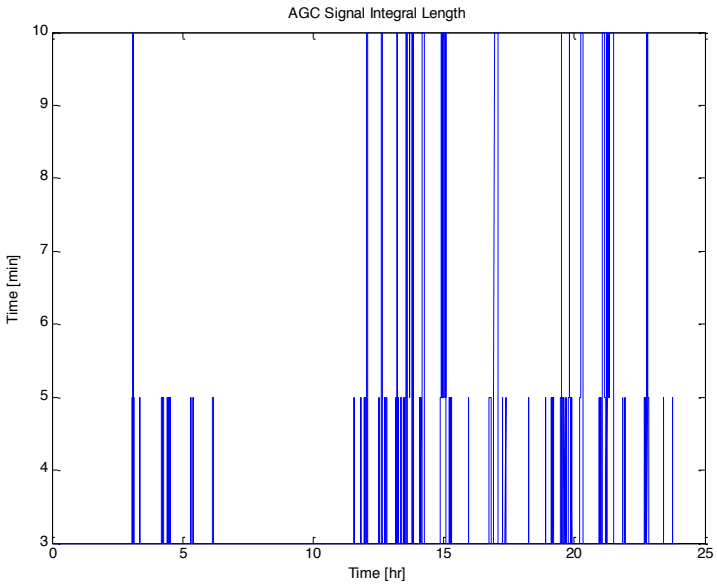


Figure 44: Example of dynamic ACE PI filter integral length

The DASCUC, RTSCUC, RTSCED, and AGC Functional Mods dialog boxes can be launched from the main FESTIV GUI by pressing the appropriately marked button. The RPU Functional Mods dialog box can be launched after pressing the 'RPU' button on the main FESTIV GUI and pressing the 'RPU Mods' button in the lower left hand corner. The remaining Mods can be added via the 'Other Func Mods' button on the

main FESTIV GUI. If changes are needed to be made to the inputs before the sub-model is run, add the Mod to the left side "Mods Before …" of the dialog box. If changes are needed to be made to the outputs from the sub-model following completion, add the Mod to the right side "Mods After …".

Note that any Mod added to a FESTIV simulation will be automatically loaded into FESTIV for any subsequent simulations. In order to remove a Mod from a simulation, the Mod must be specifically removed from the appropriate dialog box via the 'Remove' button. In addition, the full set of Mods along with the parameters in the FESTIV GUI can be saved to a data file by pressing the "Save All" button. It is important to note that this does not save all parameters within the GUI (e.g., multiple runs and Debug) nor does it save Formulation Mods. When pressing Load All, all Functional Mods and parameters from the FESTIV GUI will be loaded from a previously saved file.

The variety of Functional Mods that can be added to FESTIV is essentially limitless. General Matlab knowledge will allow for users to add various Mods, with the complexity of the Mod depending on the experience of the user. In some cases, it may be important for users to be familiarize themselves with data names in FESTIV and what data is what. The transparency of the source code can allow for users to learn this.

Below is a short list of reasons why users may want to add model rules to help illustrate the numerous opportunities:

- Adding a new parameter to a unit or branch that may be used in a later model rule not already used in FESTIV (after the input data is initialized)
- Using a new algorithm to create forecasts based on the actual data (after the load, vg, and reserve forecasts are created)
- Enforcing must-run units (before the DASCUC, RTSCUC, or RPU is solved)
- Capping pricing or changing pricing rules from those that are calculated from sub-model (After the DASCUC, RTSCUC, RTSCED, or RPU is solved)
- Changing how ACE is calculated (before/after the ACE is calculated)
- Implementing uplift rules to adjust total revenues (after the post processing of results)

6.3 Formulation Mods

The second form of Mods are related to the formulation changes of individual sub-models that use optimization. Users can add inputs to the optimization, new sets, change or introduce new constraints, and modify what constraints are used for each solution. Formulation Mods are created using .txt files. The FESTIV GUI will create a new DASCUC, RTSCUC, and RTSCED file every time it runs – if user does nothing, it will use the Default Formulation. If the user plans on modifying the formulations of any of the

optimization models, he/she must first press on “Formulation Mods” button. This will open up a new interface that allows the user to place individual Formulation Mods in the locations needed.

The GAMS models are organized in the following sequential sections:

1. Header
2. User Defined Input Block 1
3. Declare Sets
4. User Defined Input Block 2
5. Declare Parameters
6. User Defined Input Block 3
7. Load inputs from .gdx file
8. User Defined Input Block 4
9. Define/modify Sets
10. User Defined Input Block 5
11. Define/modify Parameters
12. User Defined Input Block 6
13. Declare variables
14. User Defined Input Block 7
15. Define Variables
16. User Defined Input Block 8
17. Declare equations
18. User Defined Input Block 9
19. Define the equations
20. User Defined Input Block 10
21. Model definition
22. User Defined Input Block 11
23. Declare solver options
24. User Defined Input Block 12
25. Define solve statement(s)
26. User Defined Input Block 13
27. Post processing rules
28. User Defined Input Block 14
29. Footer

Note: The developers have found no reason to use the “user defined input blocks” placed throughout. All additional Formulation Mods should fit inside the pre-specified categories following the list of all

The Optimization Formulation options dialog box is shown again here for convenience. Each area is summarized below.

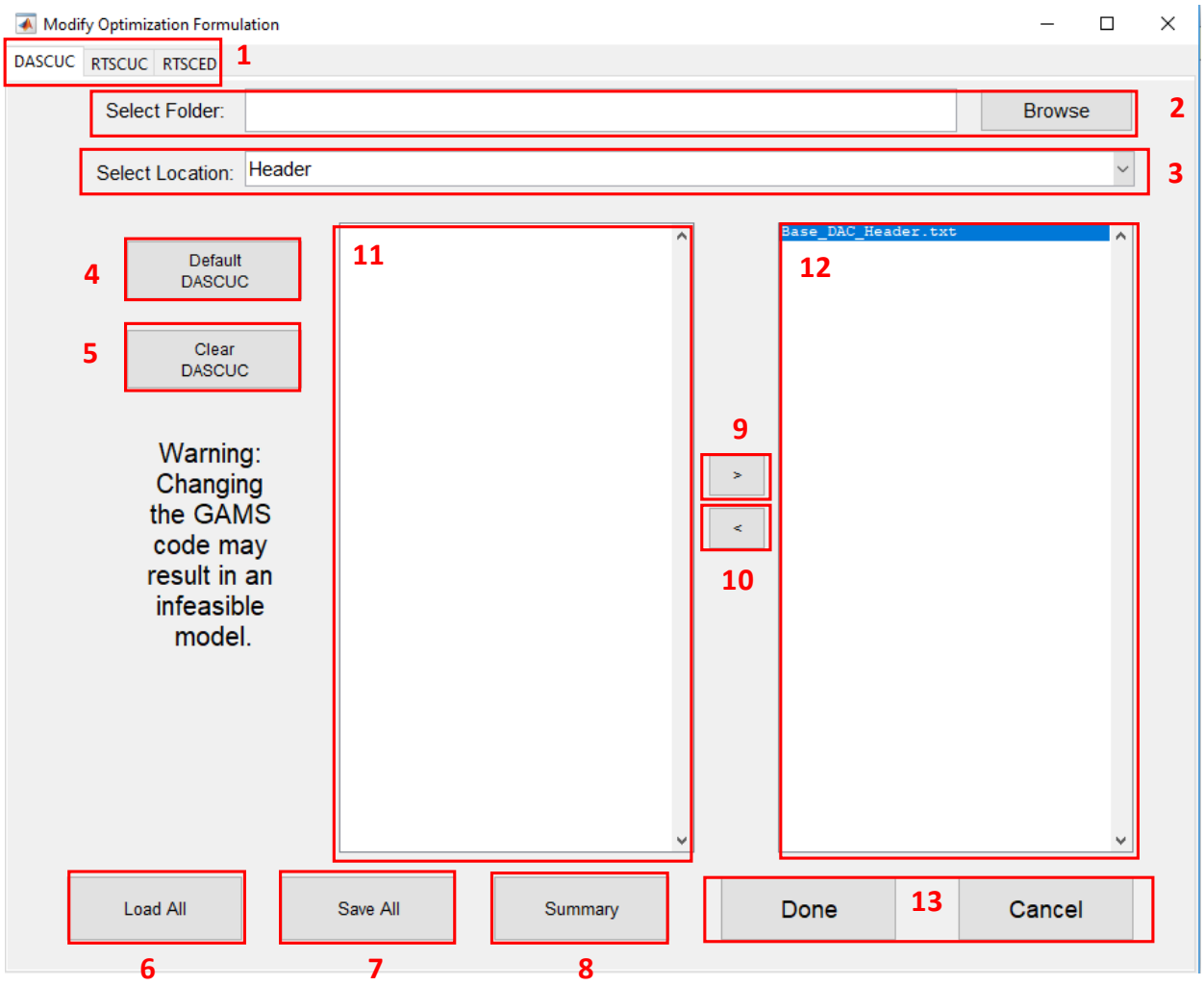


Figure 45: Optimization Formulation options dialog box details

1. GAMS model selection: These tabs are used to select which sub-model is being changed.
2. This area is used to browse for a directory containing the Formulation Mods.
3. This area is used to select which section the Formulation Mods will be modified.
4. Default DASCUC/RTSCUC/RTSCED: Pressing this button loads the default sub-model formulation for the selected sub-model

5. Clear DASCUC/RTSCUC/RTSCED: Pressing this button clears the currently loaded sub-model
6. Load All: This button can be used to load a previously saved configuration for all sub-models.
7. Save: Pressing this prompts the user for an output name and the current configuration for all sub-models is then saved as a MATLAB workspace.
8. Summary: This button prints, and displays if possible, a summary log in the form of a text file summarizing all of the Formulation Mods used for all sub-models.
9. The button is used to add a selected Mod(s) from the list in area 12 to be used as part of the configuration.
10. The button is used to remove a selected Mod from the current configuration section.
11. These buttons are used upon completion to close and build the optimization model or cancel changes.
12. This area displays all of the Formulation Mmod files in the current directory selected in area 2.
13. This area displays all of the Formulation Mods currently associated with selected configuration section shown in area 3.

The Formulation Mod files used must adhere to the following format.

1. The rule must be saved as a .txt file
2. The actual code to be utilized must be located within 2 identifiers, '=start' and '=end', that signify the beginning and ending of the new code to be written.
3. The code must adhere to all GAMS coding requirements. For more information regarding GAMS coding requirements, see the McCarl GAMS User Guide, available online. Reviewing the default code for each section will also familiarize users wanting to take advantage of Formulation Mods.

For example, if the system is constrained such that a minimum of 60% of total generation must come from thermal generators in the DASCUC, the following steps outline how this can be accomplished using Formulation Mods.

Step 1. Create a .txt file named 'DAC_Equation_Declaration_minimum_thermal_constraint.txt'

Step 2. In this file, paste the following code:

=start

EQUATION LIMITVG(INTERVAL);

=end

Step 3. Create a second .txt file named ‘DAC_Equation_Declaration_minimum_theral_constraint.txt’

Step 4. In this file, past the following code:

=start

LIMITVG(INTERVAL)..

SUM(CONVENTIONAL_GEN,GEN_SCHEDULE(CONVENTIONAL_GEN,INTERVAL)) =G= LOAD(INTERVAL)*0.60;

=end

Note: Files should be saved under MODEL_RULES\GAMS_Model_Files\. It is recommended to save files in appropriate folders corresponding to the type of Formulation Mod being written. For example, these can both be added to ‘Equations’. However, users can use different naming conventions for keeping their Formulation Mods.

Step 5. Modify the model definition file to include this constraint. It is suggested that the original model definition file is not changed. With this mind, copy the file ‘Base_CAC_Model_Definition.txt’ and rename it as ‘DAC_Model_Definition_Thermal_Gen_Constraint.txt’. In this file, add the name of the previous constraint, LIMITVG, to the end of the list of constraints in the model definition.

Step 6. This step is to select the Formulation Mods in the Optimization Formulation options dialog so that FESTIV can create the optimization code including the new code added by the user.. Using the numbered diagram in Figure 45, select the DASCUC tab in area 1. Then browse for the folder where the new Formulation Mods are located in area 2. In area 3, select *’Declare_Equations’*. In area 12, select the constraint file ‘DAC_Equation_Declaration_minimum_thermal_constraint.txt’ and click the add button shown in area 9. Do the same moving to ‘Define_Equations’ in area 3, and select the Mod

'DAC_Equation_Declaration_minimum_theral_constraint.txt' in area 12, clicking the add button shown in area 9. Then in section 3, change the section again to '*Model Definition'*. Select the base model definition file in area 13 labeled 'Base_DAC_Model_Definition.txt' and remove it by pressing the button shown in area 10. Then select the new model definition file created labeled 'DAC_Model_Definition_Thermal_Gen_Constraint.txt' and add it to the model by pressing the button in area 9. If there is a plan to use this configuration multiple times, select 'Save All' button (area 7) and provide it with a name that can be associated with the configuration. Click 'Done' in area 11 to close the dialog box. FESTIV will now write the new optimization code including the Formulation Mods that you provided.

6.3 Helper Mods

Helper Mods are scripts that can be used by the FESTIV Helper when pressing the Custom Scripts button (See Section 5.4). These are useful when there are plots and graphics that are used multiple times for different sets of results. The Helper Mods can be used for plots of either one or two cases. There is a strict naming convention that must be used for each new Helper Mod. Data must be written with the struct prefix CASE1DATA or CASE2DATA. For example, the user may want to compare the hourly costs for different cases. Create a script called Cost_Comparison_Helper.m in matlab with the following code.

```
figure
plot(sum(CASE1DATA.Cost_Result')')
hold
try
plot(sum(CASE2DATA.Cost_Result')')
catch;end;
legend(selectedNames,'interpreter','none','FontSize',14);
xlabel('Time [hr]','FontSize',14);
ylabel('Cost by hour [$/h]','FontSize',14);
title('Cost Comparison');
```

Then, when in the FESTIV Helper, select one or two cases in the left hand side. With cases selected, press the button 'Custom Script' which will open up a dialog to browse for files. Select Cost_Comparison_Helper. This will produce the graphic for costs. It is important to use try/catch commands for when the plot is creating data associated with the second case (CASE2DATA) so to avoid errors when only one case is selected. These Helper Mods should be helpful for multiple types of graphics that are not part of the default FESTIV helper.

6.3 Putting it all together

The FESTIV Mods are ways to maximize the use of FESTIV for tremendous flexibility to match study needs. However, they do take general coding expertise and a knowledge of FESTIV code and terminology. The knowledge required increases with the complexity of the Mod being added. In some cases, there may multiple mods required for a new feature that a user would like to test. For example, a user may want to

evaluate a simulation where system inertia is required and simulated to affect system frequency and ensure that a frequency nadir does not trigger under-frequency load shedding. Examples of the types of Mods that may be needed are shown below. It is important to note that whenever a new parameter or set is required as a Formulation Mod, a Functional Mod before the DASCUC/RTSCUC/RTSCED is almost always also needed to determine the input and write the data so that the optimization model can read it. Any new Formulation Mod that adds variable that needs to be recorded for all time periods and saved as a result at the end of the FESTIV simulation also requires a Functional Mod for after the DASCUC/RTSCUC/RTSCED to read the GDX file and retain the data.

1. A Functional Mod may be required for "before forecast creation" to set hourly inertia requirements.
2. A Functional Mod would be required for "before DASCUC/RTSCUC/RTSCED" to put the inertia requirement for the particular time period into a GDX file so that GAMS can read it
3. A Formulation Mod would be required for parameter declaration to declare the inertia requirement value.
4. A Formulation Mod would be required for variable declaration to declare a unit inertia variable
5. A Formulation Mod would be required for equation declaration for any equations limiting a resource inertia and an equation for system-level inertia.
6. A Formulation Mod would be required for equation definition to list in the specific constraint definitions.
7. A Formulation Mod would be required for model definition with the new constraints added.
8. A Functional Mod may be required for "after DASCUC/RTSCUC/RTSCED" to read the different amounts of inertia each resource is providing (and potential prices if applicable).
9. A Functional Mod may be required for "after the Actual Output Determination" to determine the inertia available at that instant.
10. A Functional Mod may be required for "before/after the ACE Calculation" to calculate how the system inertia affects frequency and frequency nadir and whether it impacts under-frequency load shedding.
11. A Helper Mod may be useful to evaluate the system inertia over time for comparison of different cases.

6.4 Data Naming Assistance

In order to maximize the use of FESTIV Mods, it is helpful to know the naming conventions used in FESTIV and to have an idea on what are the important results and parameters that are used throughout the simulations. Here we will provide some examples. We note that given the complexity of the software tool it is impossible to capture all parameters throughout. The development team has attempted to name terms in a way that best describes what the term represents, and often users can look through the code to fully

determine the meaning behind every data piece used in FESTIV. The forum discussions can be used to provide further information when uncertain. Below are a few important highlights to help users. As the user group grows and users have common questions and suggestions, the development team will try to expand on this information to help users understand

- **Indices and sizes –** Indices are important for various uses especially around data matrices with multiple data types. This includes system, generator, branch, and reserve parameter data. Rather than use the assumed column number from these data as shown in the Main Input File, users should use the actual index name when needed. All index names can be found by opening the file 'DECLARE_INDICES'. In addition, generator types and branch types should use the name of the type as shown in 'DECLARE_INDICES' rather than the assumed type numbers. Finally, the size of different sets of information are often useful during creations of Mods. Following are examples of size terms that may be useful: ngen (number of generators), nbus (number of buses), nbranch (number of branches), nblock (number of blocks which is equal to the largest number of blocks for all resources) and nreserve (number of reserve). Additional sizes for subsets include nvg (number variable generators), nESR (number of energy storage resources), nvcr (number of variable capacity resources), npar (number of phase angle regulators), nhvdc (number high voltage dc branches), and nctgc (number of contingencies monitored).
    - Due to the many parameters of energy storage resources and their often small presence in many simulations, storage resources are also indexed separately from other generators to avoid large sparse matrices. Many times the storage index and gen index needed to be cross-indexed when looped with one index for matrix with indices of the other. For these cases, the two parameters storage_to_gen_index and gen_to_storage_index can be used to ensure accurate data manipulation.
- **Scripts** – All scripts and functions can be run separately within Functional Mods. SCRIPTS used by FESTIV are all in capital letters. To determine which scripts are available, one can view the main FESTIV directory.
- **Major parameters –** the primary matrix with input data that is often used for Functional Mods
    - **SYSTEMVALUE_VAL –** various system data
    - **GENVALUE_VAL** – set of generator parameter data
    - **BRANCHVALUE_VAL –** parameters for branches
    - **RESERVEVALUE_VAL –** parameters of reserve types
    - **STORAGEVALUE_VAL –** parameters for energy storage resources
    -

- **Forecast time series data –** The following terms are used for the various timeseries that FESTIV uses within the Matlab portion of the code.
    - **ACTUAL_LOAD_FULL –** actual load for system-wide input.
    - **ACTUAL_VG_FULL –** actual output for all VG plants without curtailment. DAC
    - **DAC_LOAD_FULL, RTC_LOAD_FULL, RTD_LOAD_FULL –** load forecast for the corresponding sub-model.
    - **DAC_VG_FULL, RTC_VG_FULL, RTD_VG_FULL –** VG forecast for all VG plants without curtailment for the corresponding sub-model.
    - **DAC_RESERVE_FULL, RTC_RESERVE_FULL, RTD_VG_FULL –** Reserve requirements for the corresponding sub-model.
    - **DAC_INTERCHANGE_FULL, RTC_ INTERCHANGE _FULL, RTD_ INTERCHANGE _FULL –** Pre-determined interchange schedules when using fixed interchanges for the corresponding sub-model.
- **Network Data –** The following terms are used for network data:
    - **PTDF_VAL –** bus to branch power transfer distribution factors. (nbranch rows by nbus columns)
    - **GENBUS_CALCS_VAL –** gen to bus relationship. (ngen rows by three columns). Column 1 gen index, column 2 bus index, column 3 participation factor (1 if only one injection point)
    - **BRANCHBUS_CALCS_VAL –** branch to bus relationship. (nbranch rows by three columns). Column 1 branch index, column 2 from bus index, column 3 to bus index.
    - **PTDF_PAR_VAL –**PAR to branch sensitivity factor, how much flow change for 1 degree change in PAR angle (nbranch rows by nbranch columns). Zeros for non-PAR columns
    - **LODF_VAL –** branch to branch line outage distribution factor, sensitivity of change in flow on one branch on another branch (nbranch rows by nbranch columns)
    - **BUS_HVDC_VAL and BUS_PAR_VAL –** whether a bus is connected to an HVDC line or a PAR line.
    - **SYSTEMVALUE_VAL(slack_bus,1) –** index of slack bus
    - **SYSTEMVALUE_VAL(mva_pu,1) –**system base
- **DEFAULT_DATA** – this is a struct that contains all the original data that is read from the Main Input File. FESTIV rereads DEFAULT_DATA for each sub-model to reset. Such that if data is adjusted using Functional Mods, they will be reset in a separate sub-model. For example, if GENVALUE_VAL(genA,capacity) is changed to 50% of its nameplate capacity in RTSCUC using a Functional Mod, the value will not be modified in the next RTSCED because the beginning of RTSCED, GENVALUE_VAL = DEFAULT_DATA.GENVALUE_VAL.

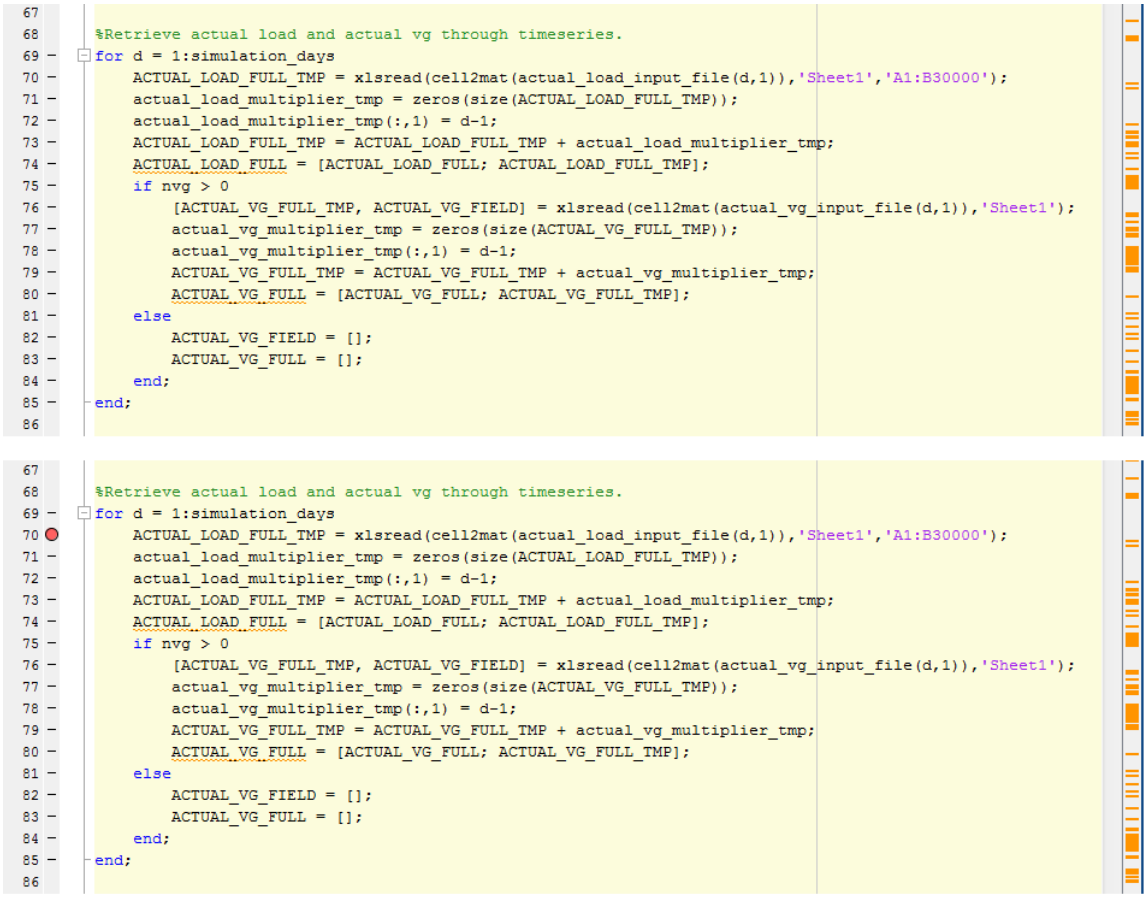
- **Important output results**
    - **STATUS** – STATUS keeps track of the unit status of all resources and will always have the latest status based on RTSCUC interval resolution. STATUS can be updated by DASCUC, RTSCUC, and RPU. STATUS has RTSCUC intervals as rows, and ngen + 1 columns, with the first column as time and other columns representing generator index (i.e., generator with index of 1 is in column 2). PUMPSTATUS is similar but for storage resources status in the charging mode.
    - **DISPATCH** – DISPATCH keeps track of the energy dispatch schedule of all resources and will always have the latest dispatch schedule based on RTSCED interval resolution. DISPATCH can be updated by RTSCED and RPU by default. DISPATCH has RTSCED intervals as rows, and ngen + 1 columns, with the first column as time and other columns representing generator index (i.e., generator with index of 1 is in column 2). PUMPDISPATCH is similar but for storage resources status in the charging mode.
    - **AGC_SCHEDULE** – AGC_SCHEDULE keeps track of the AGC schedule of all resources and will always have the latest agc schedule based on AGC interval resolution. AGC_SCHEDULE can be updated by AGC. AGC_SCHEDULE has AGC intervals as rows, and ngen + 1 columns, with the first column as time and other columns representing generator index (i.e., generator with index of 1 is in column 2). Unlike other output results, AGC_SCHEDULE contains both discharging and charging values with charging values as negative
    - **ACTUAL_GENERATION** – ACTUAL_GENERATION keeps track of the realized output of all resources and will always have the latest dispatch schedule based on AGC interval resolution. ACTUAL_GENERATION can be updated by ACTUALS_SIMULATOR . ACTUAL_GENERATION has AGC intervals as rows, and ngen + 1 columns, with the first column as time and other columns representing generator index (i.e., generator with index of 1 is in column 2). ACTUAL_PUMP is similar but for storage resources status in the charging mode.
    - **ACE –** Contains the ACE data as determined by ACE_CALCULATOR. It includes all forms of ACE as columns with the first column as time for every AGC intervals as row.
- **_VAL vs. .val –** The suffix _VAL should be used for Functional Mods and various conditional code throughout. The suffix .val should be only used for data that is directly written to GDX file for use within GAMS.

7. Debugging FESITV

FESTIV incorporates several debugging functions in order to facilitate the identification and solving of potential run time issues. Depending on the reason for debugging FESTIV, there are different tools available to help identify potential issues in order to resolve them.

7.1 Debugging Tools in MATLAB

MATLAB contains a useful feature for examining code for FESTIV execution errors known as the 'debugging mode.' This feature will break MATLAB execution at a pre-specified point in the program and allow the user to investigate variable values. This is particularly useful for FESTIV because it allows the user to ensure that the program is behaving as expected. In order to manually set a breakpoint, open the program code in the MATLAB editor. After the code has opened, click on the dash that precedes the line of code where you would like to break MATLAB execution. The dash will be covered by a red circle indicating that a breakpoint has been set at that line. MATLAB will pause execution at the line with the breakpoint before executing that line.



```
67
68    %Retrieve actual load and actual vg through timeseries.
69 -  for d = 1:simulation_days
70 -      ACTUAL_LOAD_FULL_TMP = xlsread(cell2mat(actual_load_input_file(d,1)),'Sheet1','A1:B30000');
71 -      actual_load_multiplier_tmp = zeros(size(ACTUAL_LOAD_FULL_TMP));
72 -      actual_load_multiplier_tmp(:,1) = d-1;
73 -      ACTUAL_LOAD_FULL_TMP = ACTUAL_LOAD_FULL_TMP + actual_load_multiplier_tmp;
74 -      ACTUAL_LOAD_FULL = [ACTUAL_LOAD_FULL; ACTUAL_LOAD_FULL_TMP];
75 -      if nvg > 0
76 -          [ACTUAL_VG_FULL_TMP, ACTUAL_VG_FIELD] = xlsread(cell2mat(actual_vg_input_file(d,1)),'Sheet1');
77 -          actual_vg_multiplier_tmp = zeros(size(ACTUAL_VG_FULL_TMP));
78 -          actual_vg_multiplier_tmp(:,1) = d-1;
79 -          ACTUAL_VG_FULL_TMP = ACTUAL_VG_FULL_TMP + actual_vg_multiplier_tmp;
80 -          ACTUAL_VG_FULL = [ACTUAL_VG_FULL; ACTUAL_VG_FULL_TMP];
81 -      else
82 -          ACTUAL_VG_FIELD = [];
83 -          ACTUAL_VG_FULL = [];
84 -      end;
85 -  end;
86
```

```
67
68    %Retrieve actual load and actual vg through timeseries.
69 -  for d = 1:simulation_days
70 -      ACTUAL_LOAD_FULL_TMP = xlsread(cell2mat(actual_load_input_file(d,1)),'Sheet1','A1:B30000');
71 -      actual_load_multiplier_tmp = zeros(size(ACTUAL_LOAD_FULL_TMP));
72 -      actual_load_multiplier_tmp(:,1) = d-1;
73 -      ACTUAL_LOAD_FULL_TMP = ACTUAL_LOAD_FULL_TMP + actual_load_multiplier_tmp;
74 -      ACTUAL_LOAD_FULL = [ACTUAL_LOAD_FULL; ACTUAL_LOAD_FULL_TMP];
75 -      if nvg > 0
76 -          [ACTUAL_VG_FULL_TMP, ACTUAL_VG_FIELD] = xlsread(cell2mat(actual_vg_input_file(d,1)),'Sheet1');
77 -          actual_vg_multiplier_tmp = zeros(size(ACTUAL_VG_FULL_TMP));
78 -          actual_vg_multiplier_tmp(:,1) = d-1;
79 -          ACTUAL_VG_FULL_TMP = ACTUAL_VG_FULL_TMP + actual_vg_multiplier_tmp;
80 -          ACTUAL_VG_FULL = [ACTUAL_VG_FULL; ACTUAL_VG_FULL_TMP];
81 -      else
82 -          ACTUAL_VG_FIELD = [];
83 -          ACTUAL_VG_FULL = [];
84 -      end;
85 -  end;
86
```

Figure 46: Example of setting a breakpoint in MATLAB

This feature is useful in correcting faulty run time options. For example, the PJM 5 bus example that is setup with one day of data cannot be simulated for a week. If at the FESTIV GUI the user sets the execution time for a week, the following error will be generated:

```
************************************************************************
* Flexible Energy Scheduling Tool for Integrating Variable Generation *
************************************************************************
**   FFFFFF    EEEEEE    SSSSSS    TTTTTTTT    IIIIII    VV      VV   **
**   FF        EE        SS            TT        II       VV    VV    **
**   FFFFFF    EEEEEE    SSSSSS        TT        II        VV VV     **
**   FF        EE            SS        TT        II         VVV      **
**   FF        EEEEEE    SSSSSS        TT       IIIIII       V       **
************************************************************************
**************** National Renewable  Energy Laboratory ****************
************************************************************************

Input File: PJM_5_BUS.xlsx
Reading Input Files...Index exceeds matrix dimensions.

Error in FESTIV (line 70)
    ACTUAL_LOAD_FULL_TMP =
    xlsread(cell2mat(actual_load_input_file(d,1)),'Sheet1','A1:B30000');

>>
```

Figure 47: Example of a FESTIV run time error

The above error gives several important pieces of information. First, the cause of the error is provided, in this case, there was an indexing error. Second, the location of the error is also provided, in this case, line 70 in FESTIV.m. By placing a breakpoint at line 70, we can inspect the value of the variables being operated on to discover the cause of the error.

```
67
68      %Retrieve actual load and actual vg through timeseries.
69 -    for d = 1:simulation_days
70 -        ACTUAL_LOAD_FULL_TMP = xlsread(cell2mat(actual_load_input_file(d,1)),'Sheet1','A1:B30000');
71 -        actual_load_multiplier_tmp = zeros(size(ACTUAL_LOAD_FULL_TMP));
72 -        actual_load_multiplier_tmp(:,1) = d-1;
73 -        ACTUAL_LOAD_FULL_TMP = ACTUAL_LOAD_FULL_TMP + actual_load_multiplier_tmp;
74 -        ACTUAL_LOAD_FULL = [ACTUAL_LOAD_FULL; ACTUAL_LOAD_FULL_TMP];
75 -        if nvg > 0
76 -            [ACTUAL_VG_FULL_TMP, ACTUAL_VG_FIELD] = xlsread(cell2mat(actual_vg_input_file(d,1)),'Sheet1');
77 -            actual_vg_multiplier_tmp = zeros(size(ACTUAL_VG_FULL_TMP));
78 -            actual_vg_multiplier_tmp(:,1) = d-1;
79 -            ACTUAL_VG_FULL_TMP = ACTUAL_VG_FULL_TMP + actual_vg_multiplier_tmp;
80 -            ACTUAL_VG_FULL = [ACTUAL_VG_FULL; ACTUAL_VG_FULL_TMP];
81 -        else
82 -            ACTUAL_VG_FIELD = [];
83 -            ACTUAL_VG_FULL = [];
84 -        end;
85 -    end;
86
```

simulation_days: 1x1 double = 7

Figure 48: Sample utilization of debugging mode in MATLAB

MATLAB displays a green arrow at the current line of execution while in debugging mode. In this example, a breakpoint was placed at line 70 (the line previously indicated as the location of the error as shown in Figure 47). By hovering the mouse over a variable, we are able to see the current value of that variable. In this case, the variable ‘simulation_days’ contains a value of seven rather than the appropriate value of one. After identifying the error, the mistake can be corrected and the simulation can continue.

Breakpoints can also be set at a certain time within the FESTIV Simulation Time Loop. As an example, in an infeasible or strange solution occurred in a FESTIV simulation at a certain time within the simulation (e.g., RTSCED at 11:05), then the user may want to run the simulation again and pause just before the strange solution was determined to evaluate inputs and review the entire solution in GAMS (shown in next subsection). The user can open "Debug" button in the FESTIV GUI and then enter in the time to pause the simulation (e.g., 10:59). The FESTIV simulation will now pause once it gets to that time so the user can slowly work through what caused the solution to be as it is.

Finally, breakpoints can be automatically set in FESTIV in response to certain conditions in the simulation. The FESTIV directory contains a script labeled 'STOP_FESITV.m'. This script should include a stopping criterion to suspend the FESTIV simulation and enter the debugging mode in MATLAB and so should be adjusted by the user depending on debugging needs. If the stopping criterion is satisfied the script should set the value of a variable 'stop' to be equal to one. For example, in order to stop the execution of FESTIV and enter debugging mode if a generator is starting up, the following code can be used in 'STOP_FESTIV.m' :

```
if sum(sum(UNIT_STARTINGUP_VAL)) > 0

stop = 1;

end
```

When MATLAB is operating in debugging mode, there are several useful features. As mentioned earlier, by hovering the mouse over a variable in the MATLAB editor, you can see the current value of that variable. By pressing the 'Step' button in the debugging toolbar, the FESTIV code can be executed one line at a time. The 'Run Section' button will run the current section of the FESTIV code. The user can also go through Functional Mods and see if there is an error, and correct it right away while in Debug Mode.

7.2 Debugging Tools in GAMS

If there is a failure in the optimization sub-models, FESTIV will break execution automatically and attempt to open the appropriate .lst file. These files are the output of GAMS optimizations and hold information regarding the infeasibilities. If the infeasibility occurs in the DASCUC or RTSCUC optimizations, these optimizations are rerun with the integer constraint relaxed, i.e. integer variables are converted to continuous variables and then the optimization is rerun. After the relaxed optimization is rerun, FESTIV will attempt to open the .lst file for further inspection. Within the .lst file, there is a section labeled 'Solve Summary' that will detail the status of the model and solver. A sample of this section is provided below:

```
            S O L V E     S U M M A R Y


    MODEL   SCUC              OBJECTIVE  PRODCOST
    TYPE    MIP               DIRECTION  MINIMIZE
    SOLVER  CPLEX             FROM LINE  3508


**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      8 Integer Solution
**** OBJECTIVE VALUE          838599.8168

 RESOURCE USAGE, LIMIT         14.306      1800.000
 ITERATION COUNT, LIMIT      9231        500000
```

As can be seen from the above excerpt, the solver completed normally and was able to find an integer feasible solution. In the event of an infeasibility, a summary similar to the one below may be produced by GAMS.

```
               S O L V E     S U M M A R Y


     MODEL   RTD               OBJECTIVE  PRODCOST
     TYPE    LP                DIRECTION  MINIMIZE
     SOLVER  CPLEX             FROM LINE  2140


**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      4 Infeasible
**** OBJECTIVE VALUE               4.9583

 RESOURCE USAGE, LIMIT          0.187      1000.000
 ITERATION COUNT, LIMIT       438    2000000000
```

The solve report summary will show how many constraints were infeasible. A sample of this portion of the .lst file is shownbelow.

```
**** REPORT SUMMARY :        0     NONOPT
                          1 INFEASIBLE (INFES)
                 SUM         4.9583
                 MAX         2.4792
                 MEAN        2.4792
```

This optimization resulted in one infeasible constraint. In order to track down the infeasibility, search for the term ‘infes’ in the .lst file. This will lead to the constraints that were infeasible. For the previously discussed optimization, the following constraint was infeasible:

EQU Q_GENLIMIT_HIGH CAPACITY CONSTRAINT

		LOWER	LEVEL	UPPER	MARGINAL
TG11	.1	-INF	3.3542	3.5000	.
TG11	.2	-INF	2.9167	3.5000	.
TG11	.3	-INF	2.4792	.	-1.0000 INFES
TG11	.4	-INF	2.6875	3.5000	.
TG11	.5	-INF	2.2500	3.5000	.

From the excerpt of the .lst file above, the generator named ‘TG11’ caused the constraint ‘Q_GENLIMIT_HIGH’ to be infeasible during the 3rd optimization interval. The .lst file also shows that the upper bound on the constraint was zero megawatts (the use of the period signifies a value of zero). Since the generator was on before and after the interval in question, this would lead us to suspect the status of the generator is incorrectly set, since the upper limit for the generation capacity of a unit should only be zero if that unit is turned off. Further inspection of the unit status variable does verify that the unit status was incorrectly set to zero (off) instead of one (on). Fixing this mistake allows the optimization to solve normally.

If the infeasibility occurs in either the DASCUC model or the RTSCUC model, FESTIV will execute another script named ‘DEBUG_DASCUC’ or ‘DEBUG_RTSCUC’ respectively. As mentioned before, these are the scripts that will relax the integer constraints (i.e. turn the integer variables into continuous variables) and resolve the optimization. Upon completion, the appropriate .lst file will open and you can begin the debugging process as discussed in this section.

References

References

E. Ela, M. Milligan, and M. O'Malley, ""A flexible power system operations simulation model for assessing wind integration," Proceedings of IEEE PES General Meeting 2011, Detroit, MI.

E. Ela, M. O'Malley, "Studying the Variability and Uncertainty of Variable Generation at Multiple Timescales," IEEE Transactions on Power Systems, vol. 27, no. 3, pp. 1324-1333, Aug. 2012.

GAMS: The Solver Manuals, 2011, version 23.7, Washington, DC.

Matlab and Matlab Simulink, version R2011a, Natick, MA: The Mathworks Inc., 2011.

Ferris, M., 2005, "MATLAB and GAMS: Interfacing Optimization and Visualization Software," Mathematical Programming Technical Report 98-10, Computer Sciences Department, University of Wisconsin, Madison, Wisconsin.

North American Electric Reliability Corporation, June 2010, Reliability Standards for the Bulk Electric Systems of North America.

Carrion, M., and Arroyo, J., Aug. 2006, "A computationally efficient mixed-integer linear formulation for the thermal unit commitment problem," *IEEE Transactions on Power Systems,* 21 (3), pp. 1371-1378.

Arroyo, M., and Conejo, A.J., Aug. 2004, "Modeling of start-up and shut-down power trajectories of thermal units," *IEEE Transactions on Power Systems*, 19 (3), pp. 1562-1568.