

CCSITM
Carbon Capture Simulation Initiative

Framework for Optimization, Quantification of Uncertainty and Surrogates User Manual

Version 2.5.0

Nov 2018



Copyright (c) 2012 - 2018

Copyright Notice

FOQUS was produced under the DOE Carbon Capture Simulation Initiative (CCSI), and is copyright (c) 2012 - 2018 by the software owners: Oak Ridge Institute for Science and Education (ORISE), Los Alamos National Security, LLC., Lawrence Livermore National Security, LLC., The Regents of the University of California, through Lawrence Berkeley National Laboratory, Battelle Memorial Institute, Pacific Northwest Division through Pacific Northwest National Laboratory, Carnegie Mellon University, West Virginia University, Boston University, the Trustees of Princeton University, The University of Texas at Austin, URS Energy & Construction, Inc., et al.. All rights reserved.

NOTICE. This Software was developed under funding from the U.S. Department of Energy and the U.S. Government consequently retains certain rights. As such, the U.S. Government has been granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable, worldwide license in the Software to reproduce, distribute copies to the public, prepare derivative works, and perform publicly and display publicly, and to permit other to do so.

License Agreement

FOQUS Copyright (c) 2012 - 2018, by the software owners: Oak Ridge Institute for Science and Education (ORISE), Los Alamos National Security, LLC., Lawrence Livermore National Security, LLC., The Regents of the University of California, through Lawrence Berkeley National Laboratory, Battelle Memorial Institute, Pacific Northwest Division through Pacific Northwest National Laboratory, Carnegie Mellon University, West Virginia University, Boston University, the Trustees of Princeton University, The University of Texas at Austin, URS Energy & Construction, Inc., et al. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Neither the name of the Carbon Capture Simulation Initiative, U.S. Dept. of Energy, the National Energy Technology Laboratory, Oak Ridge Institute for Science and Education (ORISE), Los Alamos National Security, LLC., Lawrence Livermore National Security, LLC., the University of California, Lawrence Berkeley National Laboratory, Battelle Memorial Institute, Pacific Northwest National Laboratory, Carnegie Mellon University, West Virginia University, Boston University, the Trustees of Princeton University, the University of Texas at Austin, URS Energy & Construction, Inc., nor the names of its contributors may be

used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

You are under no obligation whatsoever to provide any bug fixes, patches, or upgrades to the features, functionality or performance of the source code ("Enhancements") to anyone; however, if you choose to make your Enhancements available either publicly, or directly to Lawrence Berkeley National Laboratory, without imposing a separate written license agreement for such Enhancements, then you hereby grant the following license: a non-exclusive, royalty-free perpetual license to install, use, modify, prepare derivative works, incorporate into other computer software, distribute, and sublicense such enhancements or derivative works thereof, in binary and source code form.

Abstract

The Framework for Optimization, Quantification of Uncertainty, and Surrogates (FOQUS) serves as the primary computational platform enabling advanced Process Systems Engineering (PSE) capabilities to be integrated with commercial process simulation software. It can be used to synthesize, design, and optimize a complete carbon capture system while considering uncertainty. FOQUS enables users to effectively screen potential capture concepts in the context of a complete industrial process so that trade-offs can be appropriately evaluated. The technical and economic performance characteristics of the capture process are highly dependent on employing an effective approach for process synthesis. Since large-scale carbon capture processes are outside of current experience, heuristic and evolutionary approaches are likely to be inadequate. Thus, a key aspect of FOQUS is that it bridges this gap by supporting a superstructure-based approach to determine the optimal process configuration and equipment interconnections.

FOQUS consists of several modules.

1. SimSinter provides a wrapper to enable models created in process simulators to be linked into a FOQUS Flowsheet.
2. The FOQUS Flowsheet is used to link simulations together and connect model variables between simulations on the flowsheet. FOQUS enables linking models from different simulation packages.
3. Simulations are run through Turbine, which manages the multiple runs needed to build surrogate models, perform derivative-free optimization or conduct an Uncertainty Quantification (UQ) analysis. Turbine provides the capability for job queuing and enables these jobs to be run in parallel using cloud- or cluster-based computing platforms or a single workstation.
4. The Automated Learning of Algebraic Models for Optimization (ALAMO) module can create algebraic surrogate models to support large-scale deterministic optimization, including superstructure optimization to determine process configurations. The ALAMO module is an external product due to background Intellectual Property (IP) issues.
5. The Derivative-Free Optimization (DFO) module enables derivative-free (or simulation-based) optimization directly on the process models linked together on a FOQUS Flowsheet. It utilizes Excel to calculate complex objective functions, such as the cost of electricity.
6. The UQ module enables the effects of uncertainty to be propagated through the complete system model, sensitivity of the model to be assessed, and the most significant sources of uncertainty identified to enable prioritizing of experimental resources to obtain additional data.
7. The Optimization Under Uncertainty (OUU) module combines the capabilities of the DFO and the UQ modules to enable scenario-based optimization, such as optimization over a range of operating scenarios.
8. The SolventFit module is an uncertainty quantification tool for the calibration of an Aspen Plus solvent process model. The current state of the art is a regression that yields single best fit point estimates of some parameters. This shows neither the level of uncertainty in each parameter, nor the level of uncertainty in model output, such as equivalent work. SolventFit

allows for predictions with uncertainty bounds by accounting for uncertainty in model parameters and deficiencies in the model form. This yields an improved understanding of the model parameters and results in more complete predictions with uncertainty bounds. This distribution of parameters allows for predictions with uncertainty.

Version Log

Version Number	Release Date	Description
2.0.0	03/31/2018	Initial Open Source Release
2016.06.00	06/22/2016	2016 June Bug Fix Release
2016.04.00	04/29/2016	2016 April Bug Fix Release
2016.02.00	02/29/2016	2016 February Bug Fix Release
2015.10.00	11/20/2015	2015 November IAB Release
2015.06.00	6/30/2015	2015 June Bug Fix Release
2014.10.00	10/31/2014	2014 October IAB Release

Table of Contents

1. Introduction	1
1.1. Simulation Based Optimization	1
1.2. Uncertainty Quantification	2
1.3. Optimization Under Uncertainty	4
1.4. Heat Integration	4
1.5. Surrogate Models	5
2. Flowsheet and Settings	7
2.1. Starting FOCUS	7
2.2. Session and Menu	7
2.3. Settings	12
2.4. Flowsheet	17
2.5. Sample Results	26
2.6. Tutorial – Example Files	28
2.7. Tutorial – Creating a Flowsheet	28
2.8. Tutorial – Creating a Flowsheet with Linked Simulations	34
2.9. Tutorial – Flowsheets with Recycle	40
2.10. Tutorial – Flowsheet Result Data	44
2.11. Tutorial – Using a Remote Turbine Instance	51
3. Optimization	54
3.1. Problem Set Up	54
3.2. Solver Options	60
3.3. Running Optimization	61
3.4. Tutorial – Optimization	63
3.5. Tutorial – Parameter Estimation	68
4. Uncertainty Quantification	79
4.1. UQ User Interface	79
4.2. UQ Tutorial	90
4.3. File Formats	128
5. SolventFit	133
5.1. SolventFit Instructions	133
6. Optimization Under Uncertainty	137
6.1. OUU Variables	137
6.2. OUU Objective Functions	137
6.3. Mathematical Formulations	138
6.4. OUU User Interface	139
6.5. OUU Tutorial	140
7. Heat Integration	145
7.1. Plug-in Models	145
7.2. Heat Integration Tutorial	151
8. Surrogate Models	167

8.1. Surrogate Models Overview	167
8.2. Tutorial – ALAMO	171
8.3. Tutorial – ACOSSO	186
8.4. Tutorial – BSS-ANOVA	187
8.5. Tutorial – iREVEAL	189
8.6. Tutorial – Surrogates with UQ Tools	189
8.7. Tutorial – Surrogates with the Flowsheet	191
9. SimSinter	193
9.1. SimSinter Configuration	193
9.2. Aspen Custom Modeler Configuration	193
9.3. Aspen Plus Configuration	203
9.4. Microsoft Excel Configuration	209
9.5. gPROMS Configuration	214
9.6. DMF Integration	227
10. Debugging	228
10.1. How to Debug	228
10.2. Known Issues	228
10.3. Reporting Issues	229
11. References	230

To obtain support for the products within this package, please send an email to
ccsi-support@acceleratecarboncapture.org.

1. INTRODUCTION

The Framework for Optimization, Quantification of Uncertainty and Surrogates (FOQUS) software provides a graphical interface and standard platform for several Carbon Capture Simulation Initiative (CCSI) tools. The primary feature of FOQUS is its ability to interact with commonly-used chemical engineering process modeling software. Models constructed using a variety of software can be combined into a larger composite model. CCSI tools SimSinter and the Turbine Science Gateway (TSG) provide connectivity to external process simulation software. SimSinter provides a standard library to enable interfacing with other software; TSG provides a simulation job queuing system that can be used on: (1) a single workstation, (2) networked workstations, (3) cluster, or (4) cloud computing resources.

In FOQUS, simulations can be connected in a meta-flowsheet, which enables parts of a process to be modeled using the most appropriate software and combines them into a single large model, possibly including recycle streams. For example, in studying a carbon capture system for a coal-fired power plant: a power plant may be modeled in Thermoflex; a solvent-based carbon capture system may be modeled in Aspen Plus; and a compression system may be modeled in gPROMS. To optimize the entire system, these models can be combined into a single large model. The resulting meta-flowsheet can be used for simulation-based optimization, uncertainty quantification (UQ), or generation of surrogate models. Information extracted from the simulations can also be used to construct a heat integration optimization problem to determine the best way to use excess heat in the process. Heat integration can be combined with simulation-based optimization.

This section provides brief overview and motivating examples, for different uses of FOQUS.

Simulation Based Optimization

Simulation-based optimization considers a process simulation to be a black box model, which is a model where the mathematical details are not known. In this case, models are evaluated using process simulation software; multiple models can be combined to form larger models. Due to the long run times and the limitations of the methods used, a limited set of optimization variables (usually less than 30) is considered. Simulation-based optimization has some advantages and disadvantages, compared to equation-based optimization methods. With simulation-based optimization, there is no need to provide simplified algebraic models, problem formulation is relatively simple, and a good solution can usually be obtained; however, a provably-global optimum cannot be found and it is impractical to deal with very large numbers of variables. Large numbers of variables may be found in superstructure and heat integration problems where the structure of a process is being optimized. Both simulation and equation-based optimization methods are used in CCSI.

Capture of CO₂ from a pulverized coal-fired power plant involves several very different systems including: a boiler, steam cycle, flue gas desulfurization, carbon capture, and CO₂ compression. It is convenient to separate many of these processes into smaller, more reliable simulations. The different processes may also be better simulated in different software packages. Although some process simulation software contains optimization features, there are several reasons these may not be practical for a large composite system. It may be hard to develop a large model of the entire system that reliably converges. Many optimization methods have a difficult time dealing

with simulation errors, and many black box derivative free optimization solvers are better able to handle occasional simulation failures. It may not be practical to simulate the entire process accurately using a single tool. Derivatives are also difficult to estimate for many systems when models do not provide exact derivatives, making derivative-free methods a good option.

The motivating example used to demonstrate the optimization framework is fairly simple. The system consists of a series of bubbling fluidized bed (BFB) CO₂ adsorbers and regenerators modeled in Aspen Custom Modeler (ACM). The details of the BFB system are described in the CCSI BFB model documentation. A cost analysis for a 650 MW power plant and capture system is presented in an Excel spreadsheet. The simulation and spreadsheet files are provided in the examples directory in the FOCUS installation directory (see the tutorial in Section 2.8 for more information). The spreadsheet contains capital cost as well as operating and maintenance cost estimates, which are used to estimate the cost of electricity.

In this example, the objective function is the cost of electricity; the decision variables are design and operating variables in the ACM model. The cost of electricity is minimized while maintaining a 90 CO₂ percent capture rate. The BFB system model and the cost of electricity are contained in separate models connected in a FOCUS flowsheet, which enables the cost of electricity to be calculated in Excel, using data acquired from the ACM model. See Sections 2.8 and 3.4 for more information about the optimization problem.

Uncertainty Quantification

The Uncertainty Quantification (UQ) module of FOCUS encompasses a rich selection of mathematical, statistical, and diagnostic tools for application users to perform UQ studies on their simulation models. The PSUADE tool provides most of the UQ functionality available in FOCUS [1]. The recommended systematic multi-step approach consists of the following steps:

1. Define the objectives of the analysis (e.g., identify the most important sources of uncertainties).
2. Specify a simulation model to be studied. Acquire the model input files and the executable that runs the simulation (i.e., an executable that uses the specified inputs and generates model outputs). Identify the outputs of interest, identify all relevant sources of uncertainties, and ensure that they can be used as input variables to the simulation model.
3. Select some or all input parameters that have uncertainty attributed. Characterize the prior probability distribution of these selected parameters by specifying the upper/lower bounds. For non-uniform prior distributions (e.g., Gaussian), additional information (e.g., mean and standard deviation) is required to define the shape of the prior distribution. This prior distribution represents the user's best initial guess about the selected parameters' uncertainties.
4. Identify, if available, relevant data from physical experiments that can be used for model parameter calibration. Model calibration is a process that applies the observational data to update the prior distribution. The model calibration correlates the observational data to predict a distribution as a result.
5. Select a sample scheme and sample size. From this information, a set of input values are

sampled from the prior distribution. The choice of sampling scheme (which affects how the samples populate the input space) depends on the UQ objective(s) specified in Step 1.

6. “Run” the input samples. Running the input samples is the process where each sampled input value is fed to the simulation executable (specified in Step 2) and the corresponding output value is returned.
7. Analyze the results and make decisions on how to proceed.

Steps 1-4 are often done through expert knowledge elicitation and/or literature search. Steps 5-7 can be achieved through software provided in the FOCUS UQ module.

The FOCUS UQ module provides a number of sampling and analysis methods, including:

- Parameter screening methods: computes the importance of input parameters to identify which are important (to be kept in subsequent analyses) and which to ignore (to be weeded out).
- Response surface (used interchangeably with ‘surrogate’) construction: approximates the relationship between the input samples and their outputs via a smooth mathematical function. This response surface or surrogate can then be used in place of the actual simulation model to speed up lengthy simulations.
- Response surface validation methods: evaluates how well a given response surface fits the data. This is important for choosing different response surfaces.
- Basic uncertainty analysis: propagates input uncertainty to output uncertainty.
- Sensitivity analysis methods: quantifies how much varying an input value can impact the resulting output value.
- Bayesian calibration: applies observational data to refine the estimate of input uncertainties.
- Visualization tools: views computed distributions and response surfaces.
- Diagnostics tools (mainly in the form of scatter plots): checks samples and model behaviors (e.g., outliers).

The adsorber 650.1 subsystem process model is used to demonstrate the UQ framework. The A650.1 process model was developed and is continuously refined by our Process Synthesis and Design Team. The model is based on their design and optimization of an initial full-scale design of a solid sorbent capture system for a net 650 MW (before capture) supercritical pulverized coal power plant. The A650.1 model describes a solid sorbent-based carbon capture system that uses the NETL-32D sorbent. NETL-32D is a mixture of polyethyleneamine (PEI) and aminosilanes impregnated into the mesoporous structure of a silica substrate. CO₂ removal is achieved through chemical reactions between the amine sites within the sorbent. The A650.1 model is implemented in Aspen Custom Modeler (ACM) and contains many components (e.g., adsorbers, regenerators, compressors, heat exchangers). For the UQ analyses, this manual focuses is on the adsorber units, which are responsible for the adsorption of CO₂ from the input flue gas.

In its original form, the A650.1 model consists of a deterministic simulation model, which means to consider all the parameters (e.g. chemical reaction parameters, heat and mass transfer coefficients) to have a fixed value (most likely fixed to a mean value, lower or upper bound for robustness). With the FOCUS UQ module, the model uncertainties can be addressed. Thus, UQ analysis of the A650.1 model would help to develop a robust design by addressing the following questions:

- How accurately does each subsystem model predict actual system performance (under uncertain operating conditions)?
- Which input parameters should be examined to improve prediction accuracy?
- What is each input parameters' contribution to prediction uncertainty?

Optimization Under Uncertainty

The Optimization Under Uncertainty (OUU) module in FOCUS is an extension of simulation-based optimization by including the contribution of model parameter uncertainties in the objective function. OUU is useful when inclusion of uncertainties may significantly alter the optimal design configurations. A straightforward approach to include the effect of uncertainty is to replace the objective function with its statistical mean on an ensemble drawn from the probability distributions of the continuous uncertain parameters (other options are available in FOCUS). Alternatively, users can provide a set of ‘scenarios’, where each scenario is associated with a probability. The latter case is often called ‘scenario optimization.’ The FOCUS OUU accommodates both continuous and scenario-based uncertain parameters. OUU makes use of the flowsheet for evaluations of the objective function. Naturally, OUU requires more computational resources than deterministic optimization. However, the ensemble runs can be launched in parallel so ideally, the turnaround time remains about the same as that of deterministic optimization if high performance computing capability (such as the CCSI Turbine gateway) is used in conjunction with FOCUS.

Heat Integration

The Heat Integration tool maximizes heat utilization within the entire process. For example, in a carbon capture process, the regenerator needs to be heated by steam while the adsorber (or absorber) needs to be cooled by water. Heat integration aims to match all available heat sources and sinks within the process so that heating or cooling requirements can be satisfied internally as much as possible without using utilities. In carbon capture, heat integration not only reduces steam consumption for the regenerator, but also recovers part of heat from the adsorber to be used in other parts of the process. The heat integration tool is tightly integrated with process simulation and optimization software. The graphical interface transfers the relevant information from process simulation results to heat integration inputs. The heat integration tool consists on a mathematical model solved using mathematical programming to minimize utility usage. The General Algebraic Modeling System (GAMS) software is used for this purpose. Finally, the graphical interface sends the heat integration results to other simulation inputs or optimization inputs. Capital cost calculation for the heat exchanger network is not considered in the current version of heat integration tool due to its complexity.

There are many heat integration opportunities in a power plant with steam cycles, carbon capture

processes and CO₂ compressors. For example, potential heat sources include adsorbers in the capture process and intercoolers between CO₂ compressors. Potential cold sources include feed water heaters in the steam cycle and regenerators in the capture process. Note: In a power plant, steam is usually extracted from the steam cycle instead of purchasing it from outside; therefore, heat integration not only reduces utility cost but also increases the net power output and efficiency. The motivating example consists of a BFB CO₂ adsorber/regenerator process model (in ACM), a multi-stage CO₂ compression process model (in ACM), steam cycle calculations (in Python), and heat integration calculations (in GAMS). Heat integration returns minimum utility cost, minimum hot and cold utility consumption, and minimum number of heat exchangers. These results can be used as final outputs or inputs for steam cycle calculations. Correlations for net power outputs to steam extraction or heat addition, which are obtained from the Thermoflex model, are used in steam cycle calculations. Steam cycle calculations provide the net power output and efficiency with carbon capture and sequestration (CCS), and possibly heat integration. This example demonstrates the net efficiency of a power plant with CCS, which can be (potentially) significantly increased after using the heat integration tool.

Surrogate Models

Process simulations are often time consuming and occasionally fail to converge. For mathematical optimization, it is sometimes necessary to replace a simulation with a surrogate model, which is a simplified model that executes much faster. FOCUS contains tools for creating and quantifying the uncertainty associated with surrogate models.

ALAMO

While simulation based optimization, can often do a good job of providing optimal design and operating conditions for a predetermined flowsheet, it cannot provide an optimal flowsheet. To obtain a more optimal flowsheet, a mixed integer nonlinear program must be solved. These types of problems cannot generally be solved using simulation based optimization. A solution is to generate relatively simple algebraic models that accurately represent the high fidelity models. FOCUS currently provides an interface for ALAMO [2], which builds surrogate model that are well suited for superstructure optimization.

ACOSSO

The Adaptive Component Selection and Shrinkage Operator (ACOSSO) surface approximation was developed under the Smoothing Spline Analysis of Variance (SS-ANOVA) modeling framework [3]. As it is a smoothing type method, ACOSSO works best when the underlying function is somewhat smooth. For functions which are known to have sharp changes or peaks, etc., other methods may be more appropriate. Since it implicitly performs variable selection, ACOSSO can also work well when there are a large number of input variables. To facilitate the description of ACOSSO, the univariate smoothing spline is reviewed first. The ACOSSO procedure also allows for categorical inputs [4].

BSS-ANOVA

The Bayesian Smoothing Spline ANOVA (BSS-ANOVA) is essentially a Bayesian version of ACOSSO [5]. It is Gaussian Process (GP) model with a non-conventional covariance function that borrows its form from SS-ANOVA. It tackles the high dimensionality (of inputs) on two fronts: (1) variable selection to eliminate uninformative variables from the model and (2) restricting the level of interactions involved among the variables in the model. This is done through a fully Bayesian approach which can also allow for categorical input variables with relative ease. Since it is closely related to ACOSSO, it generally works well in similar settings as ACOSSO. The BSS-ANOVA procedure also allows for categorical inputs [4].

2. FLOWSHEET AND SETTINGS

This chapter provides general information about using FOCUS and constructing flowsheets. The FOCUS flowsheet provides the basis for the tools described in Chapters 3 through 8. The Sections 2.1 through ?? provide an overview of the FOCUS user interface, and Sections 2.6 through 2.11 provide tutorials.

Starting FOCUS

The Windows FOCUS installer will add a FOCUS menu under “CCSI Tools” in the Windows start menu. Upon software installation (see the FOCUS bundle installation manual), the user can start the graphical interface using one of the following methods:

1. Select FOCUS from the Windows start menu. Note: FOCUS may be slow starting the first time, which oftentimes is related to installed anti-virus software.
2. Double-click a file with the “.foqus” extension in the Windows file explorer. FOCUS will automatically load the selected file.
3. The easiest way to launch FOCUS from the command line, if it has been installed from the Windows installer, is to open “Turbine Console” from the FOCUS menu in the Windows start menu. The following command can be used in the command prompt:
 - `foqus`
 - `foqus --load <path to session file>`

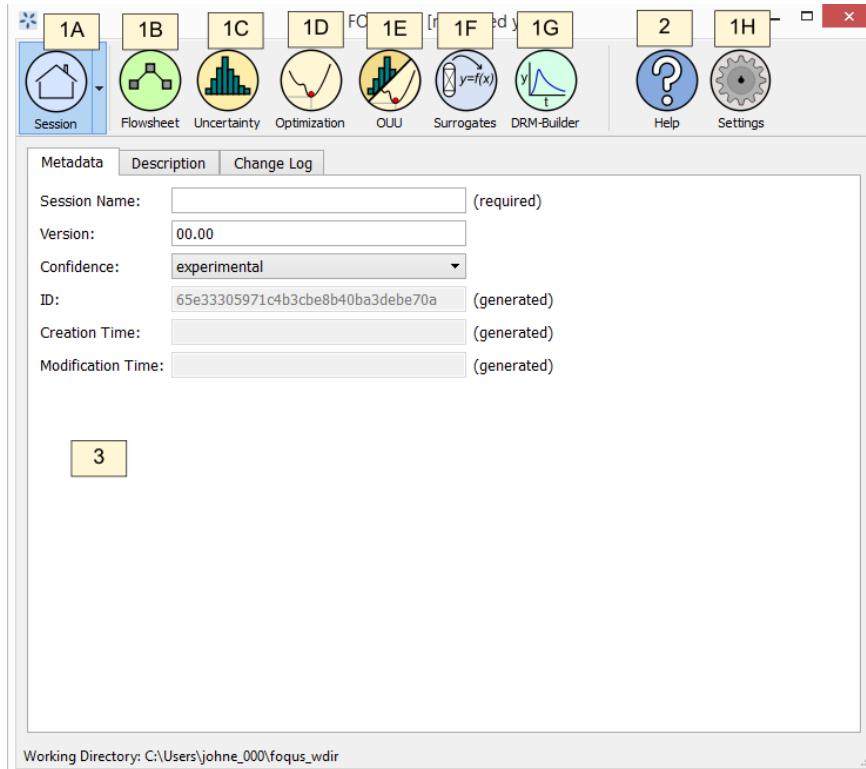
The load option can take a relative or absolute path. If a relative path is specified, it is relative to the directory where the command is executed.

The first time FOCUS is started, the user is prompted to specify a working directory. The working directory preference is stored in “%APPDATA%\foqus.cfg” on Windows (APPDATA is an environment variable). On Linux or OSX, the working directory is specified in “\$HOME/.foqus.cfg”. Additionally the user can override the working directory when starting FOCUS by using the “--working_dir <working dir>” or “-w <working dir>” command line option. Log files, user plugins, and files related to other FOCUS tools are stored in the working directory. The working directory can be changed at a later time from within FOCUS. A full list of FOCUS command line arguments is available using the “-h” or “--help” arguments.

Session and Menu

Session Information Display

Information related to a flowsheet and tool settings is organized into a FOCUS session. The session screen displays information about the current session. A menu is available by clicking the Session drop-down menu. Figure 1 illustrates the Home window.

**Figure 1: Home Window**

1. The buttons displayed at the top of the Home window, excluding **Help**, are tab-like buttons that change the window when selected. The depressed button indicates the currently displayed window.
 - (a) **Session** displays the Session window, which contains a description of the session that is currently open. **Session** has a drop-down menu that displays the Session menu, which is illustrated in Figure 2.
 - (b) **Flowsheet** displays the meta-flowsheet editing window.
 - (c) **Uncertainty** displays the interface for PSAUDE and UQ visualization.
 - (d) **Optimization** displays the simulation-based optimization interface.
 - (e) **OUU** displays the optimization under uncertainty interface.
 - (f) **Surrogates** displays the surrogate model generation window.
 - (g) **DRM-Builder** displays the dynamic reduced model builder, which can be used to develop reduced models for dynamic simulations.
 - (h) **Settings** displays the main FOQUUS settings window.
2. **Help** toggles the Help browser. The Help browser contains HTML help, as well as additional licensing information about other libraries and software included in the FOQUUS installation.
3. The main Session window displays information about the current session and is divided into three tabs:

- **Metadata** displays information about the current FOCUS session. The **Session Name** provides a descriptive name for the session. This name is used by the data management framework and when running flowsheets remotely, so a name is required. Entering a name should be the first step in creating a FOCUS flowsheet. **Version** number can be used to keep track of changes to a FOCUS session. **Confidence** describes whether the FOCUS session is expected to produce reliable results or not. **ID** is a unique identifier to identify a particular saved version of the session. **Creation Time** is the date and time that the flowsheet was first saved. **Modification Time** is the time and date that the flowsheet was last saved.
- **Description** displays a detailed explanation of the purpose of the current session file, the problem being solved, and other useful information provided by the creator of the session file.
- **Change Log** displays a record of changes made to the file. If the **Automatically create backup session file, when saving changes** checkbox is selected in FOCUS **Settings**, a backup file should exist for entries in the **Change Log**. The backup can be matched to the **Change Log** by the unique identifier appended to the file name.

Session Menu

Figure 2 illustrates the **Session** menu displayed with the data management framework (DMF) features disabled. When using the DMF, the **Open Session** and **Save/Save Session As** options include using the local file system or DMF database. To learn more about DMF options, see Section ??.

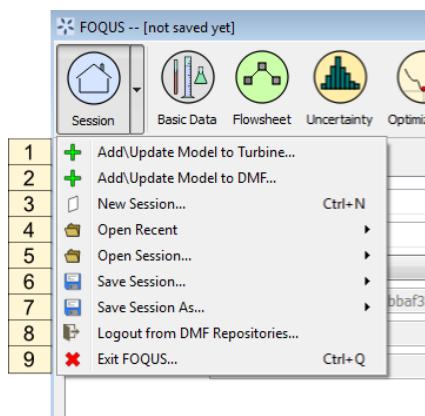


Figure 2: Home Window, Session Drop-Down Menu

The menu items in Figure 2 are:

1. **Add\Update Model to Turbine** enables additional models to be uploaded to Turbine. Turbine provides simulation job queuing functionality so models cannot be run in FOCUS until they have been added to the Turbine server.

2. **Add\Update Model to DMF** enables additional models to be uploaded to the DMF. If models are uploaded to the DMF FOCUS can automatically upload the models to Turbine as needed.
3. **New Session** clears all session information so that a new session can be started.
4. **Open Recent** shows a list of recently open FOCUS sessions that can be quickly reloaded for convenience.
5. **Open Session** opens a session that was previously saved to a file.
6. **Save Session** saves the current session with the current session file name. If the session has not been previously saved, the user will be prompted to enter a file name. **Save Session** commands the user to save two session files: (1) a file with the selected name and (2) if backup option is enabled, a backup file with a name constructed from the **Session Name** and **ID**. The Session **ID** is shown on the **Session, Metadata** tab. The backup file is saved to the working directory. This system prevents accidental saving over an important file. It also enables the user to open any previously saved session.
7. **Save Session As** is similar to **Save Session**; however, the user is prompted for a new file name.
8. **Logout from DMF Repositories** Allows the user to logout of a DMF server.
9. **Exit FOCUS** exits FOCUS. The user is asked whether to save the current session before exiting.

Adding or Changing Turbine Simulations

Before running any flowsheet where a node is linked to a simulation, the simulation must be uploaded to the Turbine gateway. To use a simulation at least two things are required: (1) the simulation file (e.g., Aspen Plus file, Excel file) and (2) the SimSinter configuration. The SimSinter configuration file is a JavaScript Object Notation (JSON) formatted file that specifies the simulation, input, and output. Any additional files required to run the simulation must also be uploaded.

Figure 3 illustrates the dialog box for uploading a simulation to Turbine.

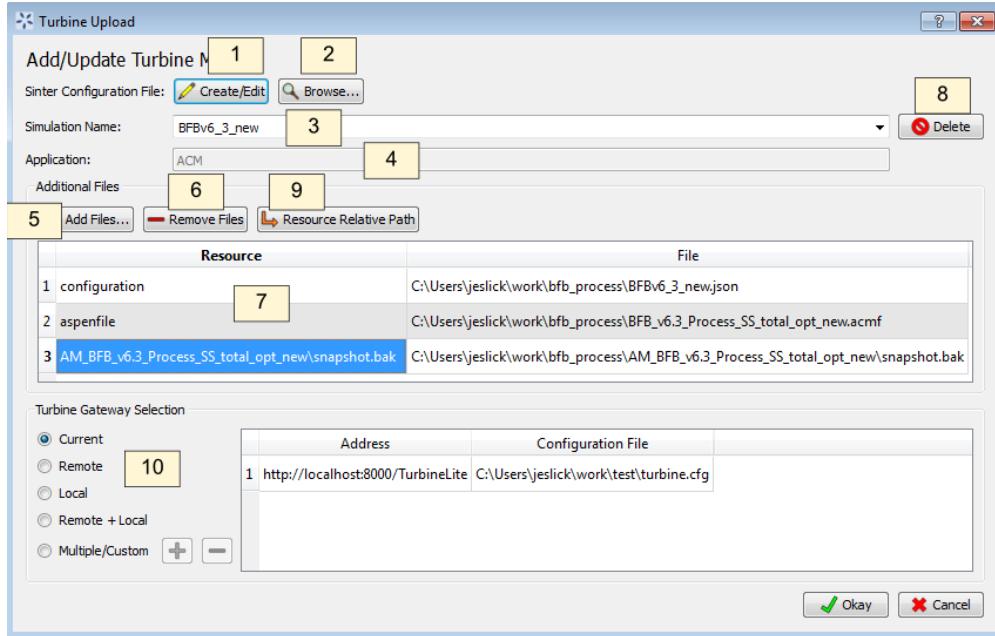


Figure 3: Turbine Upload Dialog Box

1. **Create/Edit** enables use of the SimSinter configuration Graphical User Interface (GUI) to create a SimSinter configuration file. See Chapter 9 and the SimSinter documentation for more information.
2. **Browse** displays a file browser, which can be used to select an existing SimSinter configuration file. Once a SimSinter configuration file is selected, the **Application** type is filled in. The SimSinter **Configuration File** and simulation file are automatically added to the file upload table.
3. **Simulation Name** enables entry of a new name if uploading a new simulation. An existing simulation can be selected from the drop-down list if an existing simulation is being modified. After selecting a SimSinter configuration file, the simulation name is guessed from the SimSinter configuration file name, but it can be edited.
4. **Application** displays the application that will be used to run the simulation. This is filled in automatically based on information in the SimSinter configuration file, and cannot be edited.
5. **Add Files** enables uploading of any auxiliary files that may be required by the simulation. Multiple files may be selected at once.
6. **Remove Files** enables added files to be removed from the list of files to upload.
7. **File Table** displays a list of files to be uploaded to Turbine.
8. **Delete** allows the simulation with the name currently displayed in the **Simulation Name** drop-down list to be deleted from Turbine. Only simulations that have not been run can be deleted.
9. **Resource Relative Path** enables the user to set the path of resource files relative to the simulation working directory. To set the directory, select files in the **File Table**. Multiple

files can be selected. Click **Resource Relative Path**, and type the relative path to assign to the selected resource files.

10. **Turbine Gateway Selection** enables the user to select the instance of Turbine to which to upload the simulation. **Current** is the Turbine instance currently set to run simulations. **Remote** is configured Remote instance. **Local** is the TurbineLite instance installed on the local computer. **Remote + Local** allows simulations to be uploaded to both the local and remote instances of Turbine. **Multiple/Custom** allows simulations to be uploaded to other Turbine instances by selecting Turbine configuration files.

Adding or Changing Data Management Framework Simulations

Please see Section ?? for information about setting up the data management framework (DMF) for use with FOCUS.

The DMF allows storing and tracking changes for simulations. Simulations from the DMF can be used in the FOCUS flowsheet, and FOCUS will automatically upload the simulations to Turbine as needed. The procedure for adding and updating simulations in the DMF is similar to uploading to Turbine, described in Section 2.2.3.

The upload to DMF dialog is shown below in Figure 4. To change between the local DMF lite and DMF server, use the **SelectDMF** button (see item 1 in Figure 4). For more information see Section 2.2.3 and ??.

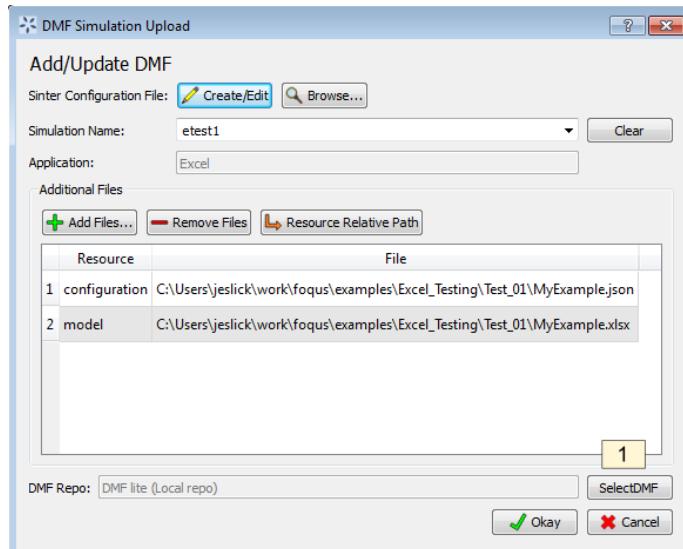


Figure 4: DMF Upload Dialog Box

Settings

The settings screen shows FOCUS settings that are related to the general FOCUS setup, and are unlikely to change between sessions. The settings screen is accessible by clicking the **Settings** button

at the top of the Home window. The FOCUS settings can be stored in two locations: (1) “%APPDATA%\foqus.cfg” on Windows or “\$HOME/.foqus.cfg” on Linux or OSX, (2) “foqus.cfg” in the working directory.

The Settings screen displays settings grouped into tabs. Figure 5 shows Settings, FOCUS tab.

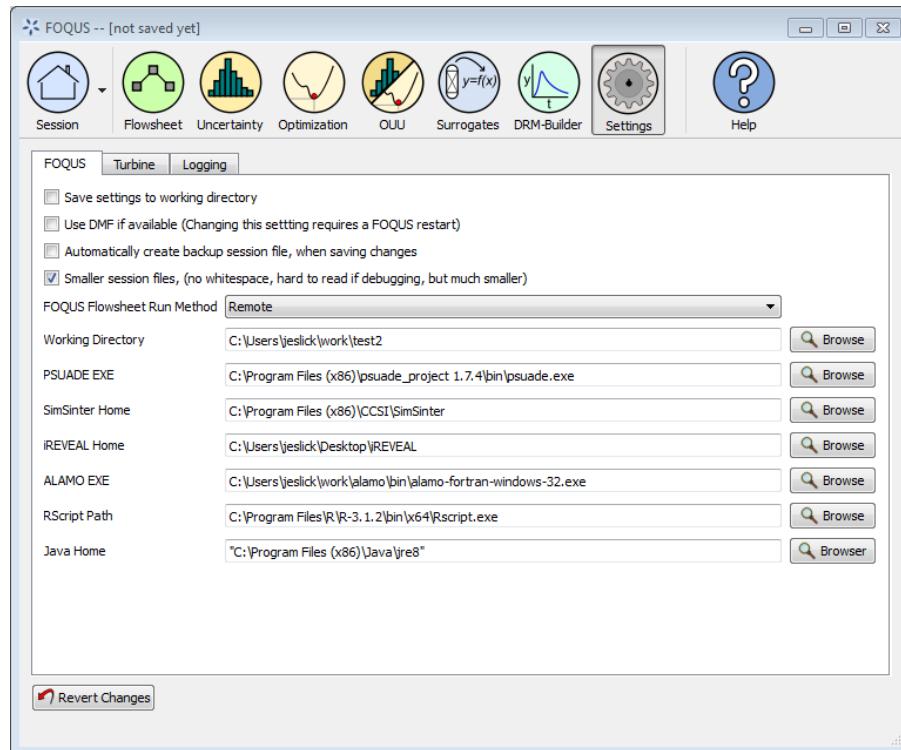


Figure 5: Settings, FOCUS Tab

Options in the Settings, FOCUS tab are described below.

1. **Save settings to working directoy**, when checkbox is selected the settings file will be read from the specified working directory. This setting is useful for running multiple copies of FOCUS to ensure the settings do not conflict. When starting additional copies of FOCUS, it is best to start them from the Working Directory command line giving each copy of FOCUS its own independent working directory. If FOCUS is started more than once from the Windows start menu, each copy will use the same working directory. Starting FOCUS multiple times with the same working directory may cause unusual behavior in FOCUS.
2. **Use DMF if available**, when checkbox is selected the Data Management Framework (DMF) module will be loaded and the DMF options will be shown in the **Session** menu.
3. **Automatically create backup session file**, when checkbox is selected each time a FOCUS session is saved it will be saved twice. A backup copy with a universally unique identifier appended to the file name will be saved. This will allow the user to load any previous save point of the session.

4. **Smaller session files**, when checkbox is selected significant storage space is saved by excluding formatting from the session file; this makes the session files less human readable. A more readable session file can be useful for debugging.
5. **FOQUS Flowsheet Run Method** enables the user to select between running simulations on the same computer as FOQUS, or on a remote Turbine gateway. Running simulations remotely allows parallel execution.
6. **Working Directory** is the path to the FOQUS working directory. The **Working Directory** is where FOQUS reads and writes files needed to function. When running multiple copies of FOQUS, the **Working Directory** can also be specified from the command line using the “-w” or “–workingDir” options. After changing the **Working Directory**, FOQUS should be restarted.
7. **PSUADE EXE** is the path to the PSUADE executable. PSUADE provides FOQUS’s UQ features.
8. **SimSinter Home** is the path to the SimSinter interface for creating Sinter configuration files for simulations to be run with FOQUS. This setting is not required but it allows easy access to the SimSinter configuration GUI when uploading simulation to Turbine.
9. **iREVEAL Home** is the path the iREVEAL installation. This is required to use the iREVEAL surrogate model module.
10. **ALAMO EXE** is the path to the ALAMO executable. This is required to use the ALAMO surrogate model module.
11. **RScript Path** is the path to the RScript executable. This is required for surrogate model modules that use R as a platform.
12. **Java Home** is the path to the Java installation. The DMF and the iREVEAL surrogate modules require Java.
13. **Revert Changes** The settings changes are applied when the user navigates away from the settings screen. To undo changes made to settings the revert button can be clicked before changing to another screen.

The **Turbine** tab contains settings for configuring the local and remote instance of Turbine. Figure 6 shows the FOQUS Turbine settings.

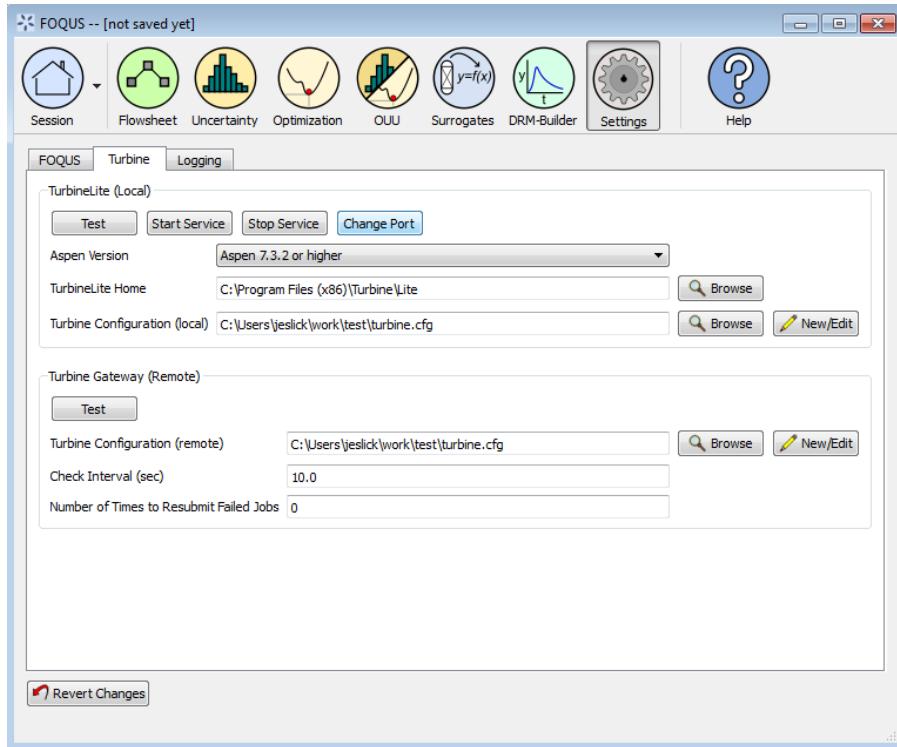


Figure 6: Settings, Turbine Tab

The first section in the **Turbine** tab is **TurbineLite (Local)**. This section contains settings related to the local installation of Turbine, and is only applicable when running FOCUS on the windows platform.

1. **Test** tests the connection to the local Turbine server to make sure it is configured and running properly.
2. **Start Service** starts the Turbine server service on Windows. The user must have permission to start services to use this button.
3. **Stop Service** stops the Turbine server service on Windows. The user must have permission to stop services to use this button.
4. **Change Port** can reconfigure the local Turbine server service on Windows to use a different port. This may be necessary if Turbine conflicts with another service.
5. **Aspen Version**, Aspen 7.3 is still in common use but the API differs slightly from newer versions. This option allows FOCUS to be used with Aspen 7.3.
6. **TurbineLite Home** is the location of the TurbineLite installation. For local simulation runs FOCUS needs to know where TurbineLite is installed so it can launch Turbine consumers to run simulations. This setting is not needed if simulations are only run remotely.
7. **Turbine Configuration (local)** is the path to the TurbineLite gateway configuration file for running simulations locally. If simulations are only run remotely, this setting is not needed.

New/Edit displays a form to create or edit a Turbine configuration file. Having a setting for both local and remote Turbine allows easy switching between run methods.

The second section in the **Turbine** tab is **Turbine Gateway (Remote)**. This section contains settings related to a remote instance of Turbine.

1. **Test** tests the connection to the remote Turbine server to make sure it is configured and running properly.
2. **Turbine Configuration (remote)**, is the path to the Turbine gateway configuration file for running simulations remotely. If simulations are only run locally, this setting is not needed. **New/Edit** displays a form to create or edit a Turbine configuration file. Having a setting for both local and remote Turbine allows easy switching between run methods.
3. **Check Interval (sec)** is the number of seconds between checking the remote Turbine server for job results. This number should not be set too low to avoid overwhelming the Turbine server with requests.
4. **Number of Times to Resubmit Failed Jobs** is the number of times to resubmit jobs that fail. Jobs occasionally fail due to software bugs. This allows a job to be retried.

The **Logging** tab contains settings related to the FOCUS log files, which provide debugging information. The FOCUS log files are stored in the logs directory in the working directory. Figure 7 show the FOCUS log settings. There are two log files (1) FOCUS and (2) Turbine Client.

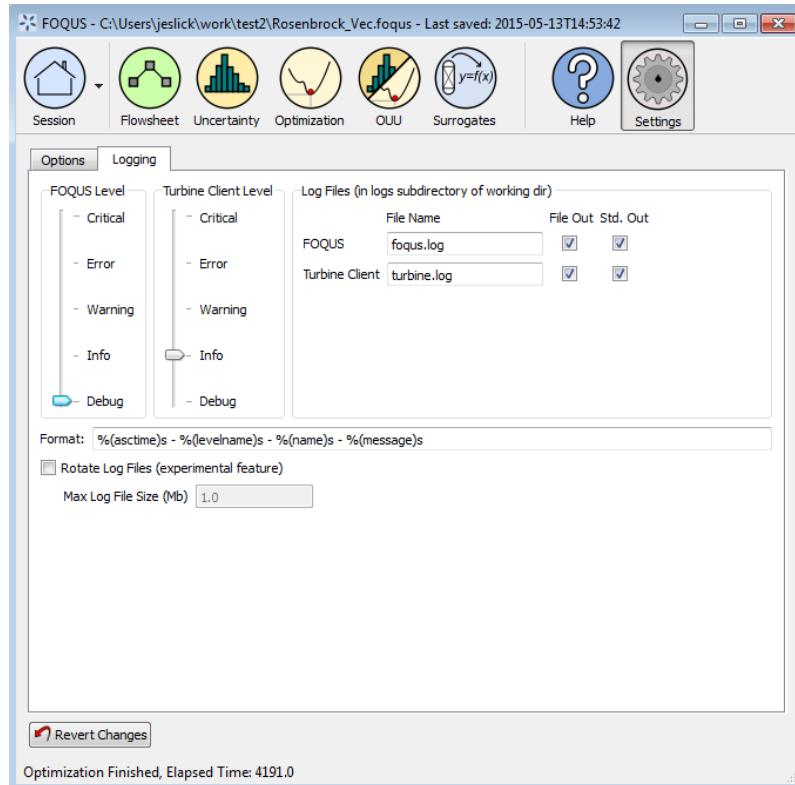


Figure 7: Settings, Logging Tab

1. The level sliders indicate how much information to send to the logs.
2. The **Log Files** section enables the user to specify where the log information is sent. The **File Out** checkboxes turn on or off the file output of logs. The **Std. Out** checkboxes enable or disable the output to the screen.
3. **Format** allows the format of the log messages to be changed. See the documentation for the Python 2.7 logging module for more information.
4. **Rotate Log Files** turns on or off log file rotation. When a log file reaches a certain size, a new log file is started and the contents of the old log are moved to a new file. There currently seems to be a bug in the log file rotation which occasionally makes the log file output stop; therefore, the **Rotate Log Files** option is labeled as an experimental feature.

Flowsheet

The meta-flowsheet defines connections between simulations. The flowsheet defines the order that simulations are performed and what data is transferred between them. Simulations are represented as nodes in the flowsheet. These simulations may be links to external simulation software through the Turbine gateway, or custom simulations or simulation wrappers written in Python. Directed edges in the flowsheet connect nodes. The edges also specify which variables in the simulations are equivalent.

If the flowsheet contains cycles, they are solved iteratively. Tear streams are selected by FOCUS based on two criteria: (1) minimize the maximum number of times any cycle is torn and (2) minimize the total number of tear edges (which only is considered when two tear sets have the same value for the first criteria).

FOCUS currently has two methods available for solving flowsheets with recycle: (1) direct substitution and (2) Wegstien [6]. FOCUS will solve strongly connected components in the order they are encountered in the flowsheet. FOCUS flowsheets are generally not very complicated, so if a strongly connected component contains more than one tear stream, they are solved simultaneously. More advanced solution options will be added if a need arises. Figure 8 shows how a simple flowsheet with recycle would be solved.

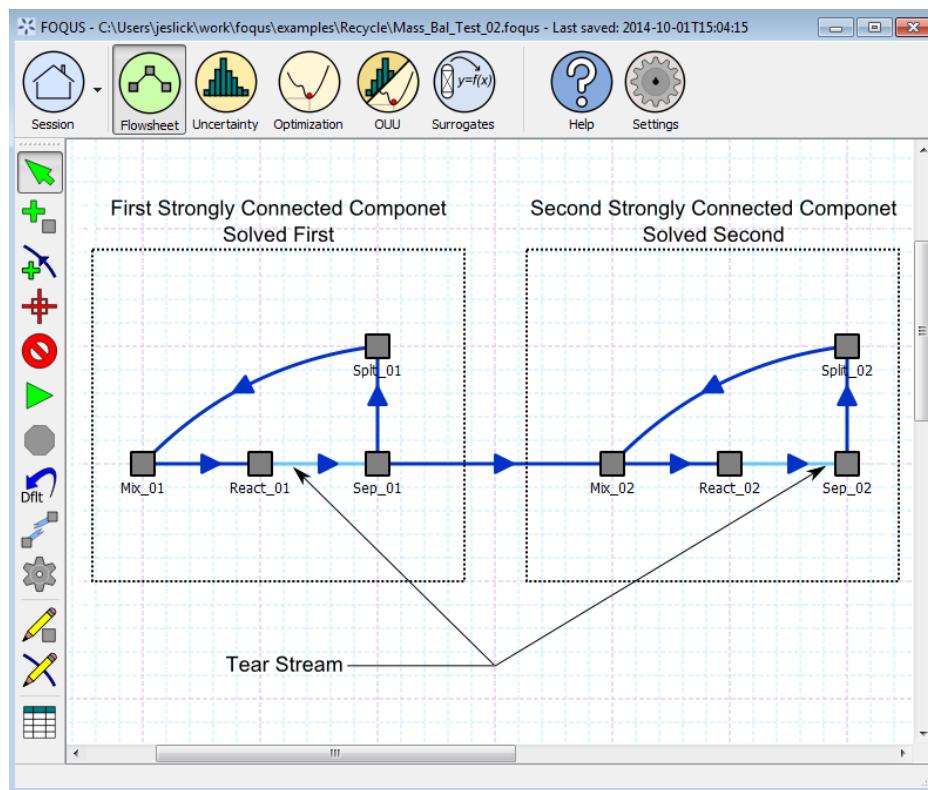
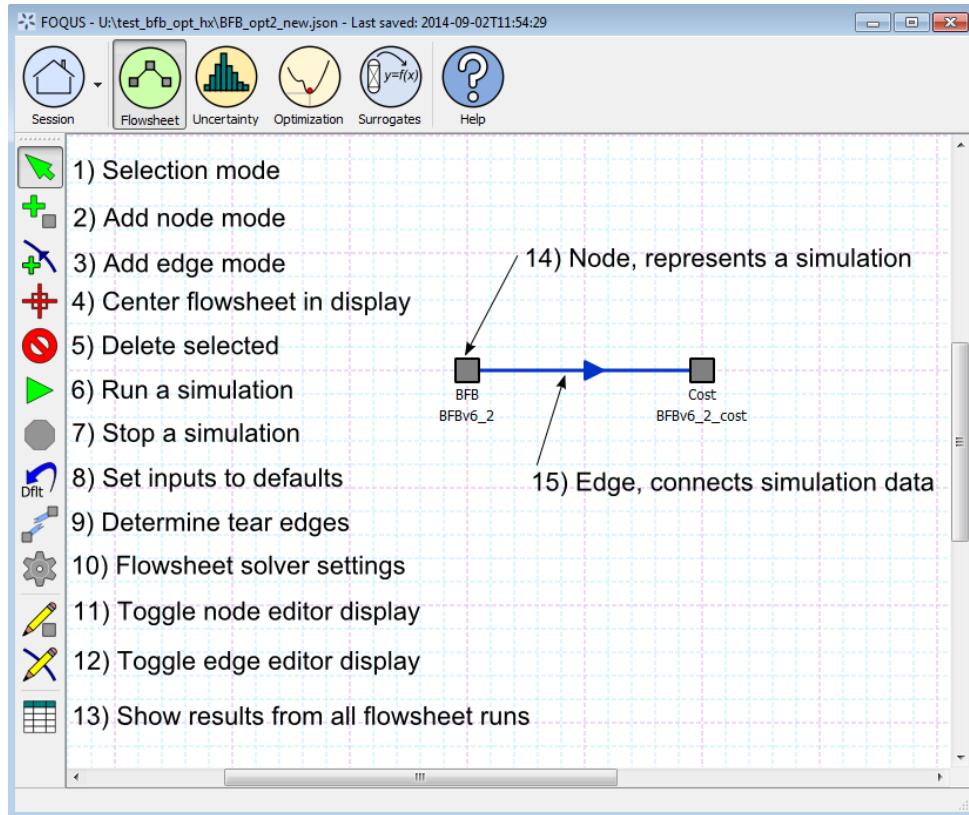


Figure 8: Flowsheet Recycle

Flowsheet Editor

Figure 9 illustrates the main **Flowsheet Editor** screen and a description of the pieces. The toolbar on the left contains various flowsheet tools.

**Figure 9: Flowsheet Editor**

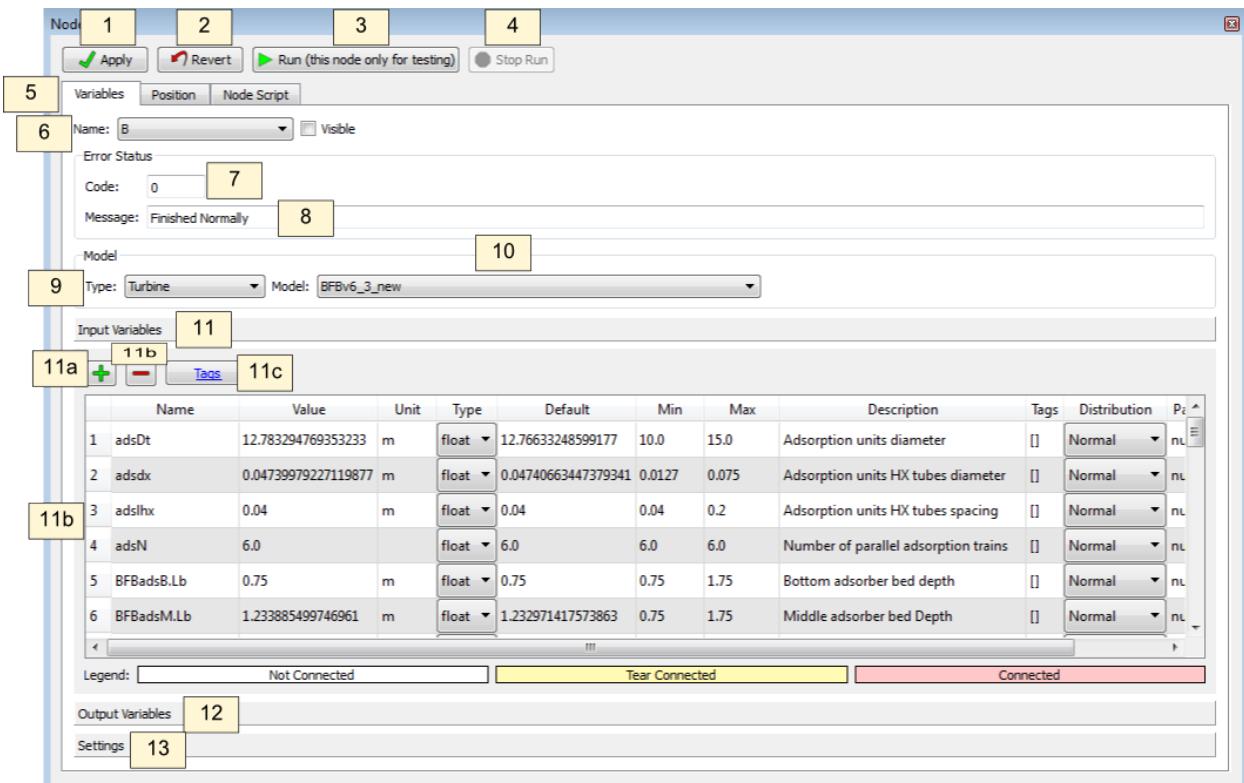
The first three buttons are mouse mode buttons. The current mouse mode is shown by the depressed button. The remaining buttons on the toolbar perform an action. The flowsheet editing toolbar and flowsheet are described in detail below.

1. **Selection mode** enables the user to select nodes and edges. Multiple items may be selected by holding down the Shift key. To deselect everything, click an empty area of the flowsheet while not holding the Shift key. Selected items can be moved by dragging them. To move multiple items, hold down the Shift key while dragging. The last item selected becomes the current object to be edited in the **Node** or **Edge Editor**.
2. **Add node mode** enables the user to add a node by clicking anywhere on the flowsheet. Once a location is clicked, a dialog box opens where the new node name can be entered. If **Cancel** is selected, no node is added. The new node name cannot be “graph” and cannot match any existing node name.
3. **Add edge mode** enables edges to be added by selecting the node that the edge originates from, followed by the node the edge terminates at.
4. **Center flowsheet in display** centers the display on the flowsheet.
5. **Delete selected** deletes all selected nodes and edges. If a node is deleted, all edges connecting to that node are also deleted.

6. **Run a simulation** starts a single simulation run. This is primarily used to test a simulation before running optimization or UQ.
7. **Stop a simulation** is enabled when a simulation is running and stops any running simulation. The simulation may take several seconds to stop.
8. **Set inputs to defaults** returns all of the inputs to their default values.
9. **Determine tear edges** makes it easier to see where initial guesses are needed and makes it possible to edit the tear set before running the flowsheet. If tear streams are needed but not specified before running a flowsheet, they will be automatically specified, however inputs that will be used for the initial guess will not be known before running.
10. **Flowsheet solver settings** contains options related to tear solvers.
11. **Toggle node editor display** displays or hides the **Node Editor**. The user can change the node being edited by selecting from **Name** in the **Node Editor** or selecting it on the flowsheet in selection mode.
12. **Toggle edge editor display** displays or hides the edge editor. The user can change the edge being edited in the **Edge Editor**, or by selecting it in selection mode.
13. **Show results from all flowsheet runs** displays the results of all flowsheet runs in a table view. This can be exported to a spreadsheet.
14. **Node** represents a simulation or calculations.
15. **Edge** connects simulation data, represents data transfer between two nodes.

Node Editor

The **Node Editor** enables the assignment of simulations to a node, and editing variables. Figure 10 shows the Node Editor window with the input variables section of the toolbox displayed.

**Figure 10: Node Editor Window**

1. **Apply** immediately applies any changes made in the **Node Editor**. This is not usually needed. Changes are applied when the current node is changed, the **Node Editor** is closed, or some other action is taken that requires the flowsheet, such as running the flowsheet.
2. **Revert** sets the node back to the version where the changes were last applied. This is usually the original state of the node when the editor was opened.
3. **Run** can be used to run the simulation represented by this node only. This can be used for testing to make sure the node is properly configured without running the whole flowsheet.
4. **Stop Run** is active when a simulation is currently running. It stops a single node run or a flowsheet run.
5. There are three tabs in the **Node Editor**: (1) **Variables** tab, shown in Figure 10, (2) **Position** tab displays the coordinates of the node, and (3) **Node Script** tab enabling the entry of Python code to be executed after the simulation is run.
6. **Name** displays the name of the node currently being edited. The current node can be changed by selecting from existing nodes in the drop-down menu.
7. **Code** displays the error status code for the node.
8. **Message** displays a more detailed description of the error status of the node.
9. **Type** enables the user to select the type of model to run. The model types are none, Turbine, DMF Lite, DMF Server, or Python Plugin. None allows no model to be assigned to the node;

this is useful when the node only executes a script entered directly into FOQUS. Turbine is used to execute Aspen, gPROMS, or Excel simulations. If simulations are stored in either the DMF lite or DMF server, the DMF type models can be used. FOQUS will automatically upload DMF models to Turbine as needed. Python plugins are custom simulations or wrappers written by the user. Surrogate model methods may also produce Python plugin models.

10. **Model** enables selection of the models available on Turbine or loaded Python plugins.
11. **Input Variables** enables viewing and editing the node's input variables. Most of these variables are added automatically when a simulation is selected.
 - (a) **Add variable** enables the addition of an input variable. There are two reasons to add an input: (1) to use a variable to pass information to another simulation (even if the variable is not used in any node calculation, it can receive data from the previous simulation and be passed on to the next simulation) and (2) to use in a node script. For example, a variable could be added that provides output in different units of measure.
 - (b) **Remove variable** removes variables. If an input variable is removed that originally came from a Turbine simulation, the simulation will run with the default value.
 - (c) **Tags** displays a tag browser that lists commonly used variable tags.
 - (d) **Input Variables** table displays information about variables. Most attributes can be edited, except for the **Name** column within the **Input Variables** table. The rows for input variables are color coded depending on whether they are set by an edge from results in another node. White rows are not connected. Yellow rows are set by a tear edge. These variables serve as initial guesses but their value may change once the simulation has run. Red rows are set by an edge that is not a tear edge. The value set for these inputs does not matter and it may change once the simulation has run.
12. **Output Variables** is a variable table similar to the **Input Variables** table for node output variables. This area is displayed by clicking **Output Variables**.
13. **Settings** displays simulation settings. A description is provided for each setting. The available settings vary depending on simulation.

Node Variables

Variables in the node editor are grouped into two sections, inputs and outputs. The input and output variable tables are accessible as described in the previous section. The contents of the variable tables are described here.

The columns in the input variable list are:

- **Name** is the name of the variable,
- **Value** is the current value,
- **Unit** is the unit of measure,
- **Type** is the data type (float, int, or str),

- **Default** is the default value,
- **Min** is the minimum value,
- **Max** is the maximum value,
- **Description** is a description string,
- **Tag** is a list of strings that can be used to attach additional information to a variable
- **Distribution** is a distribution type,
- **Param1** is the first parameter of a parametric distribution the exact meaning depends on the selected distribution, and
- **Param2** is the second parameter of a parametric distribution the exact meaning depends on the selected distribution.

The minimum and maximum values for are not enforced when running simulations are not enforced. A value can be given outside the range. Optimization and UQ features make use of these values to set upper and lower bounds on decision variables or sampling. The distribution information is used when setting up sampling for UQ. In the future, this may also be used for things like optimization under uncertainty. Integer and string type variables cannot currently be used as optimization decision variables, or sampled with the UQ tool.

The rows of the input variable table are color coded. Some of the input variables may be set by connections to other nodes. White rows are variables who's values are not set by a connection. The variables that are red have values set by a connection, and the value given will be overwritten and does not matter. The values that are colored yellow are inputs set by a connection that is a tear stream. The values of these variables serves as an initial guess for solving recycles.

The output variable table is similar to the input table, however it only contains the columns: Name, Value, Unit, Type, Description, and Tags. The value of the outputs may not correspond to the inputs until the simulation has been run.

Node Script

There are three type of **Node Script** that can be used: (1) **Pre** runs before a node simulation, (2) **Post** runs after a node simulation, and (3) **Total** scripts how a node runs the simulation.

Figure 11 illustrates the **Node Script** tab of the **Node Editor** with calculations for an optimization test problem.

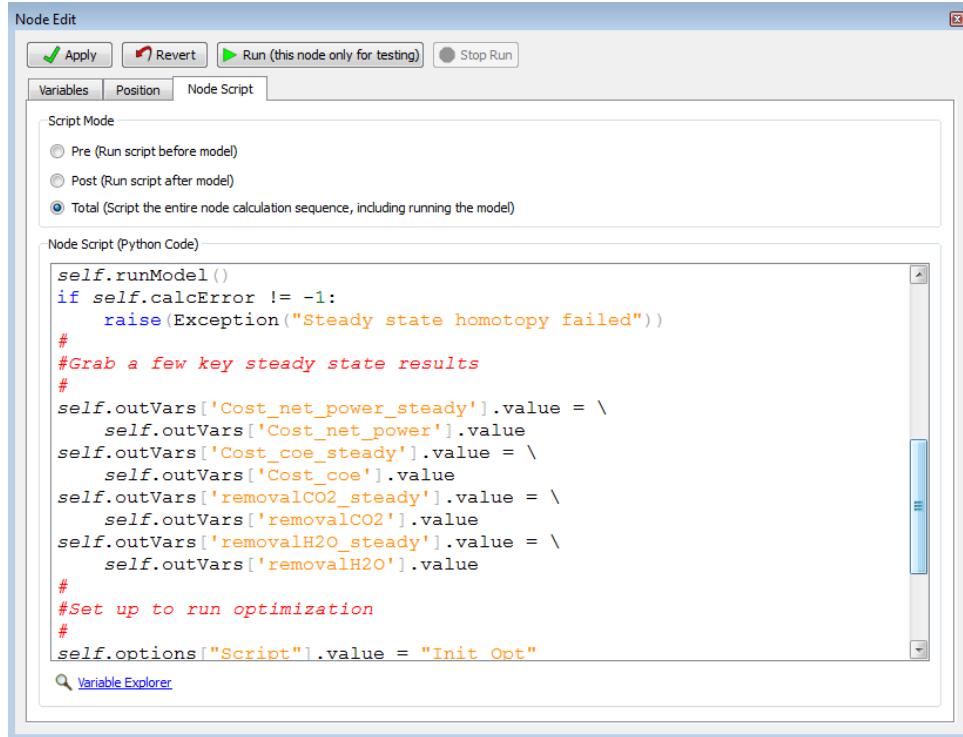
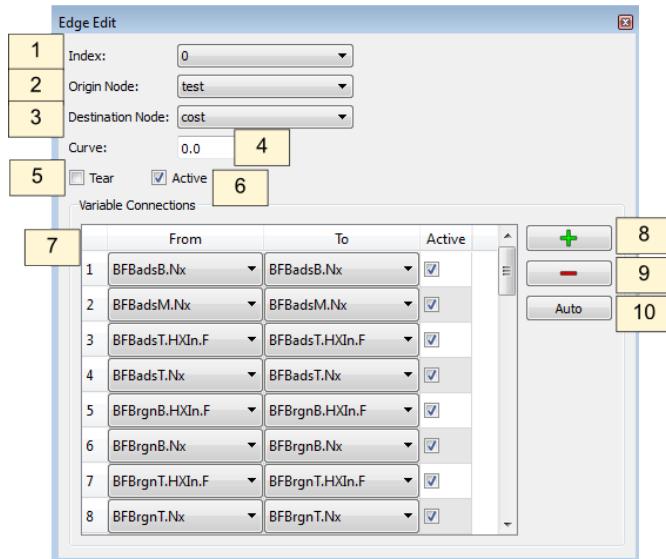


Figure 11: Node Script Tab

Node scripts can be any valid Python code. The input and output variables for node scripts are stored in dictionaries x and f . The dictionary keys are the variable names. The f dictionary is used to update the node variables after the calculations are executed.

Edge Editor

The [Edge Editor](#) is illustrated in Figure 12. The [Edge Editor](#) can be used to set connections between node variables.

**Figure 12: Edge Editor**

1. **Index** is the index of the current edge. The current edge can be changed by selecting an index from the drop-down menu, but since the index is not a very meaningful identifier it is usually more convenient to select the edge to edit with the graphical selection tool.
2. **Origin Node** is the node where an edge starts. This may be edited by selecting a different node from the drop-down menu.
3. **Destination Node** is the node to which the edge goes.
4. **Curve** can be a positive or negative number. The greater the magnitude of number, the more curved an edge will appear in the flowsheet. This setting is used to keep edges from overlapping in the flowsheet display.
5. **Tear** marks this edge as a tear. Before a simulation is run, if a valid tear set is not specified, FOQUUS locates one.
6. **Active** specifies whether the edge is active or not. This allows connections to be temporarily disabled.
7. **Variable Connections** table displays which variables are connected. Inputs or outputs in the origin node can be connected to inputs in the destination node.
8. **Add connection** adds a new connection.
9. **Remove connection** deletes the selected connections.
10. **Auto** automatically connects variables having the same name. For example, in connecting a simulation to a spreadsheet to calculate costs there are a large number of variables for which it makes sense that the variables have the same name in the simulation and spreadsheet. **Auto** should be used with great care. Connecting variables with the same name is often not what is wanted. For example two simulations may have a variable named FlowAIn; however, it is very unlikely that they should be connected. It is more likely FlowAOut should be connected to FlowAIn.

Sample Results

Flowsheet evaluations that have been run in a FOCUS session can be viewed by clicking the table button in the flowsheet toolbar (#13 in Figure 9). The results are displayed in a table, and the contents can be copied and pasted into a spreadsheet or exported to a CSV file. Figure 13 show the Flowsheet Results Table window.

The screenshot shows a Windows dialog box titled "Dialog" with the sub-tittle "Flowsheet Results". The table has 13 rows and 6 columns. The columns are labeled: SetName, ResultName, Error, Input.Node1.x1, Input.Node1.x2, and Output.Node1.z. The first three rows (0, 1, 2) are for "Single_runs" and the next 12 rows (3-14) are for "UQ_Engsemble". The table is highlighted with alternating green and white rows. Callouts are numbered 1 through 4: 1 points to the "Menu" button; 2 points to the "Current Filter" dropdown; 3 points to the "Edit Filters" button; 4 points to the left margin of the table area.

SetName	ResultName	Error	Input.Node1.x1	Input.Node1.x2	Output.Node1.z
0	"Single_runs"	"single_0"	0	0.0	0.0
1	"Single_runs"	"single_0"	0	0.0	0.0
2	"Single_runs"	"single_0"	1	0.0	-1.0
3	"Single_runs"	"single_0"	0	0.0	2.0
4	"UQ_Engsemble"	"uq_000000"	0	-0.10214362857031789	1.7843292898891165
5	"UQ_Engsemble"	"uq_000001"	0	0.46147381209837013	0.47766637086293956
6	"UQ_Engsemble"	"uq_000002"	1	-0.8966375863629568	-0.4323514546185505
7	"UQ_Engsemble"	"uq_000003"	0	0.12637509141414194	1.183360565609001
8	"UQ_Engsemble"	"uq_000004"	0	-0.5825992638164195	0.9085407895774307
9	"UQ_Engsemble"	"uq_000005"	0	0.8163724582718559	1.378861139425478
10	"UQ_Engsemble"	"uq_000006"	0	-1.8597680340799354	0.6833428176508021
11	"UQ_Engsemble"	"uq_000007"	0	1.5562567798217093	0.5051505042496838
12	"UQ_Engsemble"	"uq_000008"	1	1.0133916221621408	-0.17108132835527934

Figure 13: Flowsheet Results Table Window

1. **Menu** contains a menu with four sub menus.
 - (a) **Import** data from files or the clipboard.
 - (b) **Export** data to files or the clipboard.
 - (c) **Edit** or delete data.
 - (d) **View** options for the table.
2. The **Current Filter** drop-down list enables the user to select a data filter, which can be used to filter and sort data.
3. **Edit Filters** enables the user to create or edit data filters.

Error Codes

Error codes are listed in the **Flowsheet Results** table for the whole flowsheet and for individual nodes. Table 1 shows the flowsheet error codes and Table 2 shows the node error codes. The most common flowsheet error is 1, a node calculation failed. The most common node error is 7, Turbine simulation error. These errors are typically caused by a simulation that fails to converge or

has some other calculation error (e.g., ACM does not converge or an Excel spreadsheet simulation with a division by 0 error).

Table 1: Flowsheet Error Codes

Code	Meaning
-1	Did not run or finish
0	Success
1	A node calculation failed
3	Failed to create a worker node
5	Unknown tear solver
11	Wegstein failed to converge
12	Direct failed to converge
16	Presolve node error
17	Postsolve node error
19	Unhandled exception during evaluation (see log)
20	Flowsheet thread terminated
21	Missing session name
40	Error connecting to Turbine
50	Error loading session or inputs
100	Single node calculation success
201	Cycle in determining calculation order (invalid tear set)

Table 2: Node Error Codes

Code	Meaning
-1	Did not run or finish
0	Success
1	Simulation error (see log)
3	Exceeded maximum wait time
4	Failed to create Turbine session ID
5	Failed to add Turbine job
6	Exceeded maximum run time
7	Turbine simulation error
8	Failed to start Turbine job
10	Failed to get Turbine jobs status
11	Flowsheet thread terminated
20	Error in node script
23	Could not convert Numpy value to list
27	Cannot read variable result (see log)

Tutorial – Example Files

Many of the tutorials in this manual make use of example files provided in the FOCUS installation. This tutorial helps the user find and copy the example files.

1. Locate the example files; they are in the FOCUS installation directory. If the Windows installer was used, the files can usually be found in one of two locations:
 - (a) 64-bit OS, C:\Program Files (x86)\foqus\foqus\examples
 - (b) 32-bit OS, C:\Program Files\foqus\foqus\examples
2. Copy the example files directory to a convenient location for use in other tutorials.

Tutorial – Creating a Flowsheet

This tutorial provides information about the basic use of FOCUS and setting up a very simple flowsheet. A single node flowsheet will be created that performs a simple calculation using a square root so that simulation errors can be observed when a negative input value is provided.

1. Start FOCUS (see Section 2.1).
2. In the session form enter the **Session Name** as “Simple_Flow” (Figure 14).

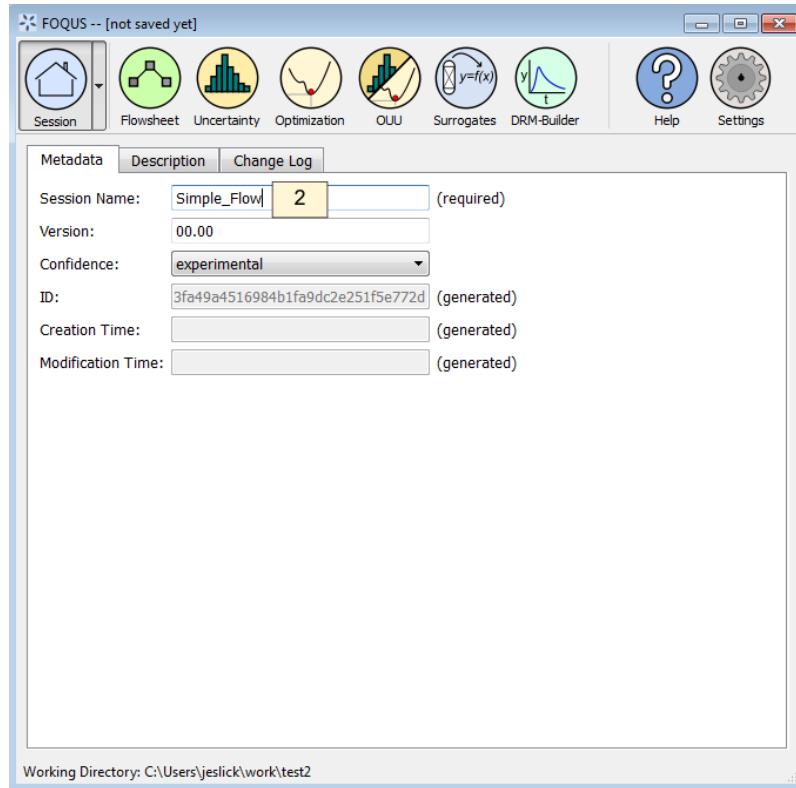


Figure 14: Setting the Session Name

3. Set the session description.
 - (a) Select the **Description** tab (Figure 15).
 - (b) Type the description shown in Figure 15. The buttons above the **Description** tab box can be used to format the text.

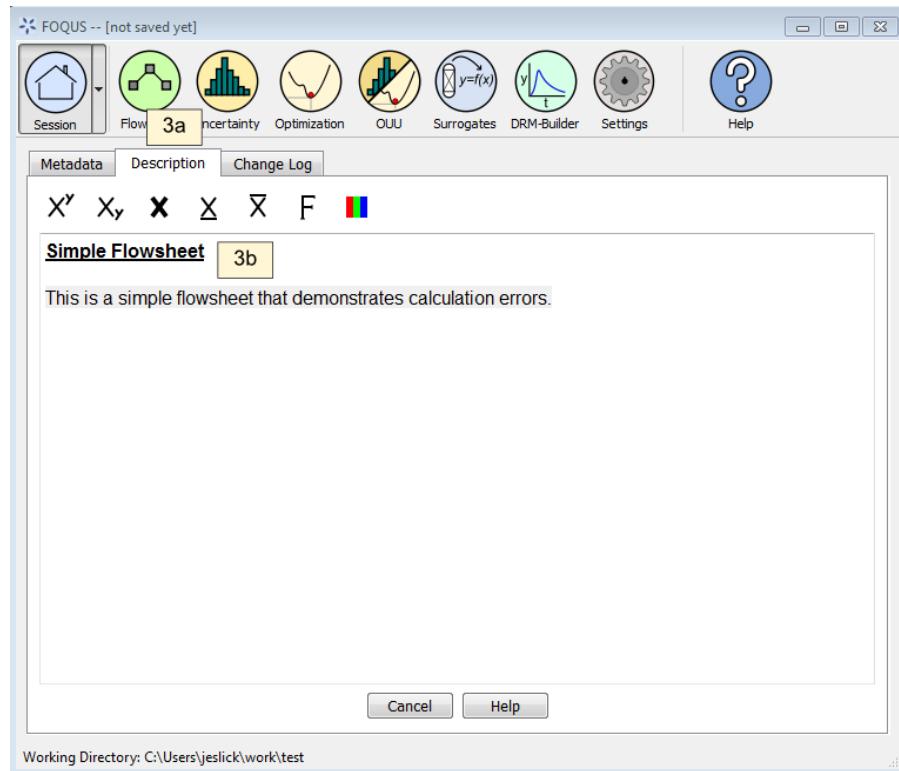


Figure 15: Setting the Session Description

4. Click the **Flowsheet** button at the top of the Home window (Figure 16).
5. Add a node named “calc.”
 - (a) Click the **Add Node** button in the toolbar on the left side of the Home window.
 - (b) Click a location on the gridded flowsheet area.
 - (c) Enter the node name “calc” in the dialog box.
6. Click the **Select Mode** button in the toolbar.
7. Open the Node Editor by clicking the **Node Editor** button in the toolbar.
8. Add input variables to the node. (When linking a node to an external simulation the input and output variables are populated automatically, and this step is not necessary.)
 - (a) Click **+** above the **Input Variables** table.
 - (b) Enter x1 in the variable **Name** dialog box.
 - (c) Click **+** above the **Input Variables** table.
 - (d) Enter x2 in the variable **Name** dialog box.
 - (e) Enter -2 and 2 for the **Min** and **Max** of x1 in the **Input Variables** table.
 - (f) Enter -1 and 4 for the **Min** and **Max** of x2 in the **Input Variables** table.
 - (g) Enter 1 for the value of x1.
 - (h) Enter 4 for the value of x2.

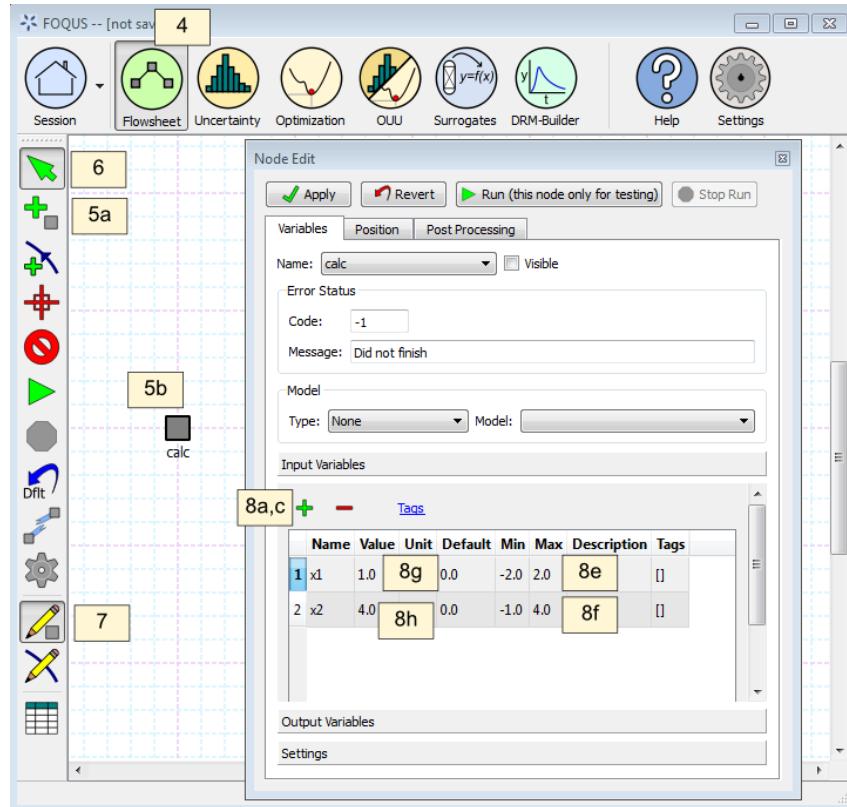


Figure 16: Flowsheet, Input Variables

9. Add an output variable to the node. (When linking a node to an external simulation the input and output variables are populated automatically.)
 - (a) Click **Output Variables** to show the **Output Variables** table (Figure 17).
 - (b) Click **+** above the **Output Variables** table to add a variable.
 - (c) Enter z in the output **Name** dialog box.

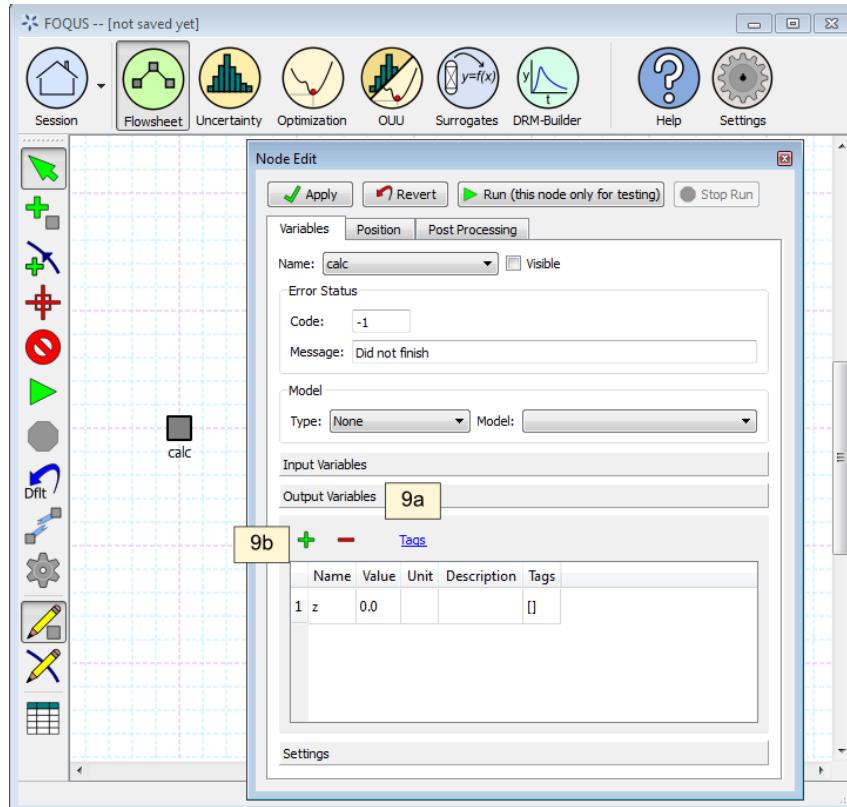


Figure 17: Flowsheet, Output Variables

In this example, the node is not linked to any external simulation. The FOCUS nodes contain a section called node script, which can be used to do calculations before, after or instead of a simulation linked to the node. The node script can be used for things such as unit conversion, simple calculations, or simulation convergence procedures. The node scripts are written as Python. The **Input Variables** are contained in a dictionary named `x` and the **Output Variables** are contained in a dictionary named `f`. The dictionary keys are the variables names shown in the input and output tables. Only **Output Variables** can be modified by a node script.

10. Add a calculation to the node.
 - (a) Click the **Node Script** tab (Figure 18).
 - (b) Enter the following code into the Python code box:

$$f['z'] = x['x1'] * \text{math.sqrt}(x['x2'])$$
11. Click the **Variables** tab.
12. Click the **Run** button (Figure 18).

The flowsheet should run successfully and the output value should be 2. Rerun the flowsheet with a negative value for x_2 , and observe the result. The simulation should report an error.

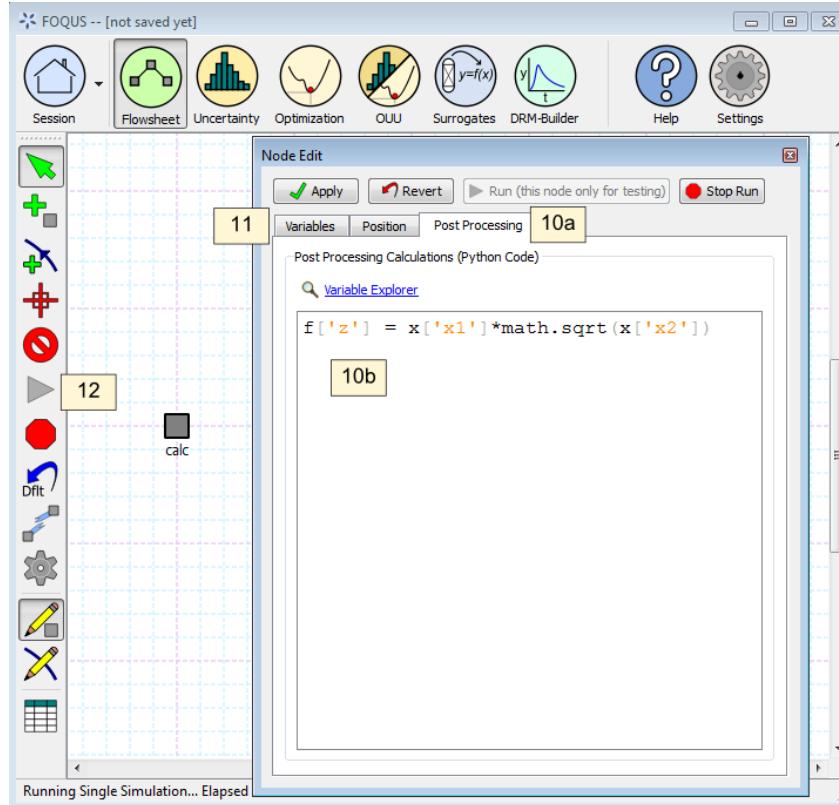


Figure 18: Node Calculation

10. Save the FOCUS session.

- Click the **Session** drop-down menu at the top of the Home window (Figure 19).
- Click **Save**. The exact location of save in the menu depends on whether or not the data management framework is enabled.
- The **Change Log** entry can be left blank.
- The default file name is the session name. Change the file name and location if desired.

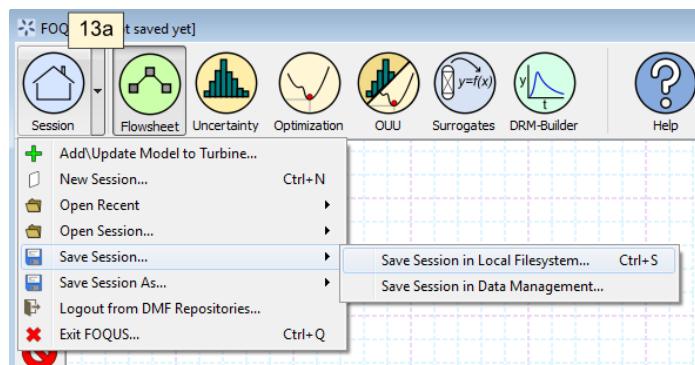


Figure 19: Save Session

Tutorial – Creating a Flowsheet with Linked Simulations

This tutorial is referenced by other tutorials. **Save the flowsheet in a convenient location for future use.**

This tutorial demonstrates how to link simulations to nodes, and how to connect nodes in a flowsheet. Two models are used: (1) a bubbling fluidized bed model in ACM and (2) a cost of electricity (COE) model in Excel. The COE model estimates the cost of electricity for a 650 MW (net before adding capture) supercritical pulverized coal power plant with solid sorbent post combustion CO₂ capture process added.

Before starting the tutorial, see Section 2.6 to locate and copy the example files to a convenient location.

1. Start FOCUS. The Session window displays (Figure 20).
2. Enter “BFB_opt” in **Session Name** (without quotes).
3. Click the **Description** tab. The problem description box displays and is shown in (Figure 21).
4. In the problem description box enter information about the problem being solved in the FOCUS session; this information can be more extensive than what is shown in the example.
5. Save the session file. Click **Save Session** from the **Session** drop-down menu. Enter change log information and a file name when prompted. The **Creation Time** in metadata page will be the time the session is first saved. The **Modification Time** will be the last time the session was saved. The **ID** is a unique identifier that changes each time the user saves the simulation. The **Change Log** tab provides a record of the changes made each time the session is saved.

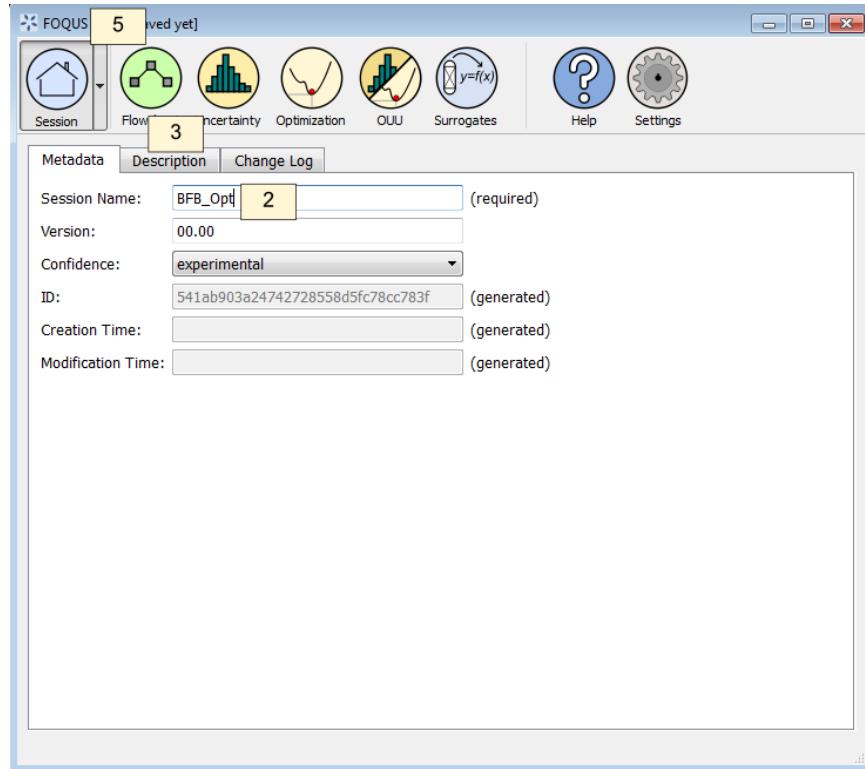


Figure 20: Session Setup

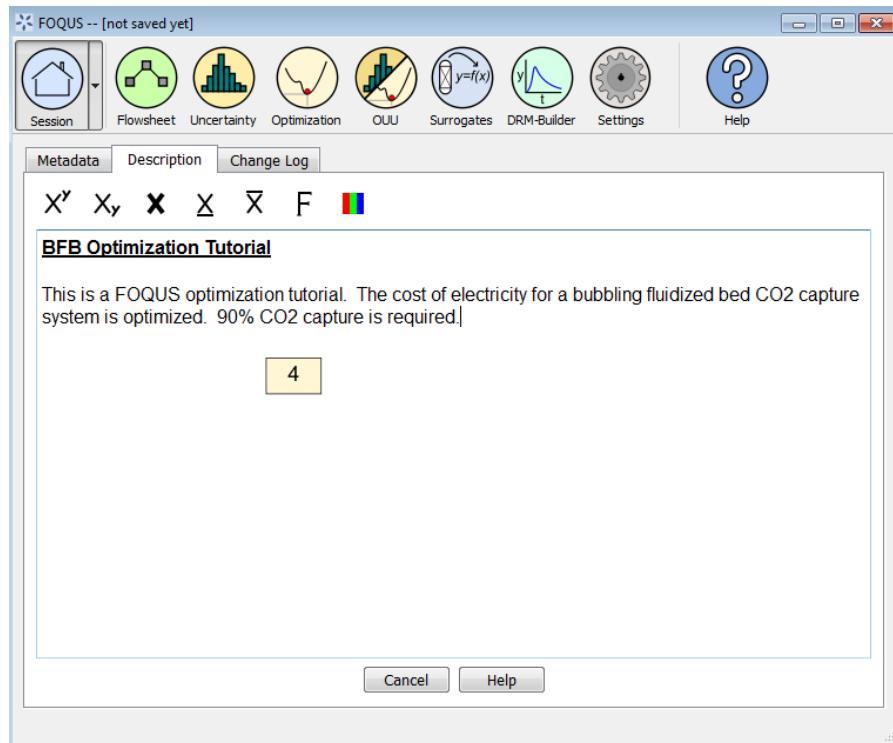


Figure 21: Session Description

There are two models needed for this optimization problem: (1) the ACM model for the BFB capture system and (2) the Excel cost estimating spreadsheet. These models are provided in the example files directory, under optimization/models (see Section 2.6). There are two SimSinter configuration files: (1) BFB_sinter_config_v6.2.json for the process model and (2) BFB_cost_v6.2.3.json for the cost model. The next step is to upload the models to Turbine.

6. Open the Add\Update Model to Turbine dialog box (Figure 22).
7. In this case, the SimSinter configuration files have already been created. If a SimSinter configuration file needs to be created for the simulation, Create/Edit displays the SimSinter configuration GUI (see Figure 23). See the SimSinter documentation or Chapter 9 for more information.
8. Click Browse to select a SimSinter configuration file (Figure 23). Once the SimSinter configuration file is selected, the simulation file and sinterconfig file is automatically added to the files to upload. The application type is entered automatically. If there are additional files required for the simulation, those files can be added by clicking Add File.
9. Enter the simulation name in Simulation Name. This name is determined by the user, but will default to the SimSinter configuration file name. For this tutorial use BFB_v6_2.
10. Click OK to upload the simulation.
11. Repeat the upload process for the cost model. Name the model BFB_v6_2_Cost.

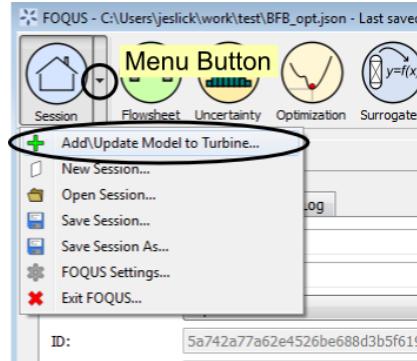


Figure 22: Open Upload to Turbine Dialog

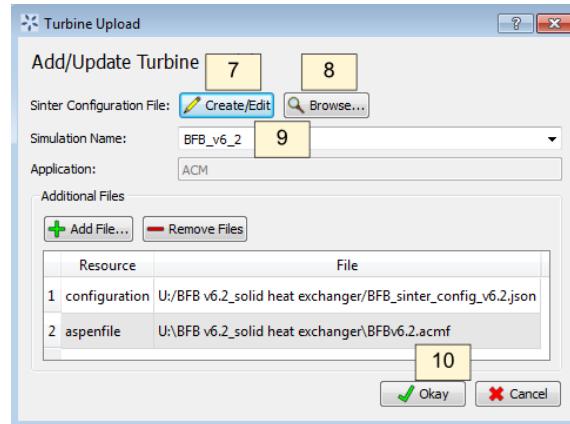


Figure 23: Upload to Turbine Dialog

The next step is to create the flowsheet. Figure 24 illustrates the steps to draw the flowsheet.

12. Click **Flowsheet** at the top of the Home window.
13. Click **Add Node mode**.
14. Add two nodes to the flowsheet. Name the first node “BFB” and the second node “cost”.
15. Click **Add Edge mode**.
16. Click the BFB node followed by the cost node.
17. Click **Selection mode** and select the BFB node.
18. Click **Toggle Node Editor**. The Node Editor displays as illustrated in Figure 25.

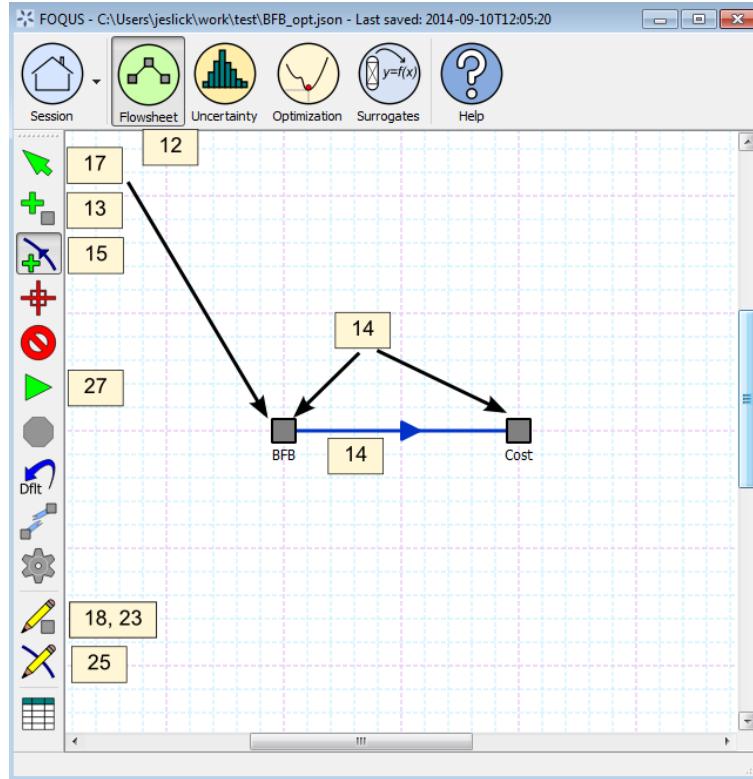
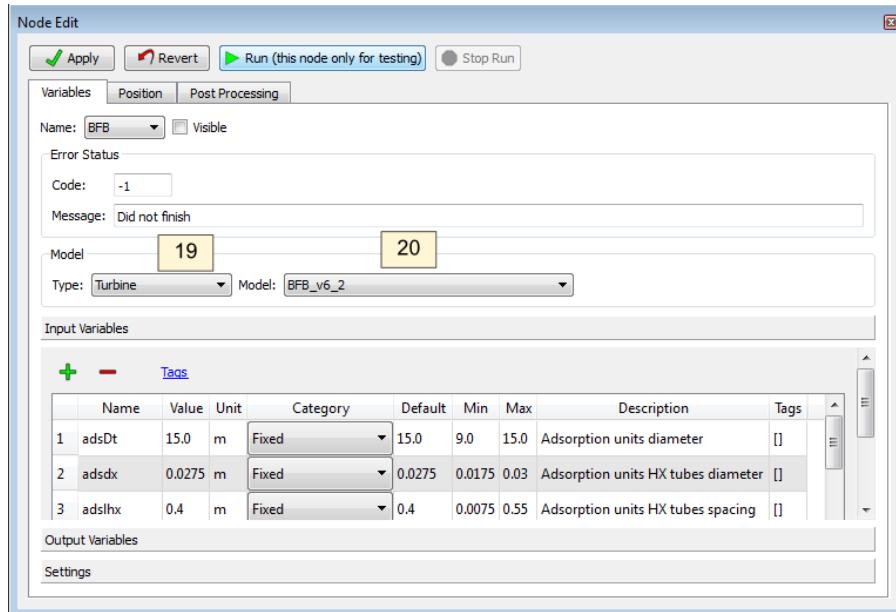


Figure 24: Flowsheet Editor

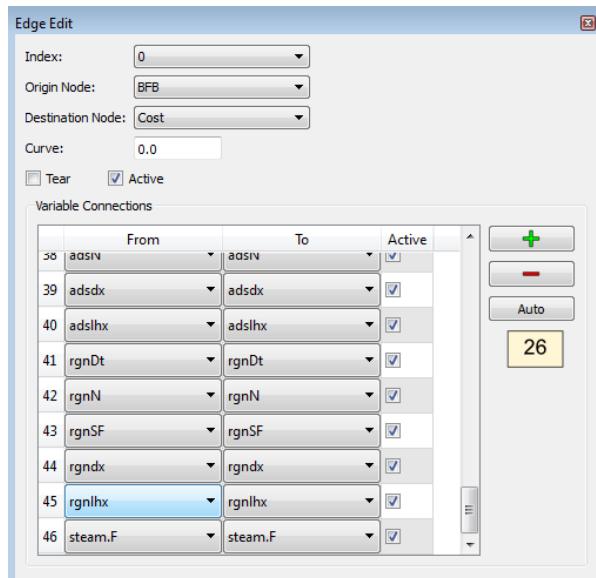
Each node must be assigned the appropriate simulation. Use the Node Editor to set the simulation type and the simulation name from simulation uploaded to Turbine. The Node Editor is illustrated in Figure 25

19. Under **Model** and **Type**, set the simulation **Type** to Turbine. This indicates that the simulation is to be run with Turbine.
20. Under **Model**, set the simulation of the BFB node to BFB_v6_2.
21. The **Variables** and **Settings** are automatically populated from the SimSinter configuration file. Variable values, Min/Max, and descriptions can be changed; however, for this problem, the values taken from the SimSinter configuration should not be changed.
22. Repeat the process for the cost node, assigning it the BFB_v6_2_cost simulation.

**Figure 25: Node Editor**

The connections between variables in the BFB simulation and the cost estimation spreadsheet must be set, so that required information can be transferred from the BFB simulation to the cost simulation.

23. Click **Toggle Node Editor** to hide the Node Editor (Figure 24).
24. Select the edge on the flowsheet with the **Selection** tool.
25. Click **Toggle Edge Editor** to show the Edge Editor. The Edge Editor is shown in Figure 26.
26. For convenience, all of the variables that should be connected from the ACM model to the Excel spreadsheet have been given the same names in their SimSinter configuration files. To connect the variables click **Auto** in the Edge Editor. **Auto** connects variables of the same name. Since this is often not desired, the **Auto** button should be used carefully. There should be 46 connected variables.

**Figure 26: Edge Editor**

The flowsheet should now be ready to run. Test the flowsheet by executing a single evaluation before setting up the optimization problem.

27. Click **Run** in the Flowsheet Editor (Figure 24).
28. The flowsheet may take a few minutes to run. The BFB simulation takes a significant amount of time to open in ACM. While running optimization, the evaluations take less time because the simulation remains opened. The simulation should complete successfully. A message box displays when the simulation is done. The status bar also indicates the simulation is running.
29. While the simulation is running, **Stop** is enabled.
30. Once the simulation runs successfully, **Save** the FOCUS session again, and **keep it for use in later tutorials**.

Tutorial – Flowsheets with Recycle

This section provides a tutorial on working with flowsheets containing recycle. Sections 2.7 and 2.8 provide tutorials for creating flowsheets, in this section a pre-constructed flowsheet is used.

1. From the example files, copy the Recycle\Mass_Bal_Test_02.foqus example file to a convenient location (see Section 2.6).
2. Open FOCUS.

3. Open the Mass_Bal_Test_02.foqus file.
 - (a) Open the **Session** drop-down menu on the right side of the **Session** button (Figure 27).
 - (b) Select **Open Session** from the drop-down menu.
 - (c) Locate Mass_Bal_Test_02.foqus in the file browser, and open it.
4. Click **Flowsheet** button from the toolbar at the top of the Home window.

The flowsheet is shown in Figure 27. The flowsheet consists of two reactors in recycle loops. The flowsheet contains mixers, reactors, separators, and splitters. Each node uses a set of simple calculations in the node script section. The tear edges are shown in light blue.

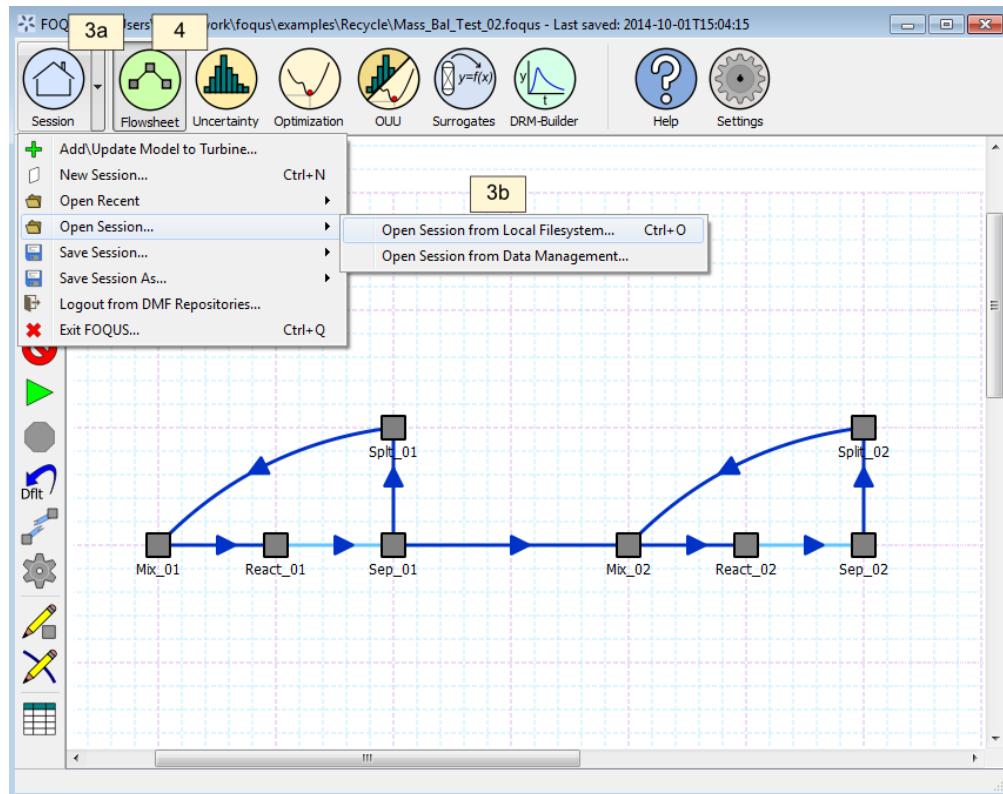


Figure 27: Flowsheet with Recycle

5. Inspect a node.
 - (a) Make sure the Selection tool is selected (Figure 28).
 - (b) Open the Node Editor by clicking the **Node Edit** button in the left toolbar in the Flowsheet view.
 - (c) Click the “React_01” node.
 - (d) Click **Input Variables** table. Note: Some input rows are colored red. This denotes that their values are set by output of the previous flowsheet node by the edge connecting “Mix_01” to “React_01.”

- (e) Click the **Node Script** tab.
- (f) Note the equations. **Input Variables** are stored in the x dictionary and **Output Variables** are stored in the f dictionary.
6. Click the gear icon in the left toolbar (see Figure 28). The tear solver settings are shown in Figure 29.

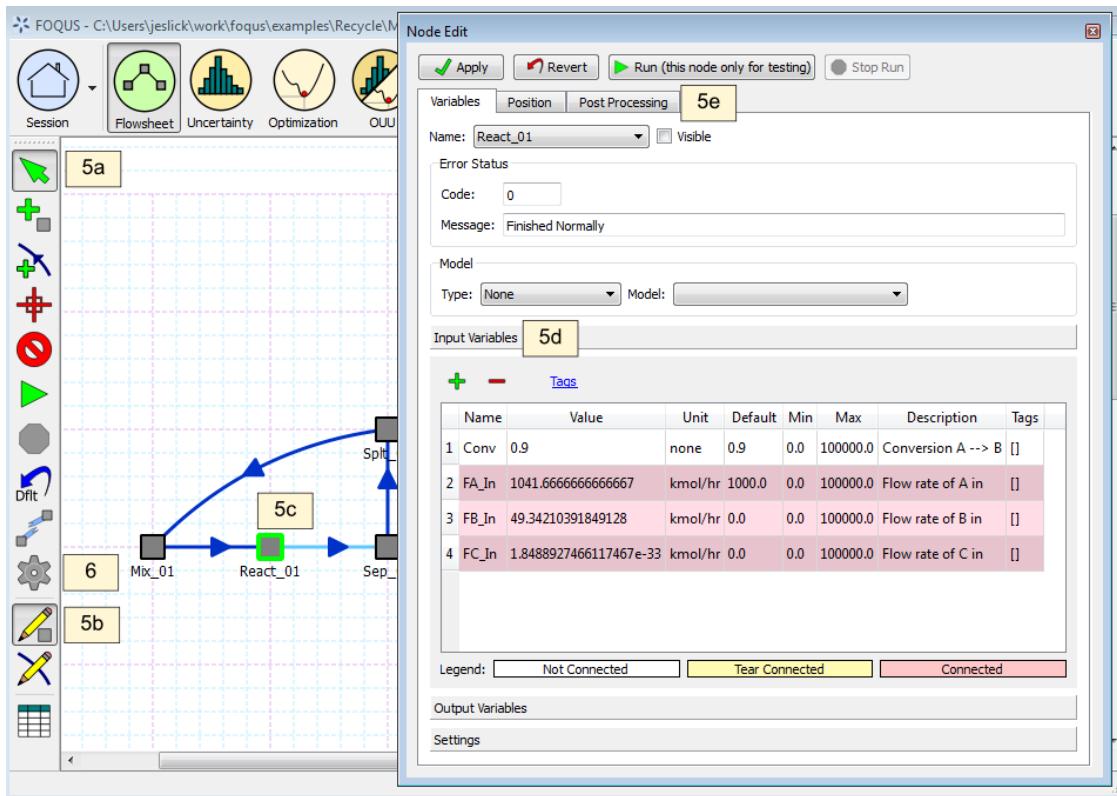


Figure 28: React_01 Node

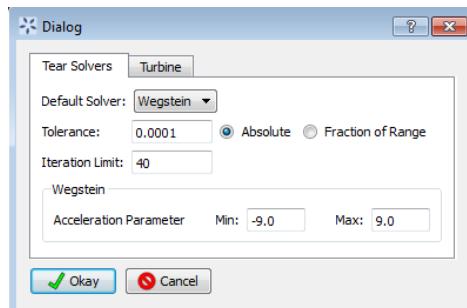


Figure 29: Tear Solver Settings

7. Remove the tear edges.

- Close the Node Editor.
 - Open the Edge Editor. Click the **Edge Editor** icon in the left toolbar (see Figure 30).
 - Click the edge between “React_01” and “Sep_01.”
 - In the Edge Editor, clear the **Tear** checkbox.
 - Repeat for the other tear edge.
8. Close the Edge Editor.

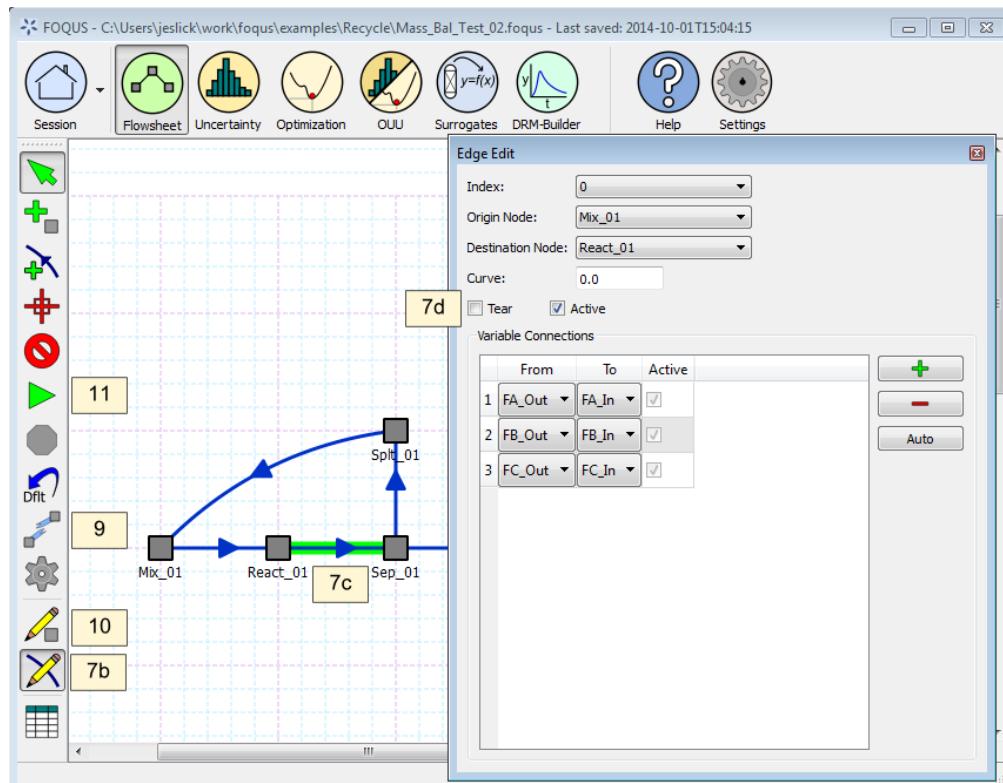


Figure 30: Edge Edit

There should now be no tear edges in the flowsheet. The user can select tear edges or FOCUS can automatically select a set. If there is not a valid set of tear edges marked when a flowsheet is run, tear edges will automatically be selected.

- Automatically select a tear edge set by clicking the **Tear** icon in the left toolbar (see Figure 30).
- Open the Node Editor and look at node “Sep_01.” In the Input Variables table, notice that some of the input lines are colored yellow. The yellow inputs serve as initial guesses for the tear solver. The final value will be different from the initial value.

11. Click the **Run** button on the left toolbar. The flowsheet should solve quickly.
12. The results of the completed run are in the flowsheet. An entry will also be created in the Flowsheet Results data table (see Section 2.10).

Tutorial – Flowsheet Result Data

Flowsheet evaluation results are stored in a table in the FOCUS session. This data can be used for many purposes. The flowsheet evaluations may be single runs, part of an optimization problem, or part of a UQ ensemble. This tutorial provide information about sorting, filtering, and exporting data.

Copy the Data\Simple_flow.foqus file from the example files to a convenient location (see section 2.6). This file is similar to the one created in the tutorial Section 2.7, but it has been run an additional 100 times using a UQ ensemble (see Chapter 4).

1. Open FOCUS.
2. Open the Simple_flow.foqus session from the example files.
3. Click the **Flowsheet** button from the Home window.
4. Click **Flowsheet Data** in the toolbar on the left side of the Home window (Figure 31).

	SetName	ResultName	Error	Time	ElapsedTime	SessionID	Tags	Error.calc	SimID.calc
0	"Single_runs"	"single_0"	0	"2015-06-11T19:42:04.193000"	0.0	0	[]	0	"None"
1	"Single_runs"	"single_0"	1	"2015-06-11T19:42:34.116000"	0.0	0	[]	21	"None"
2	"UQ_Ensemble"	"uq_000000"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
3	"UQ_Ensemble"	"uq_000001"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
4	"UQ_Ensemble"	"uq_000002"	1	"2015-06-11T19:56:55.544000"	0.0	0	[]	21	"None"
5	"UQ_Ensemble"	"uq_000003"	1	"2015-06-11T19:56:55.544000"	0.0	0	[]	21	"None"
6	"UQ_Ensemble"	"uq_000004"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
7	"UQ_Ensemble"	"uq_000005"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
8	"UQ_Ensemble"	"uq_000006"	1	"2015-06-11T19:56:55.544000"	0.0	0	[]	21	"None"
9	"UQ_Ensemble"	"uq_000007"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
10	"UQ_Ensemble"	"uq_000008"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
11	"UQ_Ensemble"	"uq_000009"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
12	"UQ_Ensemble"	"uq_000010"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
13	"UQ_Ensemble"	"uq_000011"	1	"2015-06-11T19:56:55.544000"	0.0	0	[]	21	"None"
14	"UQ_Ensemble"	"uq_000012"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
15	"UQ_Ensemble"	"uq_000013"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
16	"UQ_Ensemble"	"uq_000014"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
17	"UQ_Ensemble"	"uq_000015"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
18	"UQ_Ensemble"	"uq_000016"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"
19	"UQ_Ensemble"	"uq_000017"	0	"2015-06-11T19:56:55.544000"	0.0	0	[]	0	"None"

Figure 31: Flowsheet Results Data Table, All Data

A data table should be displayed like the one shown in Figure 31. There are 102 flowsheet evaluations. The first two evaluations are single runs, as can be seen in the SetName column, and the remaining 100 evaluation are from a UQ ensemble. The Error column shows several of the evaluations resulted in an error from a negative number being passed to the square root function.

This tutorial is broken up into mini-tutorials in the remaining subsections, which can be done independently. They each use the example data file described above.

Sorting Data

1. Open FOCUS.
2. Open the Simple_flow.foqus session from the example files.
3. Click Flowsheet in the main toolbar at the top of the FOCUS Home window.
4. Click Flowsheet Data in the toolbar on the left side of the Home window (Figure 31).
5. Click Edit Filters (Figure 31).

6. Click **New Filter**.
7. Enter “Sort1” as the new filter name.
8. Click **New Filter**.
9. Enter “Sort2” as the new filter name.
10. Select “Sort1” from the **Filter** drop-down list.
11. Select the **Sort Data** checkbox.
12. Enter `["-result"]` as the **Sort Term**. Include the square brackets. The square brackets indicate that there is a list of sort terms, although in this case there is only one. If multiple search terms are given, the additional terms will be used to sort results having the same value for the previous terms. The “-” in front of **result** indicates the results should be sorted in reverse. The names of the sort terms come from the column headings, and are case sensitive.
13. Click **Done** to save the filters and return to the results table.

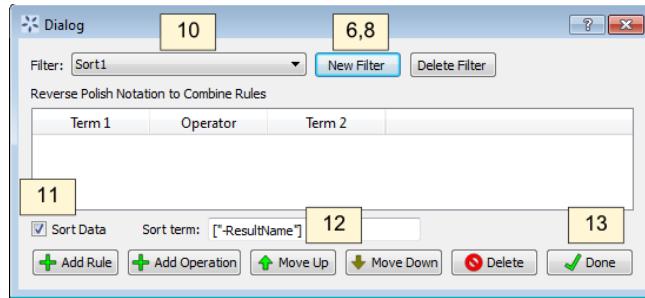


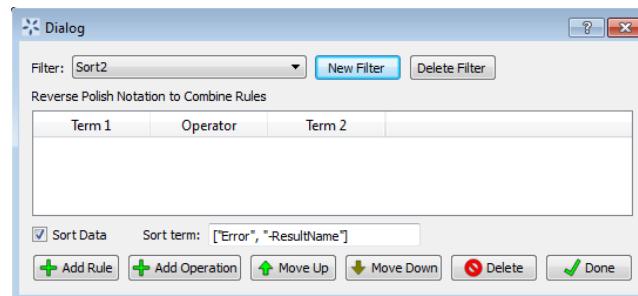
Figure 32: Sort1 Data Filter

14. Select “Sort1” from the **Current Filter** drop-down list (Figure 33).
15. The results are shown in Figure 33. The data should be sorted in reverse alphabetical order by **result**. Some of the columns are hidden to make the relevant results easier to see.

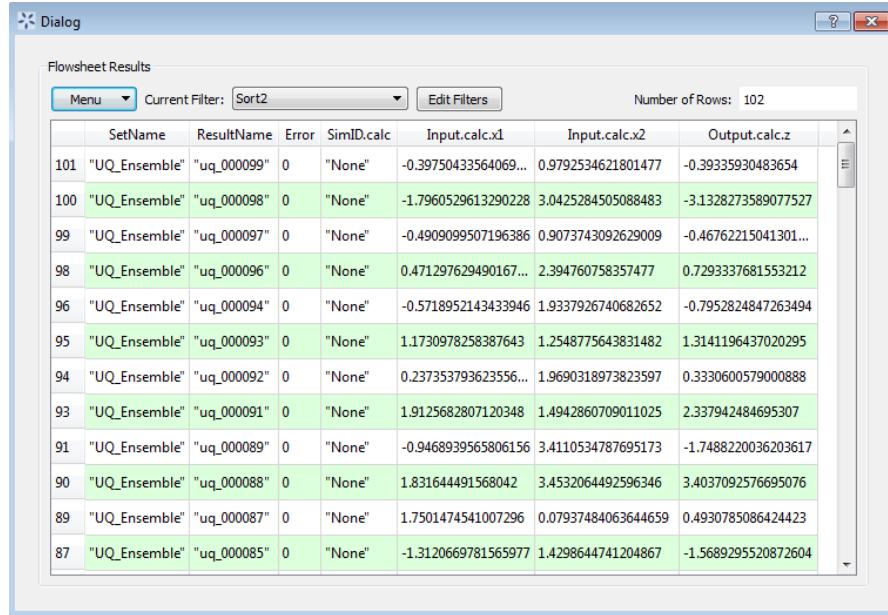
SetName	ResultName	Error	SimID.calc	Input.calc.x1	Input.calc.x2	Output.calc.z
101 "UQ_Engineer"	"uq_000099"	0	"None"	-0.39750433564069...	0.9792534621801477	-0.39335930483654
100 "UQ_Engineer"	"uq_000098"	0	"None"	-1.7960529613290228	3.0425284505088483	-3.1328273589077527
99 "UQ_Engineer"	"uq_000097"	0	"None"	-0.4909099507196386	0.9073743092629009	-0.46762215041301...
98 "UQ_Engineer"	"uq_000096"	0	"None"	0.471297629490167...	2.394760758357477	0.7293337681553212
97 "UQ_Engineer"	"uq_000095"	1	"None"	0.195479092409591...	-0.7336870602023262	-0.7952824847263494
96 "UQ_Engineer"	"uq_000094"	0	"None"	-0.5718952143433946	1.9337926740682652	-0.7952824847263494
95 "UQ_Engineer"	"uq_000093"	0	"None"	1.1730978258387643	1.2548775643831482	1.3141196437020295
94 "UQ_Engineer"	"uq_000092"	0	"None"	0.237353793623556...	1.9690318973823597	0.3330600579000888
93 "UQ_Engineer"	"uq_000091"	0	"None"	1.9125682807120348	1.4942860709011025	2.337942484695307
92 "UQ_Engineer"	"uq_000090"	1	"None"	-0.8341102535808973	-0.5413563407684473	-1.7488220036203617
91 "UQ_Engineer"	"uq_000089"	0	"None"	-0.9468939565806156	3.4110534787695173	-1.7488220036203617
90 "UQ_Engineer"	"uq_000088"	0	"None"	1.831644491568042	3.4532064492596346	3.4037092576695076

Figure 33: Sort1 Data Filter Results

16. Click **Edit Filters** (Figure 33). The dialog box in Figure 34 should display.
17. Select “Sort2” from **Filter** drop-down list.
18. Select the “Sort Data” checkbox.
19. Enter `["Error", "-result"]` in the **Sort Term** field. This will sort the data first by **Error** code then by **result** in reverse alphabetical order.
20. Click **Done**.

**Figure 34: Sort2 Data Filter**

21. Select “Sort2” in the **Current Filter** drop-down list (Figure 35).
22. The results are shown in Figure 35. The data should be sorted so all **Error** code zero results are first then sorted in reverse alphabetical order by **result**.



The screenshot shows a 'Flowsheet Results' dialog box. At the top, there is a menu bar with 'File', 'Edit', 'View', 'Help', and a 'Close' button. Below the menu is a toolbar with a 'New Filter' button, a 'Sort' dropdown set to 'Sort2', and an 'Edit Filters' button. To the right of the toolbar, it says 'Number of Rows: 102'. The main area is a table with the following columns: SetName, ResultName, Error, SimID.calc, Input.calc.x1, Input.calc.x2, and Output.calc.z. The table contains 102 rows, each representing a different simulation run. The rows are color-coded in green and white. The first few rows are:

SetName	ResultName	Error	SimID.calc	Input.calc.x1	Input.calc.x2	Output.calc.z
101 "UQ_Engineer"	"uq_000099"	0	"None"	-0.39750433564069...	0.9792534621801477	-0.39335930483654
100 "UQ_Engineer"	"uq_000098"	0	"None"	-1.7960529613290228	3.0425284505088483	-3.1328273589077527
99 "UQ_Engineer"	"uq_000097"	0	"None"	-0.4909099507196386	0.9073743092629009	-0.46762215041301...
98 "UQ_Engineer"	"uq_000096"	0	"None"	0.471297629490167...	2.394760758357477	0.7293337681553212
96 "UQ_Engineer"	"uq_000094"	0	"None"	-0.5718952143433946	1.9337926740682652	-0.7952824847263494
95 "UQ_Engineer"	"uq_000093"	0	"None"	1.1730978258387643	1.2548775643831482	1.3141196437020295
94 "UQ_Engineer"	"uq_000092"	0	"None"	0.237353793623556...	1.9690318973823597	0.3330600579000888
93 "UQ_Engineer"	"uq_000091"	0	"None"	1.9125682807120348	1.4942860709011025	2.337942484695307
91 "UQ_Engineer"	"uq_000089"	0	"None"	-0.9468939565806156	3.4110534787695173	-1.7488220036203617
90 "UQ_Engineer"	"uq_000088"	0	"None"	1.831644491568042	3.4532064492596346	3.4037092576695076
89 "UQ_Engineer"	"uq_000087"	0	"None"	1.7501474541007296	0.07937484063644659	0.4930785086424423
87 "UQ_Engineer"	"uq_000085"	0	"None"	-1.3120669781565977	1.4298644741204867	-1.5689295520872604

Figure 35: Sort2 Data Filter Result

Filtering Data

1. Open FOQUS.
2. Open the Simple_flow.foqus session from the example files.
3. Click the **Flowsheet** button in the Home window.
4. Click the Results Data button (Table icon in left toolbar).
5. In the data table dialog, click **Edit Filters**.
6. Click **New Filter** and enter “Filter1” in the **Filter** field as the new filter name.

The completed filter settings are shown in Figure 36. The buttons along the bottom of the filter editor dialog are used to add, delete, and change the order of rules and operations. Rules are conditions that are placed on data to be displayed. Operations are used to modify or combine rules. FOQUS uses reverse polish notation for the order of operations. The example filter in 36 is equivalent to (Error = 0 AND SetName ∈ {UQ_Engineer, Single_runs}) OR Index < 3.

The filter terms can be numbers, strings, lists of numbers, lists of strings or column headings. When entering column headings as search terms, do not enclose the column heading in quotes. This distinguishes column heading names from strings. The other types of terms are in JSON format. Strings are enclosed in double quotes, numbers have no quotes, and lists are enclosed in square brackets with elements separated by commas.

7. Click **Add Rule** three times.

8. Click **Add Operation** two times.
9. Fill out the rule and operation rows as shown in Figure 36.
10. Click **Done**.

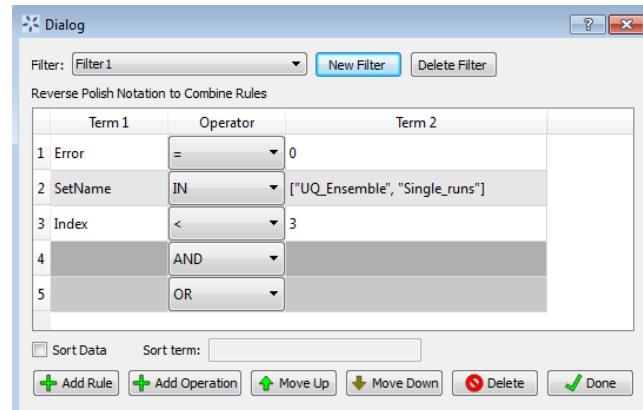


Figure 36: Filter1 Data Filter

11. In the data table dialog, select “Filter1” from the **Current Filter** drop-down list.
12. The result is displayed in Figure 37.

	SetName	ResultName	Error	SimID.calc	Input.calc.x1	Input.calc.x2	Output.calc.z
0	"Single_runs"	"single_0"	0	"None"	1.0	4.0	2.0
1	"Single_runs"	"single_0"	1	"None"	1.0	-1.0	2.0
2	"UQ_Engineering"	"uq_000000"	0	"None"	0.5097943262894611	3.593832273173068	0.9664377819199688
3	"UQ_Engineering"	"uq_000001"	0	"None"	1.4259387556770533	2.422953367383663	2.2195932772476556
6	"UQ_Engineering"	"uq_000004"	0	"None"	-1.5459458290347576	0.38594446651914266	-0.9604100172718022
7	"UQ_Engineering"	"uq_000005"	0	"None"	0.908627545828292	2.0495356084311083	1.300809298622604
9	"UQ_Engineering"	"uq_000007"	0	"None"	0.5776431548491323	0.4238267901464491	0.37605713564620236
10	"UQ_Engineering"	"uq_000008"	0	"None"	-0.6387752601498622	2.691315656081455	-1.0479254949225727
11	"UQ_Engineering"	"uq_000009"	0	"None"	-1.013818347553638	2.562607965274997	-1.6229353967116953
12	"UQ_Engineering"	"uq_000010"	0	"None"	-1.336057825859663	1.5514102688298608	-1.6641351200737864
14	"UQ_Engineering"	"uq_000012"	0	"None"	0.6427300924075441	3.7208154736649317	1.2397887757402641
15	"UQ_Engineering"	"uq_000013"	0	"None"	0.2907802977463141	1.3846866139372283	0.342169049839965

Figure 37: Filter1 Data Filter Result

Exporting Data

This tutorial uses a spreadsheet program such as Excel or Open Office. The exported data is subject to the selected filter. See the previous tutorials in this section for more information about sorting and filtering data to be exported.

Clipboard

FOQUUS can export data directly to the Clipboard. The data can be pasted into a spreadsheet or as text. Copying data to the Clipboard eliminates the need for an intermediate file when creating spreadsheets.

1. Open FOCUS.
2. Open a spreadsheet program.
3. Open the Simple_flow.foqus session from the example files.
4. Click the **Flowsheet** button in the Home window.
5. Click the Results Data button (Table icon in left toolbar).
6. Click on the **Menu** drop-down list in the data table dialog.
7. Select “Export” from the **Menu** drop-down list.
8. Click **Copy Data to Clipboard**.
9. Select Paste in the spreadsheet program. The data table in FOCUS should paste into the spreadsheet. Filters can be used to sort or reduce the exported data.

CSV File

CSV (comma separated value) files can be read by almost any spreadsheet program, and are common formats readable by many types of software. FOCUS exports CSV files using the column headings from the data table as a header.

1. Open FOCUS.
2. Open a spreadsheet program.
3. Open the Simple_flow.foqus session from the example files.
4. Click the **Flowsheet** button in the Home window.
5. Click the Results Data button (Table icon in left toolbar).
6. Click the **Menu** drop-down list.
7. Select “Export” from the **Menu** drop-down list.

8. Click **Export to CSV File**.
9. Enter a file name in the file dialog.
10. In the spreadsheet program, open the CSV file exported in the previous step.

Tutorial – Using a Remote Turbine Instance

A remote Turbine instance may be used instead of TurbineLite. TurbineLite, used by default, runs simulations (e.g., Aspen Plus) on the user's local machine. The remote Turbine gateway has several potential advantages over TurbineLite, while the main disadvantage is the effort required for installation and configuration. Some reasons to run a remote Turbine instance are:

- Simulations can be run in parallel. The Turbine gateway can distribute simulations to multiple machines configured to run FOCUS flowsheet consumers. FOCUS consumers are basically additional instances of FOCUS running on remote systems which can run a FOCUS flowsheet.
- Simulations can be run on machines other than the user's, so as not to tie-up the user's machine running simulations.

The steps below demonstrate how to set up FOCUS to run flowsheets remotely (see Figure 39).

1. Obtain a user name, password, and URL from the site's Turbine administrator.
2. Open FOCUS.
3. Click **Settings** at the top right of the Home window (Figure 38).
4. Select “Remote” from the **FOCUS Flowsheet Run Method** drop-down list.
5. Click the **Turbine** tab; this displays the Turbine settings shown in Figure 39.

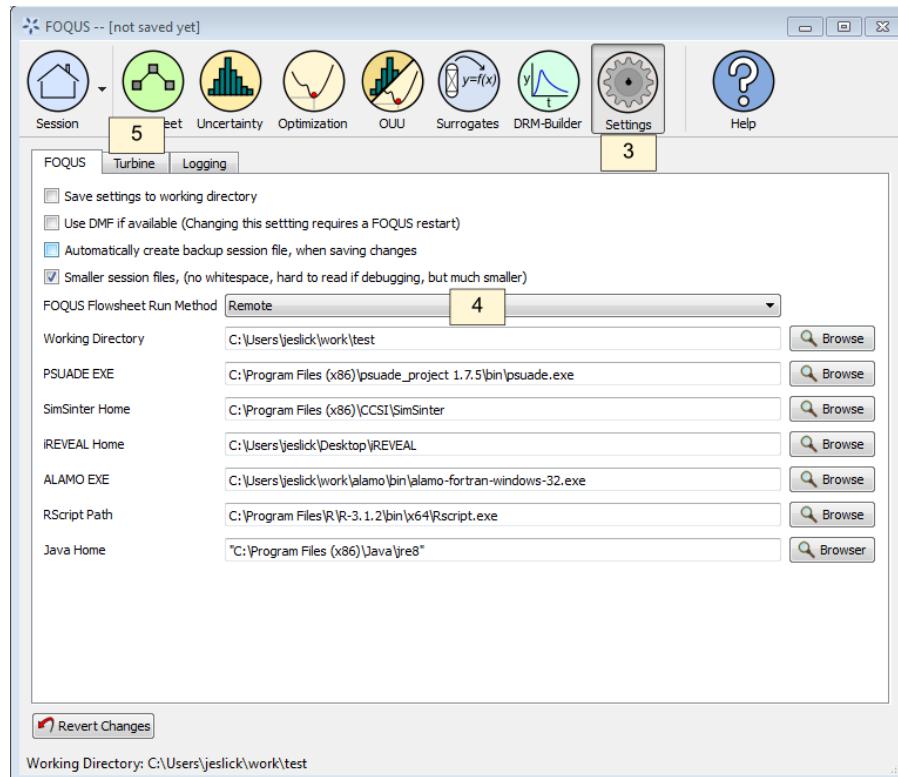


Figure 38: Run Method Settings

6. Create a Turbine configuration file; this contains your password in plain text, so it is very important that if you are allowed to choose your own password, you choose one that is not used for any other purpose.
 - (a) Click **New/Edit** next to the **Turbine Configuration (remote)** field. The Turbine Configuration window displays (see Figure 39).
 - (b) Select “Cluster/Cloud” from the **Turbine Gateway Version** drop-down list in the Turbine Configuration window.
 - (c) Enter the Turbine URL in the **Address** field.
 - (d) Enter the **User** name and **Password**.
 - (e) Click **Save as** and enter a new file name.
 - (f) Set the remote Turbine configuration file. Click **Browse** next to the **Turbine Configuration (remote)** field. Select the file created in Step 6e.

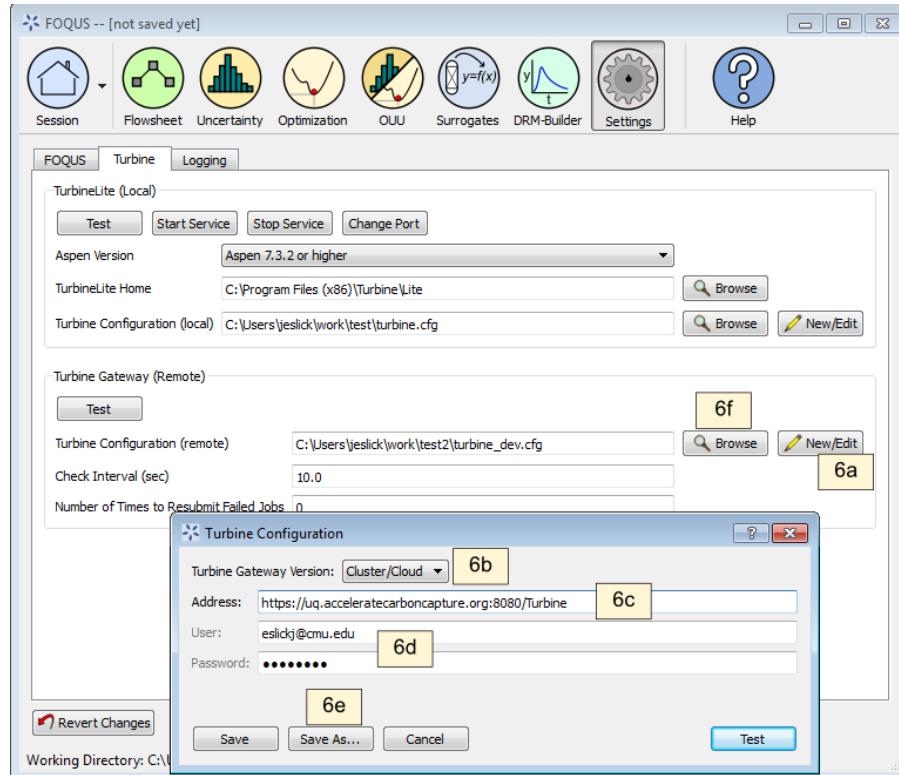


Figure 39: Remote Turbine Settings

At this point the remote gateway is ready to use. The last step is to ensure that all simulations referenced by flowsheets to be run are uploaded to the remote Turbine gateway.

7. Upload any necessary simulations to Turbine (see Section 2.2.3 and the tutorial in Section 2.8)

Once all settings are specified there is no apparent difference between running flowsheets locally or on a remote Turbine gateway, and FOCUS can readily be switched between the two.

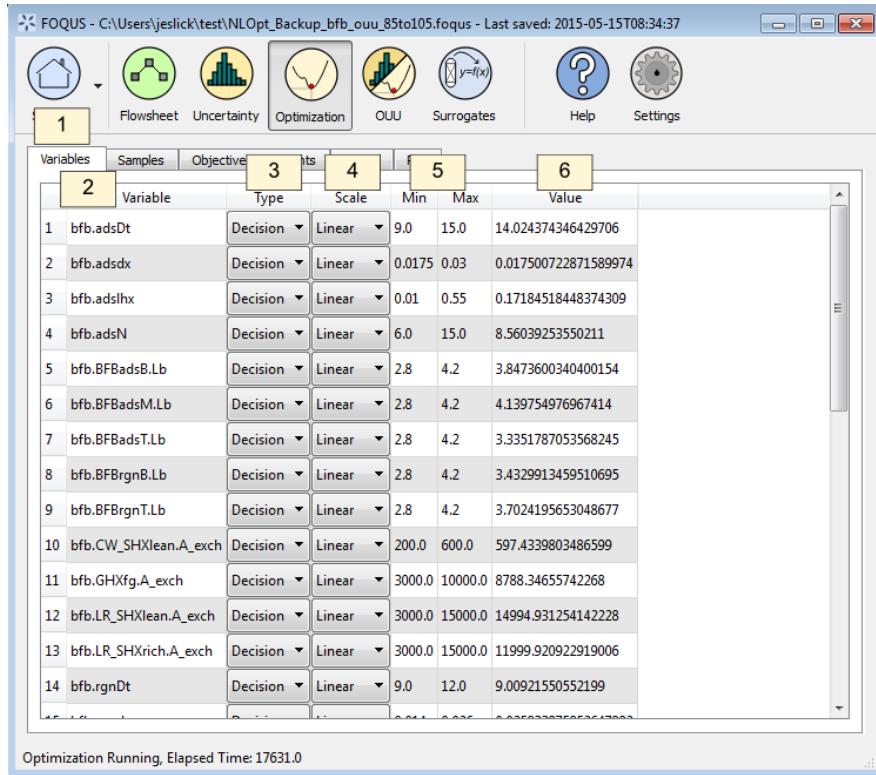
3. OPTIMIZATION

The simulation based optimization tool provides a plug-in system where different derivative free optimization (DFO) solvers can be used with FOCUS. Several solvers are provided with FOCUS. The CMA-ES solver [7] is a good global derivative free optimization (DFO) solver. The NLOpt library provides access to several DFO solvers [8]. SLSQP and BFGS from the Scipy module are also provided [9]. Since FOCUS does not generally have access to derivative information the Scipy solvers rely on finite difference approximations, and should only be used with well-behaved functions. Due to convergence tolerances in process simulators, finite difference approximations may not be good for many of FOCUS's intended applications.

CMA-ES offers a restart feature, which can be used to resume an optimization if it is interrupted for any reason. Other solvers may use an auto-save feature, which does not provide the ability to restart, but will allow optimization to start from the best solution found up to the point the optimization was interrupted. Samples making up the population in CMA-ES can be run in parallel. The NLOpt and Scipy plugins do not offer parallel computing for standard optimization. For any solver, parallel computation can be used for parameter estimation and optimization under uncertainty, where multiple flowsheet evaluations go into an objective function calculation.

Problem Set Up

See Chapter 2 for information about setting up a flowsheet in FOCUS. Once the flowsheet has been set up and tested, an optimization problem can be added. FOCUS allows multiple flowsheet evaluations to be used to calculate a single objective function value. This allows FOCUS to do parameter estimation and scenario based optimization under uncertainty. There are three types of variables used in the optimization problem: (1) fixed variables do not change during the optimization, (2) decision variables are modified by the optimization algorithm to find the best value of the objective function, and (3) sample variables, which are used to construct the multiple flowsheet evaluations that can go into an objective calculation. If no sample variables are defined, each objective function value will be based on a single flowsheet evaluation. Figure 40 shows the Variables tab selection form.

**Figure 40: Optimization Variable Selection**

1. The **Variables** tab contains the form for variables selection.
2. The **Variable** column shows the name of input variables in the flowsheet. If a variable is set by a connection to another variable through an edge, it is not shown in the table. The format for a variable name is {Node Name}.{Variable Name}.
3. The **Type** column allows the variables to be assigned as one of three types (1) fixed, (2) decision, or (3) sample.
4. The **Scale** column allows the scaling method to be set for each variable. Decision variables must be scaled. Scaling is ignored for other variables. In the FOCUS example files, there is a scaling spreadsheet that provides a demonstration of the different scaling methods. The upper and lower bound are used in the scaling calculations. Regardless of the scaling method, the optimizer sees the decision variables as running from 0 at their minimum to 10 at their maximum.
5. The **Min** and **Max** columns are used to define the upper and lower bounds for the variables. FOCUS requires that all optimization problems be bounded.
6. The **Value** column provides the starting point for the optimization. How the starting point is used depends on the optimization method. The starting point for sample variables is irrelevant. Fixed variables will remain at their starting point during the optimization.

The sample variables define a set of samples that will be used to calculate an objective function.

For each objective function, the decision variables are fixed at values set by the optimization solver, and the flowsheet is evaluated for each row on the sample table. The results of the samples can be used to calculate the objective function. Using the **Samples** tab is optional. If no sample variables are set, each objective function value will be based on a single simulation. Figure 41 shows the Samples table form.

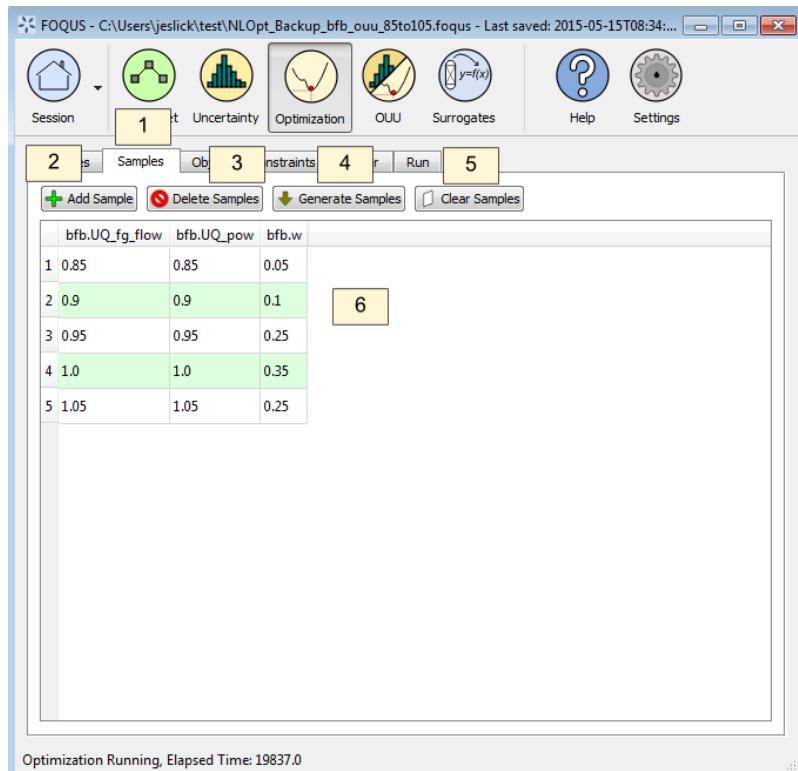


Figure 41: Optimization Sample Table

1. The **Samples** tab contains the table used to define samples for objective function calculations. If there are no sample variables, the table should be empty.
2. **Add Sample** adds a row to the Samples table.
3. **Delete Samples** deletes the selected rows from the Samples table.
4. **Generate Samples** opens a dialog box that provides a selection of methods to generate samples or read samples from a file.
5. **Clear Samples** clears the Samples table.

Once the variables and (optionally) samples have been selected, the objective function and constraints can be defined. FOCUS is set up to handle multi-objective optimization, but no multi-objective optimization plug-ins are currently provided in the FOCUS installer, so some of the options may seem to be extraneous. There are two methods for entering the objective function and constraints into FOCUS: (1) Simple Python expressions and (2) a more extensive Python function.

Python expressions are easier and sufficient for most cases. If the objective function is complicated it may be necessary to write a Python function, which can be as complex as needed.

The variables used in the Python code for the objective function or constraints are stored in two Python dictionaries, “f” for outputs and “x” for inputs. There are two ways to index the dictionaries depending on whether or not sample variables are used. For an input variable with sampling, the indexing is `x[Sample Index]['Node Name']['Variable Name'][Time Step Index]`. If no sample variables are defined, the sample index is not needed, so the indexing would be, `x['Node Name']['Variable Name'][Time Step]`. Node Name and Variable Name are strings so they should be in quotes. The sample and time step indexes are integers. For steady state simulations, the time step should be 0.

Figure 42 shows the form for entering the objective function and constraints as Python expressions.

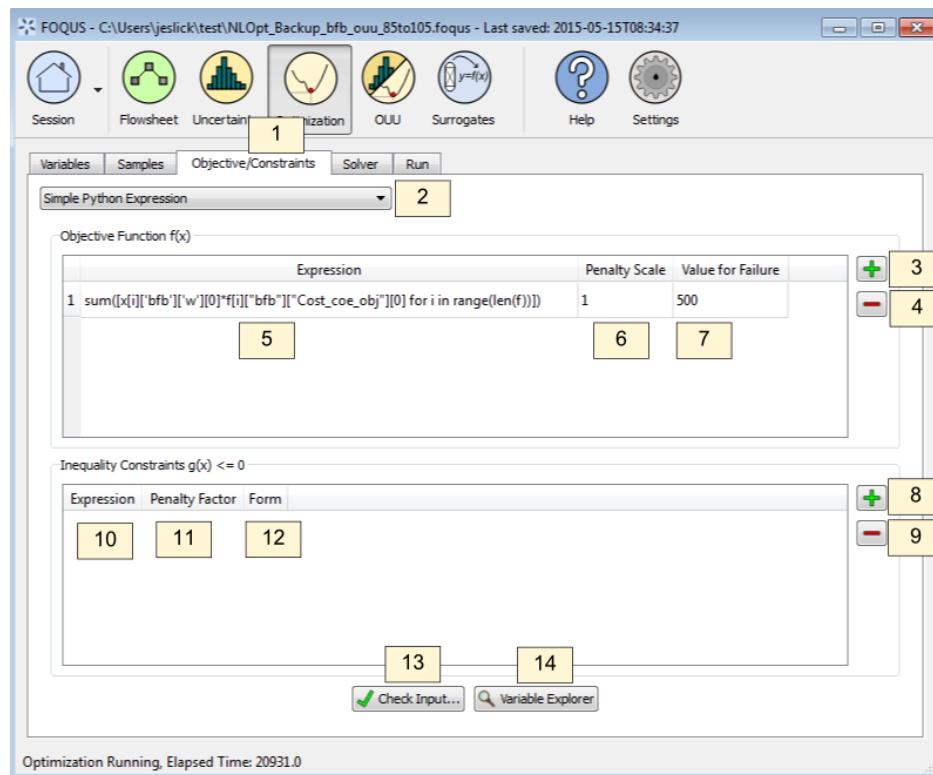


Figure 42: Optimization Simple Objective Function

1. The **Objective/Constraints** tab contains the form used to enter the objective function and constraints.
2. The drop-down list enables the selection of either the “Simple Python Expression” or “Custom Python” form of the objective function.
3. + adds an objective function to the table. The solvers currently available are single objective and will only use the first objective function.

4. - removes the selected objective from the table.
5. The Python expression for the objective function can be entered in the **Expression** column.
6. The **Penalty Scale** column is intended for use with multi-objective solvers and allows the constraint violation penalty to be applied differently to objective functions with different magnitudes.
7. The **Value for Failure** column contains the value to be assigned to the objective function if the objective cannot be evaluated for any reason. The value should be higher than the expected highest value for a successful objective.
8. + adds an inequality constraint.
9. - removes selected inequality constraints.
10. The inequality constraints are in the form $g(\mathbf{x}) \leq 0$. The **Expression** column contains the Python expression for $g(\mathbf{x})$.
11. The **Penalty Factor** contains the coefficient a used in calculating the penalty for a constraint violation, see Equations 3.1 to 3.3.
12. The **Form** column contains a selection of different methods to calculate a constraint penalty.
13. **Check Input** checks the problem for any mistakes that can be detected before running the optimization.
14. **Variable Explorer** enables the user to browse the variables in the simulation. They can be copied and pasted into the Python expression. The variables are provided without the sample index.

The calculations for each type of constraint penalty are given in Equations 3.1 to 3.3.

$$\text{Linear penalty form: } p_i = \begin{cases} 0 & \text{if } g_i(\mathbf{x}) \leq 0 \\ a \times g_i(\mathbf{x}) & \text{if } g_i(\mathbf{x}) > 0 \end{cases} \quad (3.1)$$

$$\text{Quadratic penalty form: } p_i = \begin{cases} 0 & \text{if } g_i(\mathbf{x}) \leq 0 \\ a \times g_i(\mathbf{x})^2 & \text{if } g_i(\mathbf{x}) > 0 \end{cases} \quad (3.2)$$

$$\text{Step penalty form: } p_i = \begin{cases} 0 & \text{if } g_i(\mathbf{x}) \leq 0 \\ a & \text{if } g_i(\mathbf{x}) > 0 \end{cases} \quad (3.3)$$

If the Simple Python Expression method of entering the objective function does not offer enough flexibility, the Custom Python method can be used. The Custom Python method enables the user to enter the objective calculation as a Python function, which also should include any required constraint penalties.

Figure 43 shows the Custom Python objective form. The top text box provides instructions for writing a custom objective function. The bottom text box provides a place to enter Python code. The numpy and math modules have been imported and are available as numpy and math. To use the Custom Python objective, the user must define a function called “onjfunc(x, f, fail).” The three arguments are: (1) “x” is the dictionary of input variables, (2) “f” is the dictionary of output variables, and (3) “fail” is a boolean vector that indicates whether a particular sample calculation has failed. The “objfunc” function should return three values: (1) a list of objective function values for multi-objective optimization (in most cases with single objective optimization this will be a list with one value), (2) a list of constraint violations, and (3) the total constraint penalty. The constraint violation and penalty information are only used for debugging, so they are not required. It is safe to return [0] and 0 for the constraint information regardless of whether a constraint penalty has been added to the objective.

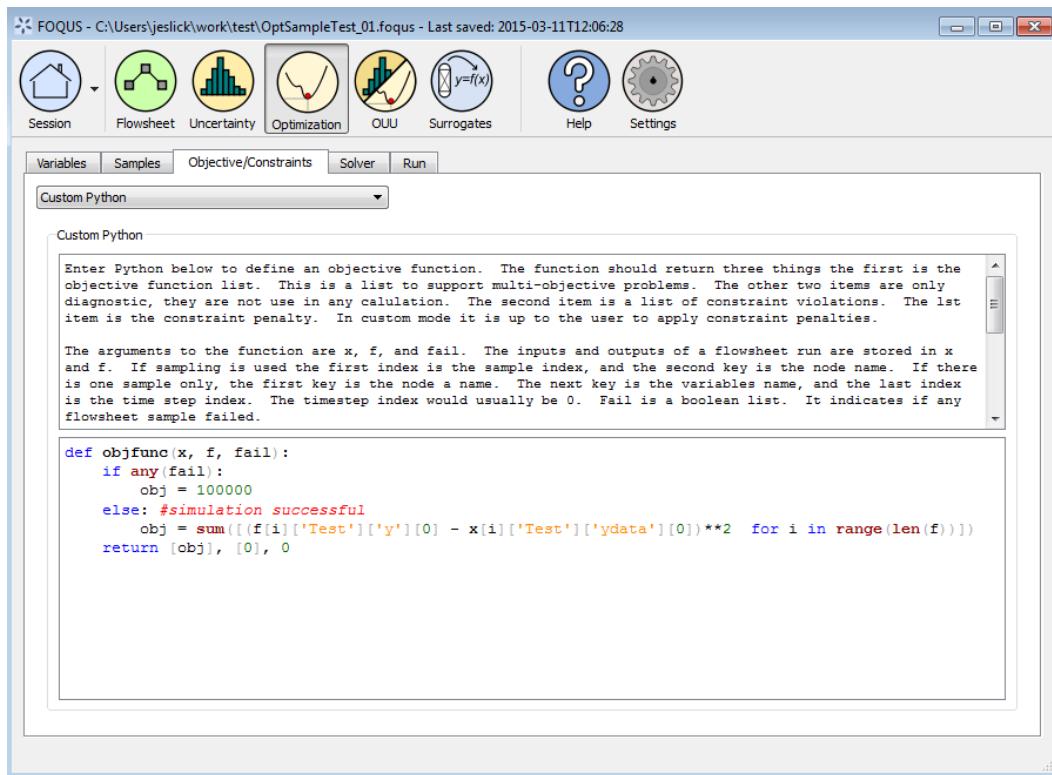


Figure 43: Custom Objective Function

The code in Figure 44 provides an example of a custom objective function for parameter estimation. The objective function minimizes the sum of the differences between simulation and empirical data. In this case the decision variables would be model parameters. The first line defines a function with three arguments. The “x” and “f” arguments are the input and output variables. The variable indexing is explained in the simple objective function section. The “fail” argument is a boolean vector where element “i” is true if sample “i” failed. If there are no sample variables, “fail” will only have one element.

The “if” in the function determines if any flowsheet evaluation failed, and assigns a bad objective function value if so. If all the flowsheet evaluations were successful, the results are used to calculate the objective function. In the objective function calculation, Python list comprehension is used to calculate the sum of squared errors. In this case, no constraint penalty is needed. The objective function is returned as a list with only one element. The last two return values are debugging information for constraints. In this case, the “zeros” are just place holders and have no real utility.

```
def objfunc(x, f, fail):
    if any(fail): # any simulation failed
        obj = 100000
    else: #simulations successful
        obj=sum([(f[i]['Test']['y'][0] - x[i]['Test']['ydata'][0])**2 \
                  for i in range(len(f))])
    return [obj], [0], 0
```

Figure 44: Custom Objective Function Code

Solver Options

The **Solver** tab in the **Optimization** button tool enables the selection of the DFO method and setting of solver parameters. Figure 45 illustrates the solver form.

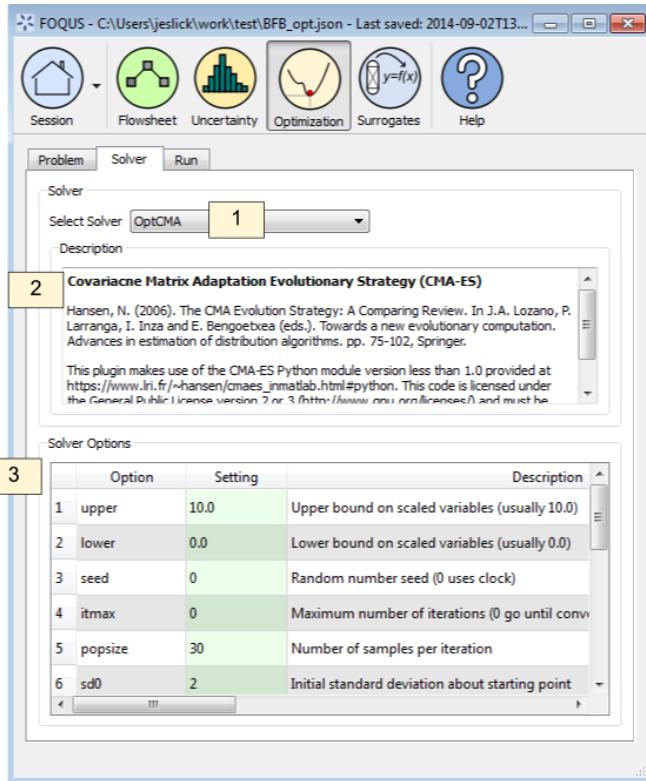


Figure 45: Optimization Solver Form

Elements of the solver form are:

1. **Select Solver** drop-down list, which enables the user to select from available DFO solvers.
2. **Description** text box provides a description of the selected DFO solver.
3. **Solver Options** table contains the solver settings and a description of each option. The settings depend on the selected plug-in.

Running Optimization

The optimization monitor is displayed under the **Run** tab in the **Optimization** button tool. The optimization monitor, illustrated in Figure 46, is used to monitor the progress of the optimization as it runs.

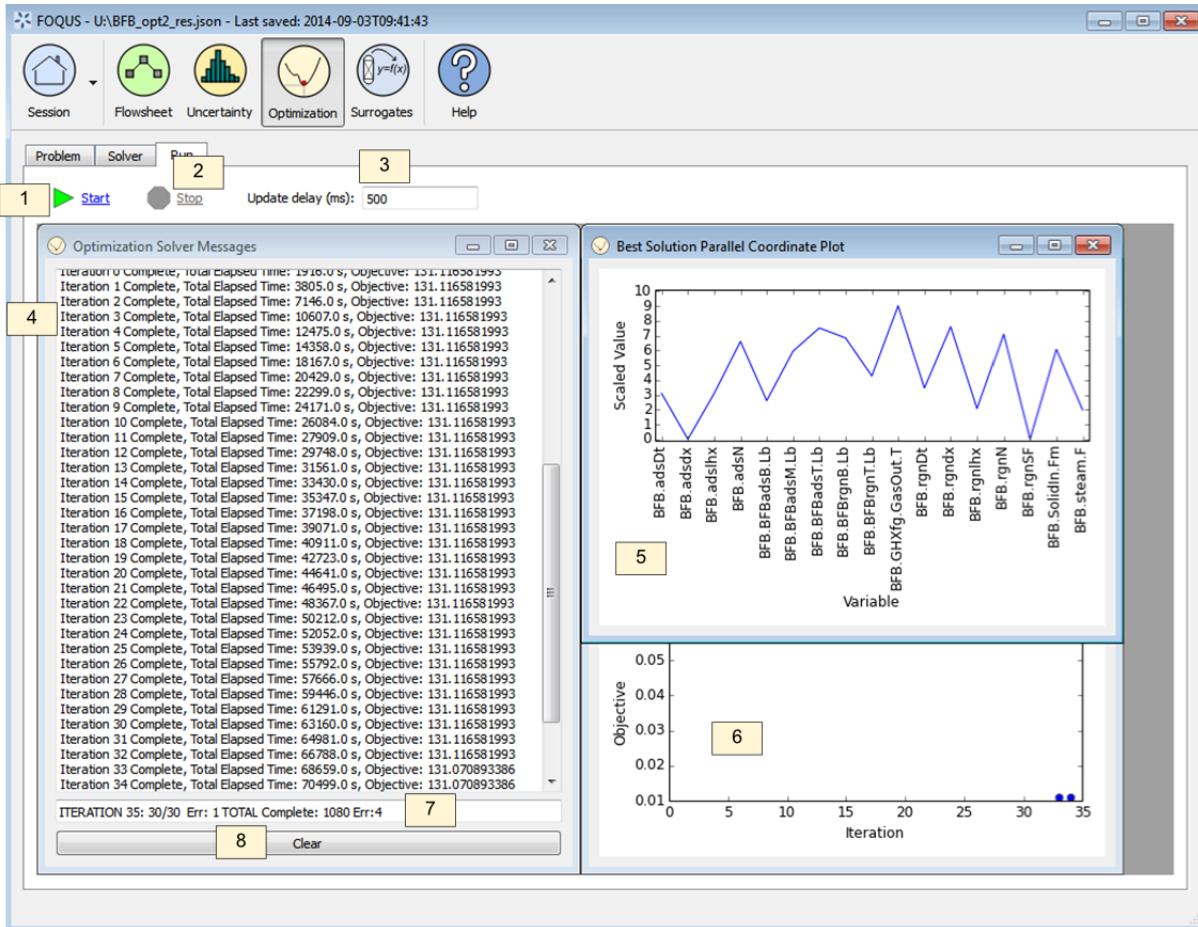


Figure 46: Optimization Monitor Form

Elements of the optimization monitor are:

1. **Start** starts the optimization.
2. **Stop** stops the optimization. The best solution found when optimization is stopped is stored in the flowsheet.
3. **Update delay** is how often the user interface communicates with the optimization thread to update the display.
4. **Optimization Solver Messages** displays output from the optimization solver.
5. **Best Solution Parallel Coordinate Plot** displays the values of the decision variables scaled. This plot is helpful in identifying when variables are at, or near, their bounds.
6. **Objective Function Plot** displays the objective function value at each iteration.
7. **Status Box** displays the current iteration, how many samples have been run, how many sample were successful, and how many failed.
8. **Clear** deletes solver messages from the solve message box.

As the optimization runs, the FOCUS flowsheet is updated to include the best solution found. If sampling is used, the first sample in the best objective function is stored in the flowsheet. If for any reason the optimization terminates, the best solution found is available in the flowsheet. The results for all flowsheet evaluations done for the optimization are available in the Results table in the Flowsheet Editor.

Tutorial – Optimization

This tutorial is a step-by-step walk through of simulation-based optimization. This tutorial builds on the tutorial in Section 2.8.

1. Open FOCUS.
2. Load the FOCUS session from the tutorial “Creating a Flowsheet with Linked Simulations” in Section 2.8 or if that tutorial has not yet been completed, complete it first.

Problem Set Up

If the simulation runs successfully and the results are reasonable, proceed to define the optimization problem. There are four steps to setting up the optimization problem: (1) select the variables, (2) define samples (optional), (3) define the objective function, and constraints and (4) select and configure the solver.

3. Select the **Optimization** button from the toolbar at the top of the Home window (Figure 47). Select the **Variables** tab.
4. Select “Decision” from the drop-down list in the **Type** column as the variable type for all 17 variables shown. If more than 17 variables are shown, the edge connecting the “BFB” node to the “Cost” node was most likely not configured properly. The scale will automatically change to linear, which is acceptable for most problems.
5. The **Min**, **Max**, and **Value** columns can be changed. The **Min** and **Max** columns define the lower and upper bounds. The **Value** column specifies the initial point. For this example the defaults are acceptable.

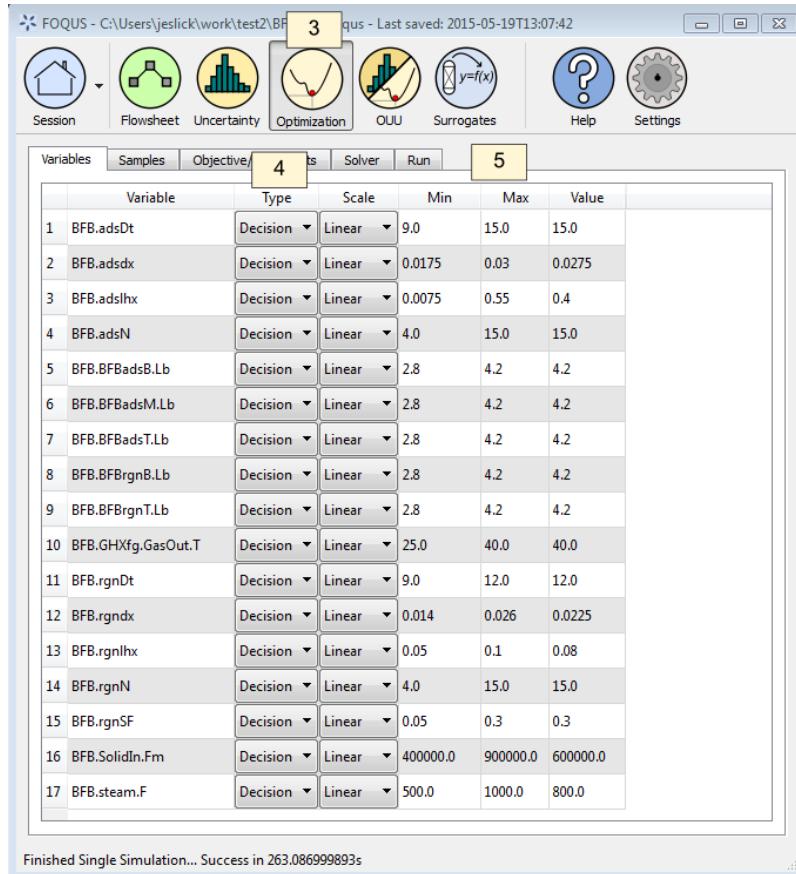


Figure 47: Optimization Problem Variables

If more than one flowsheet evaluation is used in the objective function calculation (e.g., parameter estimation or optimization under uncertainty), the next step is to setup the samples under the Samples tab. In this case only one evaluation is used to calculate an objective function value, so the sample setup is not needed. The next step is to define the objective function and constraints using the form under the Objective/Constraints tab as shown in Figure 48.

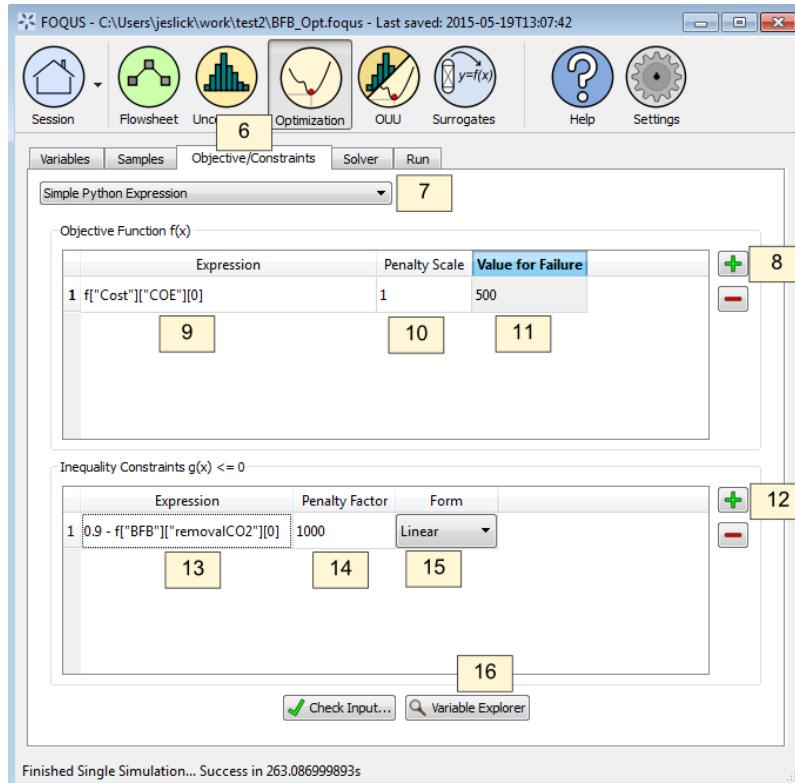


Figure 48: Optimization Problem Objective

6. Select the **Objective/Constraints** tab (see Figure 48).
7. In the drop-down list, verify “Simple Python Expression” is selected.
8. Add an objective function by clicking **+** to the right of the Objective Function table.
9. The objective function is the cost of electricity from the cost spreadsheet. Enter:
 $f["Cost"]["COE"][0]$
in the **Expression** column.
10. Enter 1 in the **Penalty Scale** column. This setting is used mostly for multi-objective optimization to apply the constraint penalty to different objectives.
11. Enter 500 in the **Value for Failure** column. This should be worse than the objective for any non-failed simulations.
12. Add a constraint by clicking **+** next to the Inequality Constraints table.
13. The constraint is that the fraction of CO₂ captured must be greater than or equal to 0.9. The constraint is in the form $g(x) \leq 0$; therefore, in the **Expression** column enter:
 $0.9 - f["BFB"]["removalCO2"][0]$.
14. Enter 1000 for the **Penalty Factor**.
15. The constraint penalty **Form** should be linear.
16. The **Variable Explorer** button can be used to help select flowsheet variables.

Solver Settings

The last step before running the optimization is to select and configure the solver. The solver configuration form is shown in Figure 49.

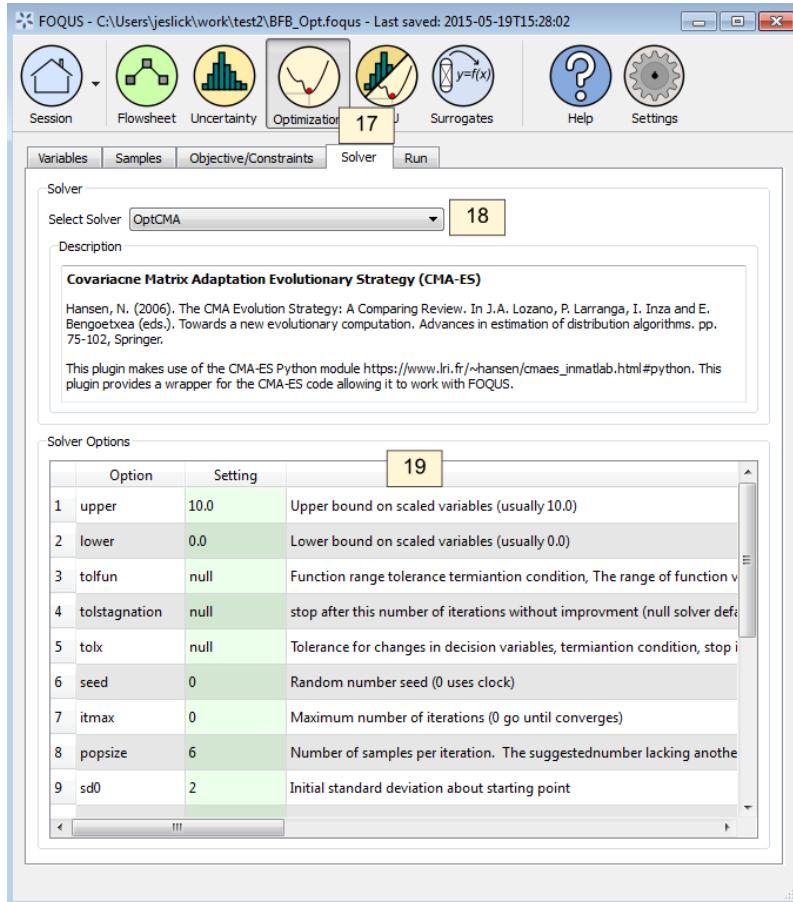


Figure 49: Optimization Solver Setup

17. Select the **Solver** tab (see Figure 49).
18. Select “OptCMA” from the **Select Solver** drop-down list.
19. The default options are acceptable. Solver options are described in the Solver Options table.

Running Optimization

The optimization run form is shown in Figure 50.

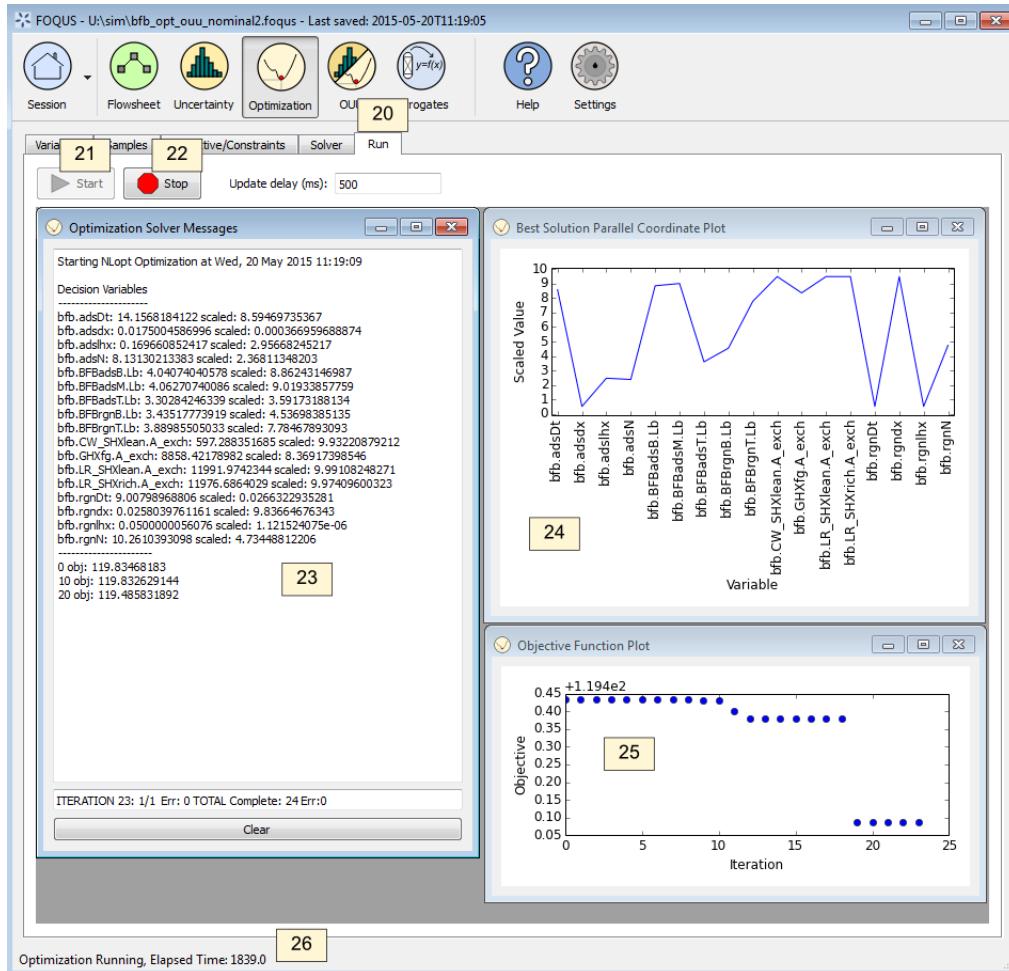


Figure 50: Optimization Monitor

20. Click the **Run** tab to display the optimization run form (see Figure 50).
21. Click **Start**.
22. Once the optimization has run for while click **Stop**.

As the optimization run, the best result found is stored in the Flowsheet. If an optimization is run with sample variables the first sample in the set with the best objective function will be stored in the flowsheet. All simulation results can be viewed in the Flowsheet Results table.

The run form displays some diagnostic information as the optimization runs. The parts of the display labeled in Figure 50 are described below.

23. The Optimization Solver Messages window displays information from the solver.
24. The **Best Solution Parallel Coordinate Plot** shows the value of the scaled decision variables, which is useful to see where the best solution is relative to the variable bounds.

25. The **Objective Function Plot** shows the best value of the objective function found as a function of the optimization iteration or sample number.
26. While the optimization is running, the status bar shows the amount of time that has elapsed since starting the optimization.

Tutorial – Parameter Estimation

Note: The NLOpt solvers are used for the tutorial, but are an optional to the installation. See the install instructions for more information about installing NLOpt.

This tutorial provides a very simple example of using the sampling with optimization. Sampling can be used to do optimization under uncertainty where there are several scenarios with differing values of uncertain parameters. Sampling can also be used to do parameter estimation, where estimated values must be compared against several data points. This tutorial will focus on parameter estimation.

At any point in this tutorial, the FOQUS session can be saved and the tutorial can be started again from that point.

The model is given by Equation 3.4. The unknown parameters are a , b , and c . The x and y data are given in Table 3.

$$y = ax^2 + bx + c \quad (3.4)$$

Table 3: x-y Data

Sample	1	2	3	4	5
x	0	1	2	3	4
y	1	0	3	10	21

The first step is to create a flowsheet with one node. The node will have the input variables: a , b , c , x , and $ydata$; and output variable y .

1. Open FOQUS.
2. In the **Session Name** field, enter “PE_tutorial” (see Figure 51).
3. Click the **Flowsheet** button in the top toolbar.

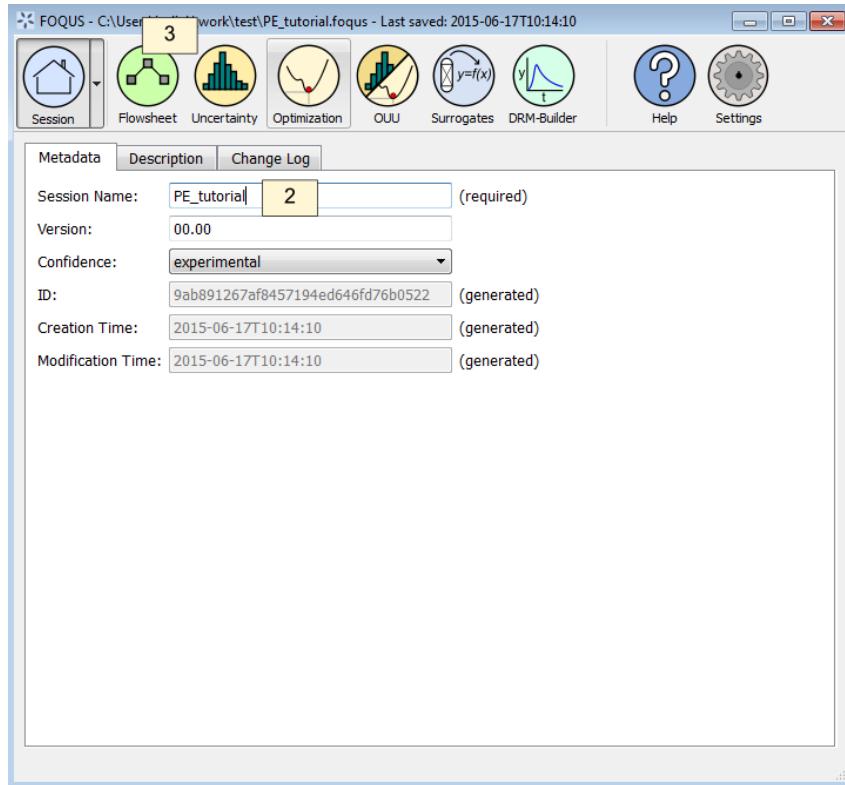


Figure 51: Session Setup

4. Add a node to the flowsheet named “model.”
 - (a) Click **Add Node** in the left toolbar (see Figure 52).
 - (b) Click anywhere on the gridded flowsheet area.
 - (c) Select “model” in the **Name** drop-down list and then click **OK**.
5. Click the **Selection Mode** icon in the left toolbar (see Figure 52).
6. Click the **Node Editor** icon in the left toolbar (see Figure 52).
7. In the Node Edit input table, add the variables a, b, c, x, and ydata. The ydata variable will be used as an input for the known y sample point data, later in the tutorial.
 - (a) Click the **Add Input** icon (see Figure 52).
 - (b) Enter “a” for the variable name in the **Name** column.
 - (c) Enter -10 and 10 for the min and max in the **Min** and **Max** columns for a, b, c, and x.
 - (d) Repeat for all of the inputs.
 - (e) Enter 1 for the value of a, b, and c in the **Value** column.
 - (f) Enter 2 for the value of x in the **Value** column.
 - (g) The **Value**, **Min**, and **Max** for ydata do not matter.

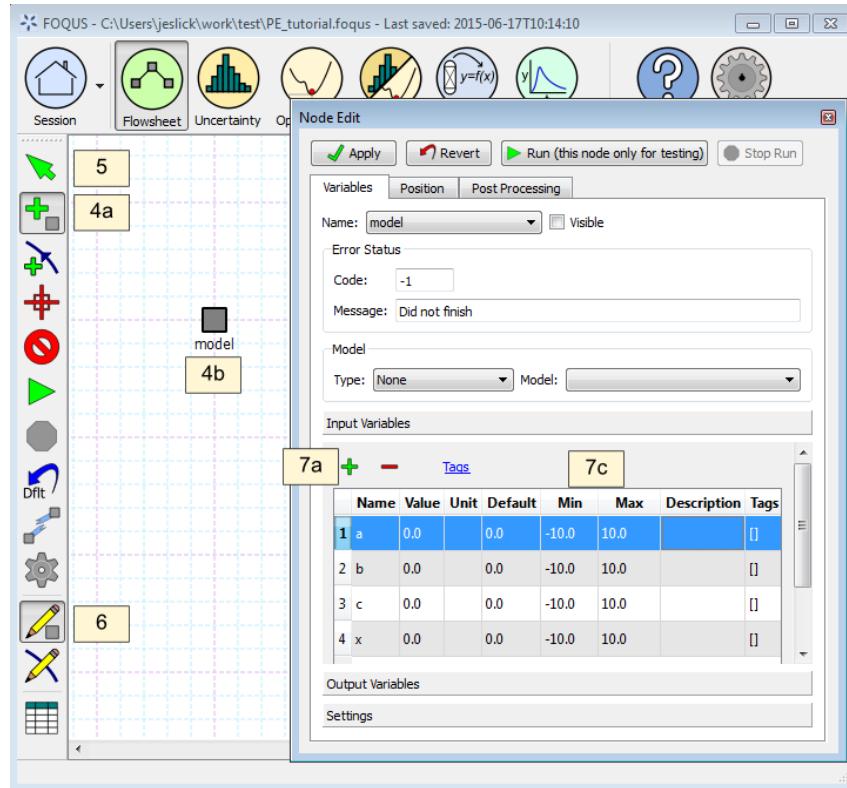


Figure 52: Adding Node and Inputs

8. Click Output Variables (see Figure 53).
9. Add the output variable y .
 - (a) Click the Add Output icon (see Figure 53).
 - (b) Enter “ y ” for the variable name in the Name column.

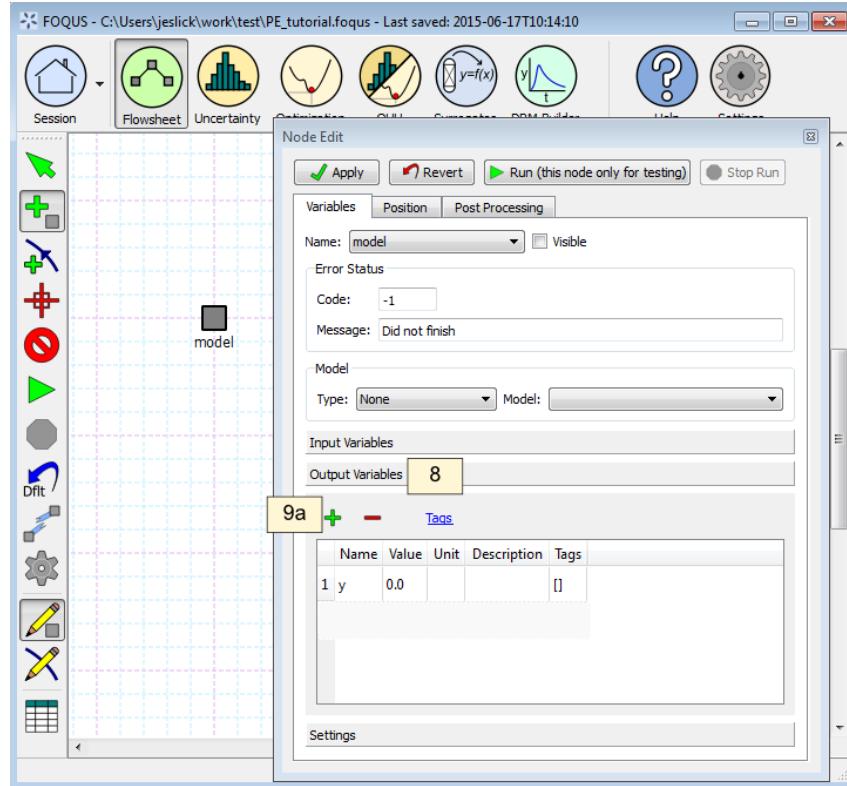


Figure 53: Adding Outputs

10. Add the model equation to the node.
 - (a) Click the Node Script tab.
 - (b) Enter the following code in the calculations box:

```
f['y'] = x['a']*x['x']**2 \
+ x['b']*x['x'] + x['c']
```

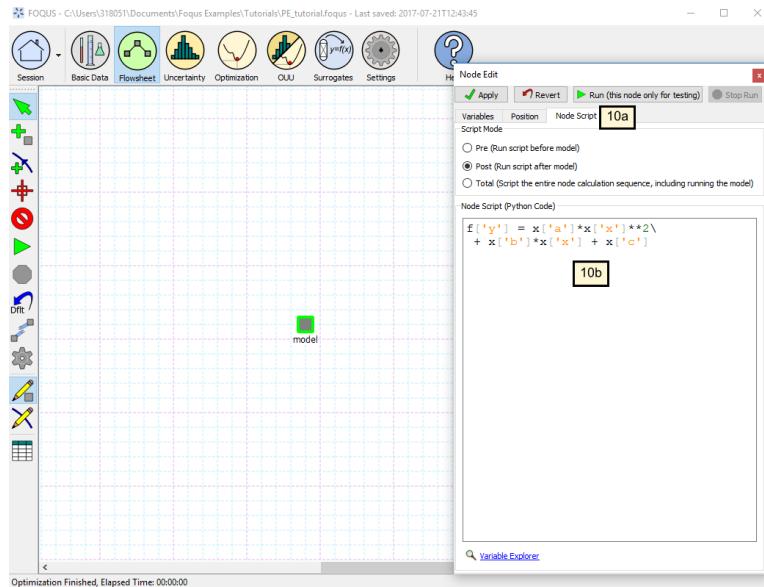


Figure 54: Adding Node Calculation

11. Return to the Output Variables table in the Node Editor, by clicking on the **Variables** tab, and selecting **Output Variables**.
12. Click **Run** in the left toolbar in the FOCUS Home window, to test a single flowsheet evaluation and ensure there are no errors.
13. When the run is complete, there should be no error and the value of y should be 7 in the Output Variables table.

The next step is to setup the optimization. The objective function is to minimize the sum of the squared errors between the estimated value of y and the observed value of y. There are five data points in Table 3, so there are five flowsheet evaluations that need to go into the calculation of the objective.

14. Click the **Optimization** button in the top toolbar of the Home window (see Figure 55).
15. Select “Decision” in the **Type** column drop-down lists for “model.a,” “model.b,” and “model.c.” The **Scale** column will automatically be set to linear.
16. Select “Sample” in the **Type** column drop-down lists for “model.x” and “model.ydata.”

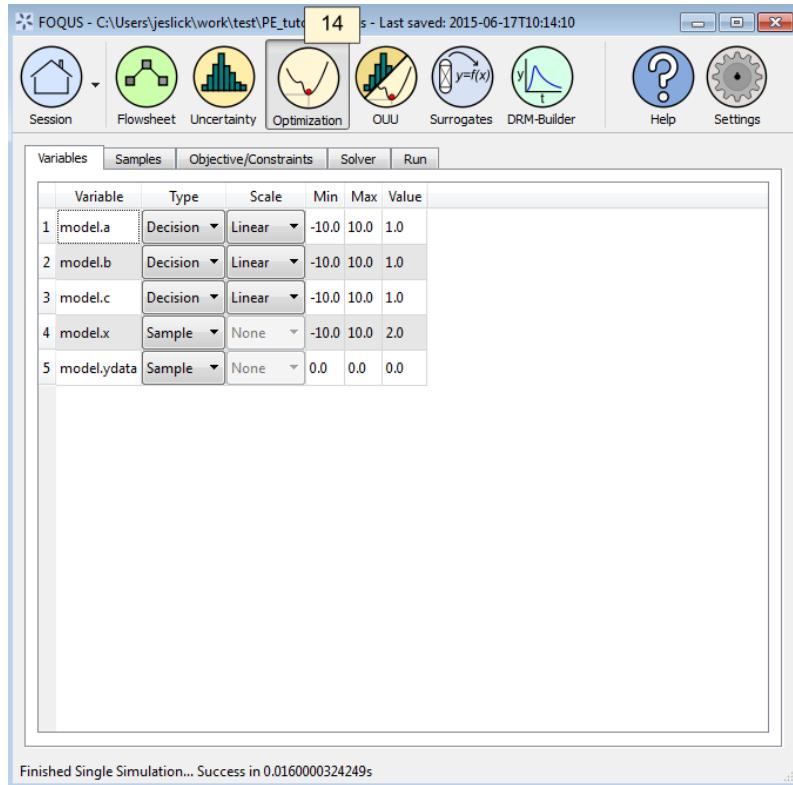
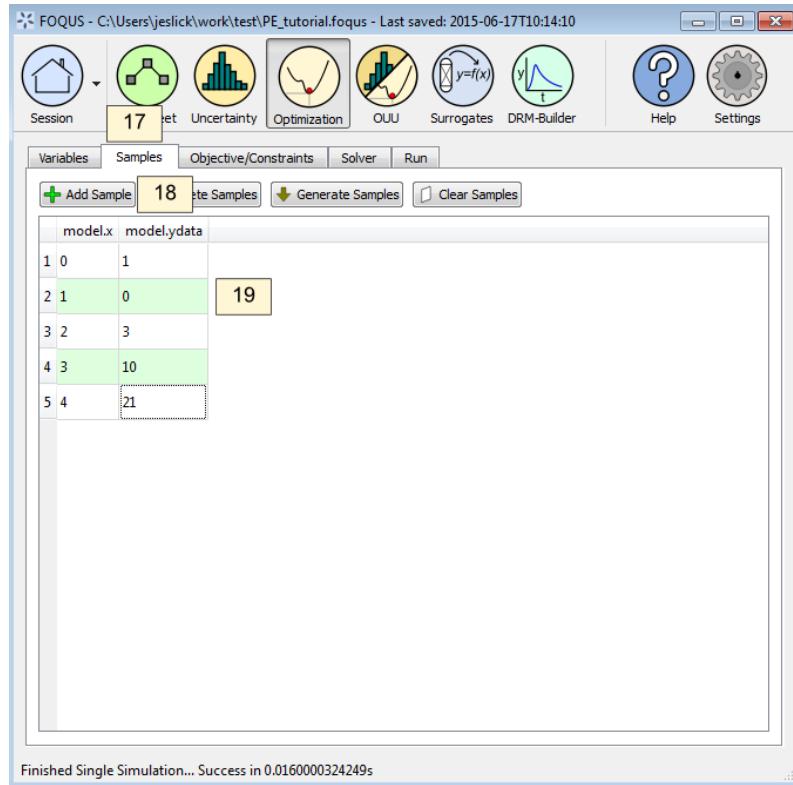


Figure 55: Optimization Variables

The decision variables in the optimization problem will be changed by the optimization solver to try to minimize the objective, and the sample variables are used to construct the samples that will go into the objective function calculation.

17. Select the **Samples** tab (see Figure 56).
18. Click **Add Sample** five times to add five samples.
19. Enter the data from Table 3 in the Samples table.
20. For larger sample sets, **Generate Samples** has an option to load from a CSV file.

**Figure 56: Optimization Samples**

The objective function is the sum of the square difference between y and ydata for each sample in Table 3. The optimization solver changes the a, b, and c to minimize the objective.

21. Click the Objective/Constraints tab.
22. Click the Add Objective icon on the right side of the Objective Function table (see Figure 57).
23. In the Expression column, enter the following (without the line break):

```
sum([(f[i]['model']['y'] - x[i]['model']['ydata'])**2
     for i in range(len(x))])
```

The above expression uses Python list comprehension to calculate the sum of squared errors. The keys for x and f are: sample index, node name, variable name, time step.

24. Enter 1 for the Penalty Scale.
25. Enter 100 for the Value for Failure.
26. No constraints are required.

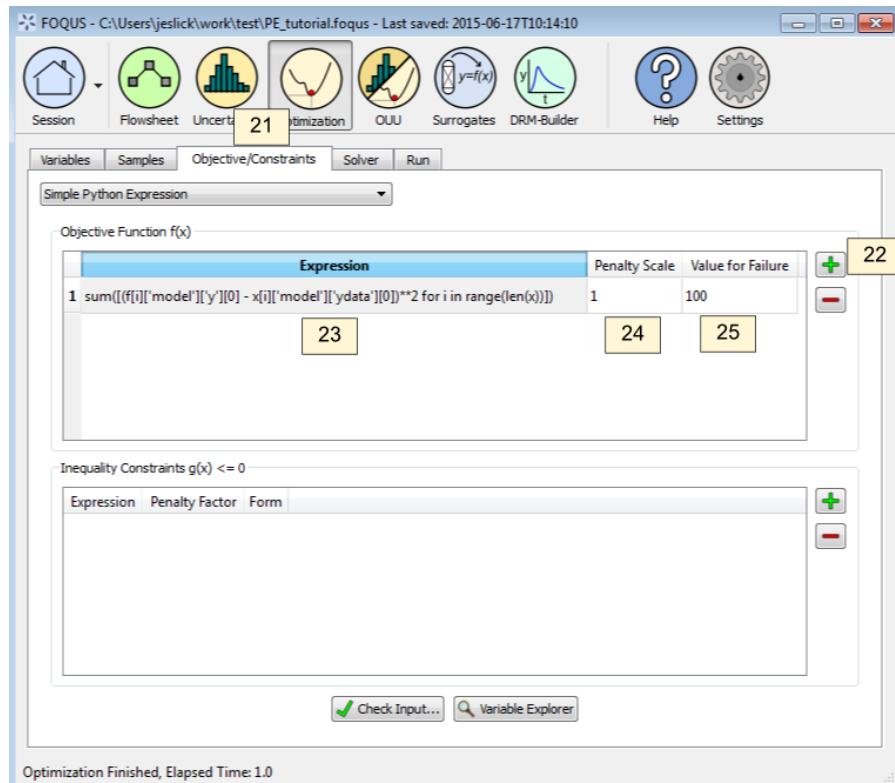
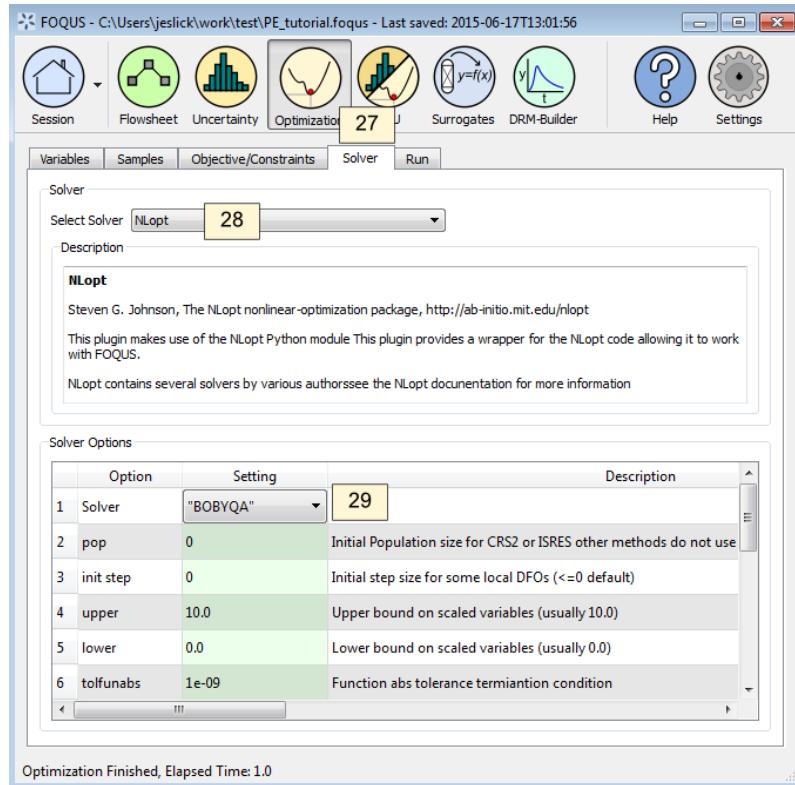


Figure 57: Objective Function

Once the objective is set up, a solver needs to be selected and configured. Almost any solver in FOCUS should work well for this problem with the default values.

27. Click the **Solver** tab (see Figure 58).
28. Select “NLOpt” from the **Select Solver** drop-down list. NLOpt is a collection of solvers that share a standard interface [8].
29. Select “BOBYQA” under the Solver Options table in the **Settings** column drop-down list.

**Figure 58: Optimization Samples**

30. Click the **Run** tab (see Figure 59).
31. Click the **Start** button.
32. The Optimization Solver Messages window displays the solver progress. As the solver runs, the best results found is placed into the flowsheet.
33. The **Best Solution Parallel Coordinate Plot** shows the scaled decision variable values for the best solution found so far.
34. The **Objective Function Plot** shows the value of the objective function as the optimization progresses.

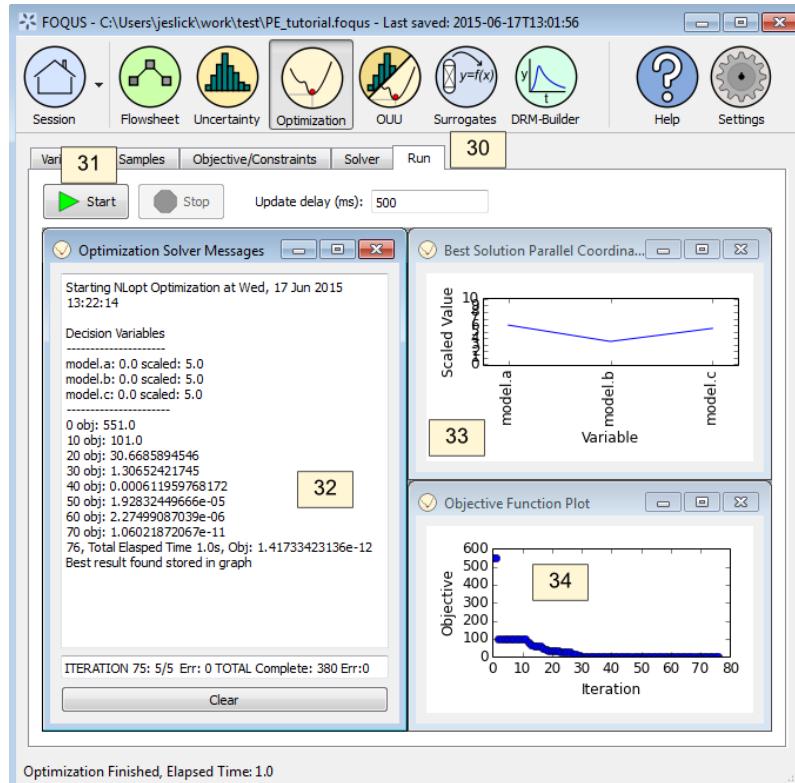


Figure 59: Running Optimization

The best result at the end of the optimization is stored in the flowsheet. All flowsheet evaluations run during the optimization are stored in the flowsheet results table.

35. Once the optimization has completed, click **Flowsheet** in the top toolbar.
36. Open the **Node Editor** and look at the **Input Variables** table. The approximate result should be $a = 2$, $b = -3$, and $c = 1$ (see Figure 60).

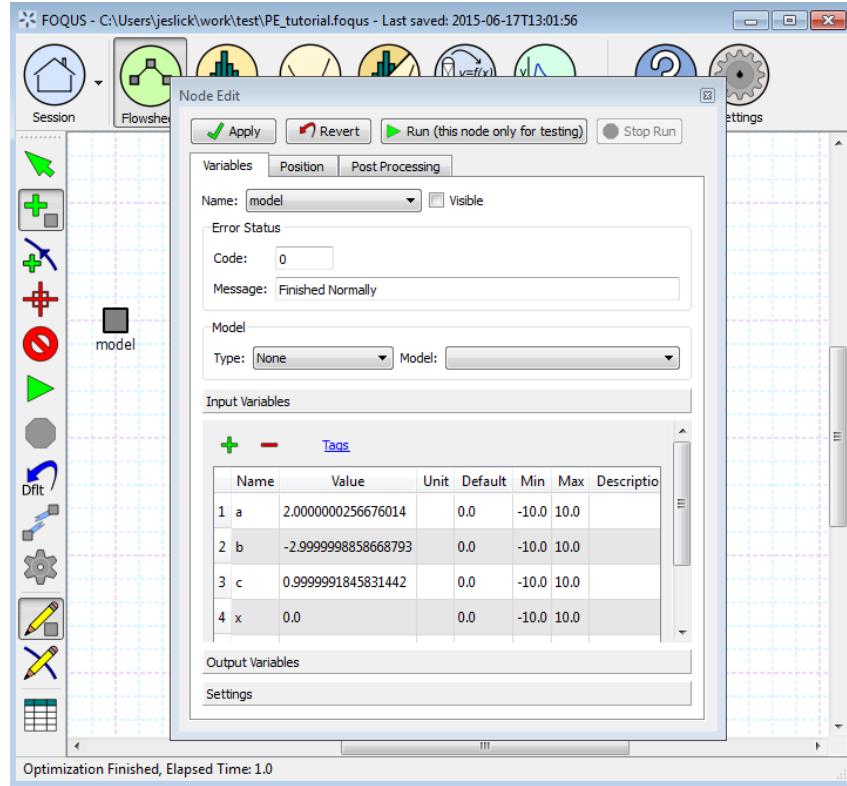


Figure 60: Flowsheet, Input Variables Results

4. UNCERTAINTY QUANTIFICATION

The Uncertainty Quantification (UQ) module of FOCUS provides a multitude of analysis and visualization capabilities to facilitate the understanding of uncertainty's impact on a given system. In a generic UQ study, the workflow is usually comprised of the following steps:

1. Define the objectives of the analysis.
2. Specify and acquire the simulation model, which implements an input-to-output mapping from inputs to outputs.
3. Select the inputs that have uncertainty and characterize said uncertainty in the form of *prior* distributions.
4. Identify relevant data from physical experiments that can be used to refine these prior distributions on the inputs.
5. Generate a set of input samples according to the input distribution.
6. Propagate the set of input samples through the simulation model to get the corresponding output values.
7. Analyze the results to make informed decisions about subsequent analyses.

FOCUS UQ provides tools to perform Steps 5-7. With respect to Step 7, a variety of analysis capabilities are available. They include parameter screening methods, response surface construction/validation/prediction, uncertainty analysis, sensitivity analysis, and visualization.

In this chapter, components of the UQ user interface are first explained, then the use of these components for UQ analyses is illustrated.

UQ User Interface

The UQ module enables the user to perform UQ studies on a flowsheet. From the Uncertainty button on the Home window, the user can configure different simulation ensembles (different sets of samples generated using different sampling schemes), run them, and perform analyses. This screen is illustrated in Figure 61.

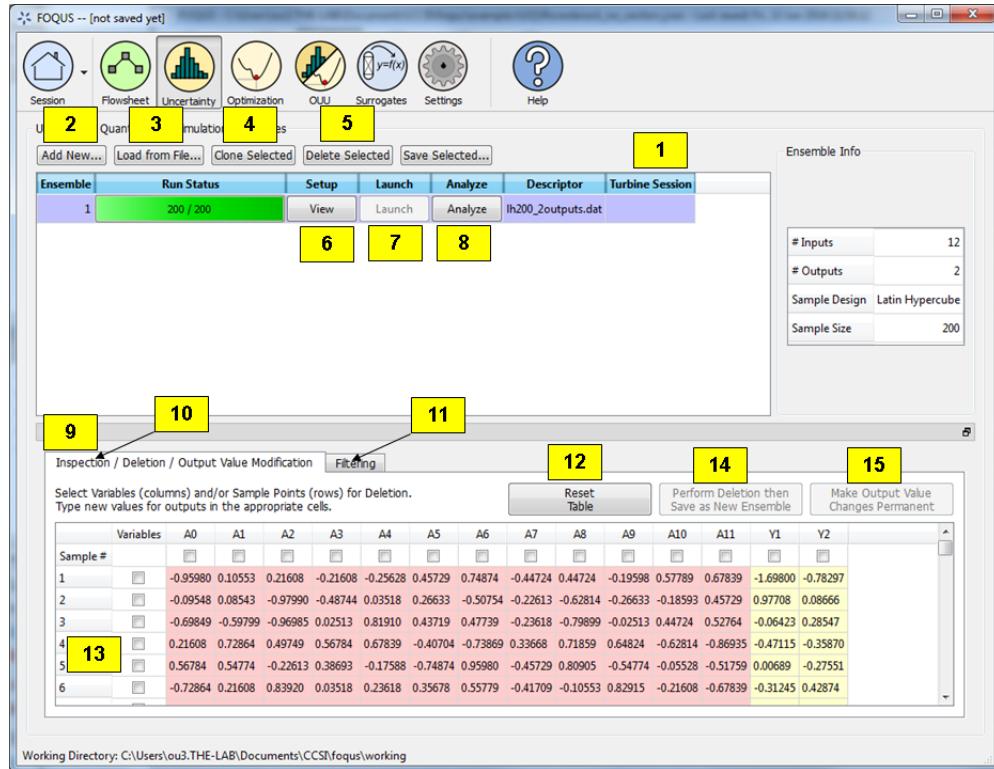


Figure 61: Uncertainty Quantification Screen

1. **Simulation Ensemble Table** displays all of the simulation ensembles: each ensemble being a row in the table. A simulation ensemble is a collection of sample points where each sample point has a different set of values for the uncertain variables. The values of these variables are generated based on the sampling scheme designated by the user. When launched, the output values of the sample points are calculated based on the generated sample input values. Subsequently, the corresponding simulation outputs can be analyzed. For each ensemble, the table displays the **Ensemble** index, **Run Status** (how many have been completed), **Setup** and **Launch** options (discussed below), and a **Descriptor**. The **Descriptor** contains the name of the corresponding node in the flowsheet or the name of the file if the ensemble was loaded from a file. Additional sample information such as **# Inputs**, **# Outputs**, **Sample Design**, and **Sample Size** are also displayed on the right.
2. **Add New** creates a simulation ensemble (a set of input samples) as a new row in the **Simulation Ensemble Table**. Once clicked, a dialog is displayed to prompt the user to choose between using (1) a flowsheet (an exact simulation model) or (2) a response surface (an approximate simulation model or an emulator) associated with the ensemble. If using an emulator, the user must browse a PSUADE-formatted file that contains the training data for the emulator (in this version, the response surface type has been designated inside the sample file and can only be changed by editing the sample file) and select the output(s) to be evaluated by the trained emulator. Subsequently, a simulation setup dialog box is displayed for setting up the distributions of input variables and the sampling scheme to generate samples of the uncertain input variables. This **Simulation Ensemble Setup** dialog

is explained in further detail in Section 4.1.1.

3. **Load from File** loads a simulation ensemble from a sample file that conforms to the PSUADE full file format. (See Section 4.3 for details on the PSUADE full file format.) The user can click **Save Selected** to save an existing ensemble as a PSUADE full file, and load it by clicking **Load from File** to perform further analyses.
4. **Clone Selected** clones the selected simulation ensemble and adds the copy as a new row at the end of the table. This ensemble can then be edited (e.g., depending on whether the ensemble has been run, the user has different options for modifying the ensemble). This allows the user to create a new ensemble similar to the current ensemble without having to start from scratch (i.e., setting up the input parameters). For example: (1) **Clone Selected** can be used in conjunction with **Load from File** to clone an existing ensemble before input/output modification to prepare a new but similar ensemble for UQ analysis. (2) **Clone Selected** can also be used to prepare a fresh ensemble for evaluation via a different simulation model. In this case, the user should save the cloned ensemble, reload it by clicking **Add New**, associate it with a node, and then click **Launch** to start the runs.
5. **Delete Selected** deletes the currently selected simulation ensemble.
6. **Revise** enables a user to change a simulation ensemble before launching the run. If the ensemble was previously run or it is cloned from an already-generated sample, the corresponding button becomes **View** so the user can view the input samples in the simulation ensemble.
7. **Launch** starts the simulation process of the ensemble. (**Launch** is not enabled until the user has setup everything needed for simulations.) A simulation is launched for each sample point to compute the corresponding outputs.
8. **Analyze**, when enabled (after all simulation results are ready), enables the user to perform various UQ analysis to the ensemble. When clicked, a new dialog box displays, allowing the user to configure and run analysis.
9. **Data Manipulation** enables (1) the deletion of inputs, outputs, or samples, (2) the modification of output values for specific sample points, and (3) the range-based filtering of samples.
10. **Inspection/Deletion/Output Value Modification** serves three purposes: it enables the user to (1) view the numerical values of samples in table form, (2) delete variables and/or samples, and (3) edit the output values of specific samples. **Deletion** creates a new simulation ensemble as a new row in the simulation table that contains only those inputs/outputs and samples that were not selected for deletion. **Output Value Modification** changes the values within the ensemble itself.
11. **Filtering** enables the user to filter samples based on the values of an input or output. First, select the ensemble to be filtered from the **Simulation Ensemble Table**. Once filtering is complete, a new simulation ensemble is added as a new row in the simulation table. The new simulation ensemble contains only those samples that satisfy the filtering criterion (with input or output samples within the specified range).
12. **Reset Table** resets the table to default, meaning all variable and sample selections are unselected and output values within the table are reverted back to their original values, thus undoing any edits to the table.

13. The table displays the values of inputs and outputs for each sample. Inputs are highlighted in pink; outputs are highlighted in yellow. The user can select which variables (columns) to delete by selecting the checkboxes on top. Likewise, the user can select which samples (rows) to delete by selecting the checkboxes on the left. Multiple samples can also be selected/deselected by using (1) Shift+Click or Ctrl+Click to select multiple rows, or (2) right-clicking to bring up a menu to check or uncheck the checkboxes corresponding to the rows of the selected samples. In addition, the user can change any output value by editing the appropriate cell. These modified cells are highlighted green until changes are made permanent by clicking the appropriate button.
14. **Perform Deletion then Save as New Ensemble** creates a new simulation ensemble as a new row in the **Simulation Ensemble Table**. The new ensemble is without the variables and samples that were previously selected for removal.
15. **Make Output Value Changes Permanent** overwrites the output values in the current ensemble with those that are highlighted green in the table.

The **Filtering** tab is illustrated in Figure 62 and enables the user to filter samples based on the values of an input or output.

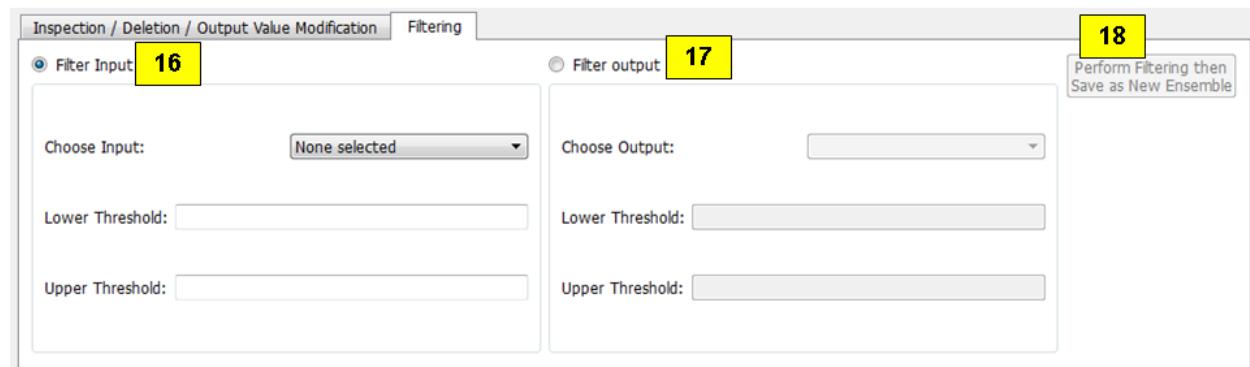


Figure 62: Filtering Tab

16. **Filter Input** filters based on the value of a certain input. Select an input variable as the filter target, and enter the lower and upper thresholds to specify a range of values to be kept in the new ensemble.
17. **Filter Output** filters based on the value of a certain output. Select an output variable as the filter target, and enter the lower and upper thresholds to specify a range of values to be kept in the new ensemble.
18. Once the filter settings are set, click **Perform Filtering then Save as New Ensemble** to apply the filter and create a new simulation ensemble.

The single-output **Analysis of Ensemble** dialog, which is displayed when **Analyze** is clicked for the selected ensemble, has two modes, as shown in Figure 63 and Figure 95.

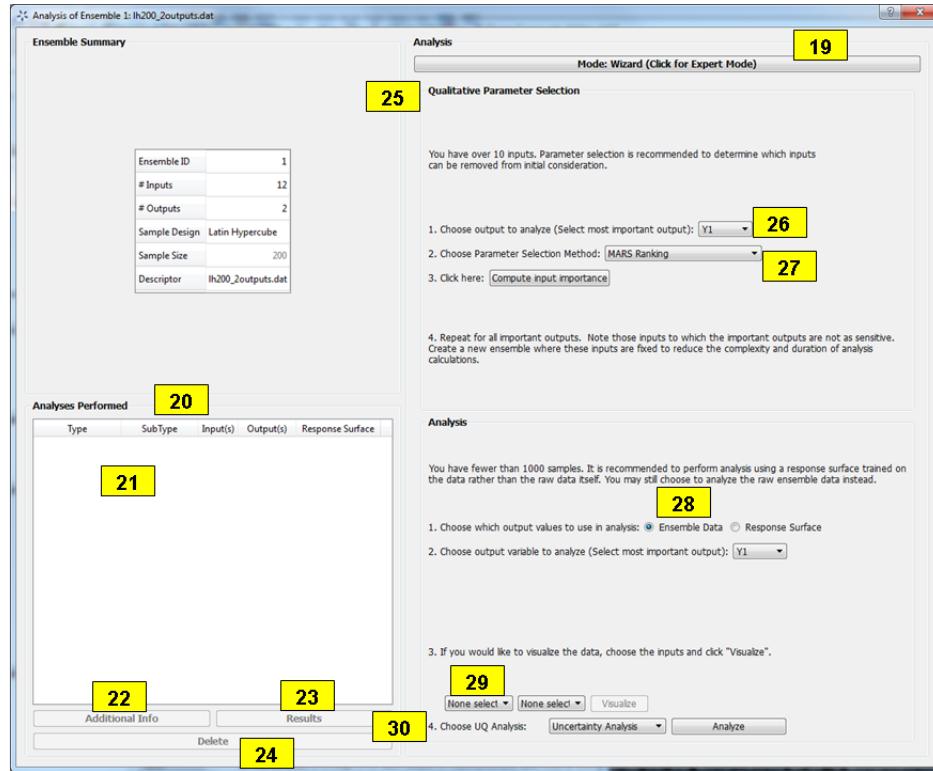


Figure 63: Analysis Dialog, Ensemble Data Analysis, Wizard Mode

19. Select **Wizard** or **Expert** mode. The **Wizard** mode provides more detailed guidance on how to perform UQ analysis. For users familiar with UQ analysis techniques, the **Expert** mode provides more functionality and flexibility but with less guidance on its use. For example, users will be able to customize the input distributions, as well as run more advanced uncertainty analysis that handles both epistemic and aleatory uncertainties.
20. The **Analyses Performed** section provides the user a history of previous analyses that were performed. The results of these analyses are cached, so the user can plot the analysis results without having to recompute them.
21. The **Analysis Table** populates as the user performs analyses. It lists previous analyses that the user has performed, along with some of the main analysis settings (analysis type, inputs and outputs analyzed, etc.)
22. Depending on the type of analysis performed, the **Additional Info** button displays any additional settings or parameters set by the user in the selected analysis that were not shown in the **Analysis Table**.
23. The **Results** button will display the results of the selected analysis.
24. The **Delete** button will delete the selected analysis from the history of previous analyses. Once deleted, the user will need to perform the analysis again to see its results.
25. The **Qualitative Parameter Selection** (top part of the **Analysis of Ensemble** dialog) houses the controls for parameter selection analysis. Parameter selection is a qualitative sensitivity analysis method that identifies a group of dominant input parameters that are recommended

for inclusion in subsequent UQ analyses, as they are the ones that most impact the output uncertainty. The parameter screening results are shown as bar graphs so that the user can rank the uncertain parameters visually.

26. Before performing parameter selection, the user must select a single output for identifying parameter sensitivities from the **Choose output to analyze** drop-down list.
27. There are several methods of parameter selection. The list of parameter selection methods available depends on the sample scheme of the selected ensemble. Select the appropriate method from the **Choose Parameter Selection Method** drop-down list. Then click **Compute input importance** to start the analysis.
28. The **Ensemble Data** radio button directs FOCUS to perform analyses on the raw ensemble data.
29. To view plots of the raw ensemble data, choose the desired input(s) from the **Select the input(s)** drop-down lists. Then click **Visualize**. If multiple inputs are selected, each must be unique.
30. To perform an analysis, select the desired analysis (“Uncertainty Analysis” or “Sensitivity Analysis”) from the **Choose UQ Analysis** drop-down list. Uncertainty Analysis computes and displays the probability distribution of the single selected output parameter and displays its sufficient statistics, such as mean, standard deviation, skewness, and kurtosis. Sensitivity Analysis computes and displays each uncertain input parameter’s contribution to the total variance of the output. If Sensitivity Analysis is selected, choose the type of sensitivity analysis desired in the next drop-down list. There are three options for Sensitivity Analysis: (1) first-order, (2) second-order, and (3) total-order.
 - First-order analysis examines the effect of varying an input parameter alone.
 - Second-order analysis examines the effect of varying pairs of input parameters.
 - Total-order analysis examines all interactions’ effect of varying an input parameter alone and as a combination with any other input parameters.

Click **Analyze** to run the analysis. (Note: Raw ensemble data analysis may not be suitable if the sample size is small. It may be useful if the data set has tens of thousands of sample points or if an adequate response surface cannot be constructed. Otherwise, response surface-based analyses are recommended.)

31. **Response Surface** enables the user to perform all analyses related to response surfaces. A response surface is an approximation of the input-to-output relationship. This is an inexpensive way to approximate the values of outputs given different input values when the actual simulation of output values is computationally intensive. FOCUS uses the data (i.e., input-output samples) to fit a response surface scheme. The first step in this analysis is to select which output to analyze.
32. Select the **Response Surface Model** to be used to approximate the input-to-output mapping. Selection of “Polynomial” or “MARS” requires one further selection in the second drop-down list. If “Polynomial” is chosen in the first drop-down list and “Legendre” is chosen in the second drop-down list, the user needs to specify a number for the **Legendre polynomial order** before analysis can proceed.

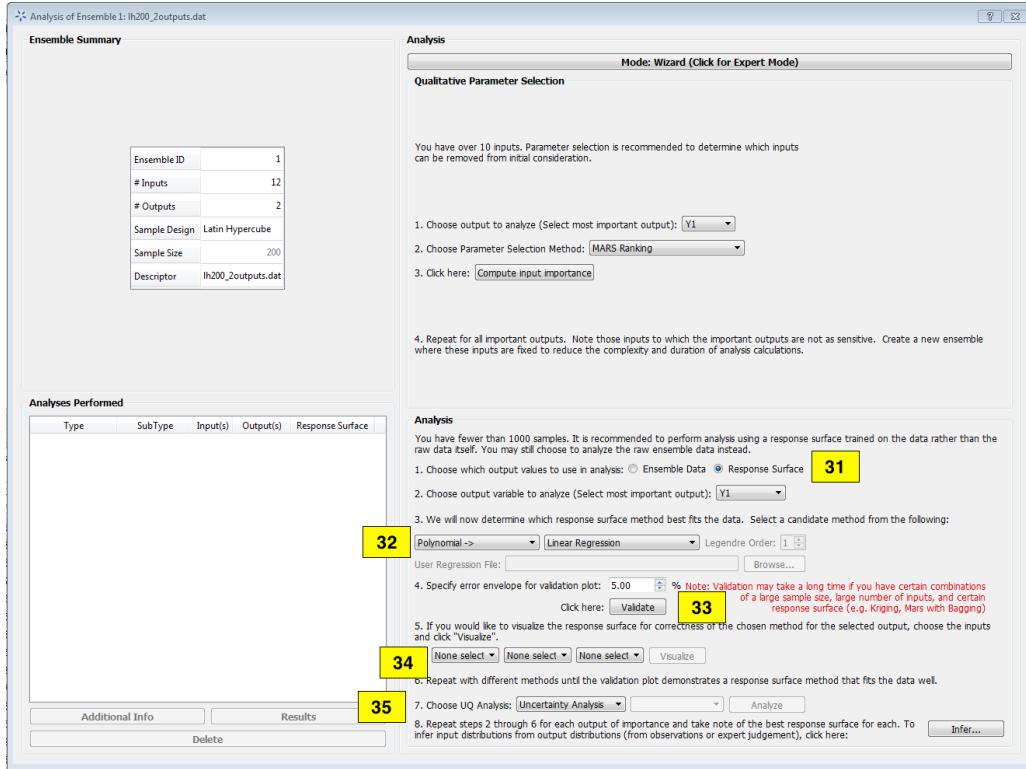


Figure 64: Analysis Dialog, Response Surface Analysis, Wizard Mode

33. The response surface selected must be validated before further analyses can be performed. The user can specify the error envelope for the validation plot. When Validate is clicked, the resulting plots display the best fit between the response surface (based on the model selected) and the actual data.
34. **Choose UQ Analysis** enables the user to perform response-surface-based UQ analyses. Select the analysis in the first drop-down list. If the desired analysis is Sensitivity Analysis, select the desired type of sensitivity analysis in the second drop-down list and then click **Analyze**. Uncertainty Analysis and Sensitivity Analysis compute and display the same quantities as in item #30. However, the results displayed are based on samples drawn from the trained response surface, not the simulation ensemble itself. Moreover, if the selected response surface has uncertainty, the resulting plots also reflect this uncertainty information.
35. FOQUS also provides visualization capabilities, enabling the user to view the response surface as a function of one or multiple inputs. Up to three inputs can be visualized at once. Click Visualize to view. A 2-D line plot displays if only one input parameter is selected. A 3-D surface plot and a 2-D contour plot display if two input parameters are selected. A 3-D isosurface plot with a slider bar displays if three input parameters are chosen. For the iso-surface plot, the user can use the slider to selectively display the 3-D input parameter space that activates a particular range in the output parameter.

Finally, the **Bayesian Inference of Ensemble** dialog (shown in Figure 65) is used to calculate the posterior distributions (prior distributions integrated with data) of the uncertain input parameters.

Inference utilizes Markov Chain Monte Carlo (MCMC) to compute the posterior distributions, using response surfaces that serve as fast approximations to the actual simulation model.

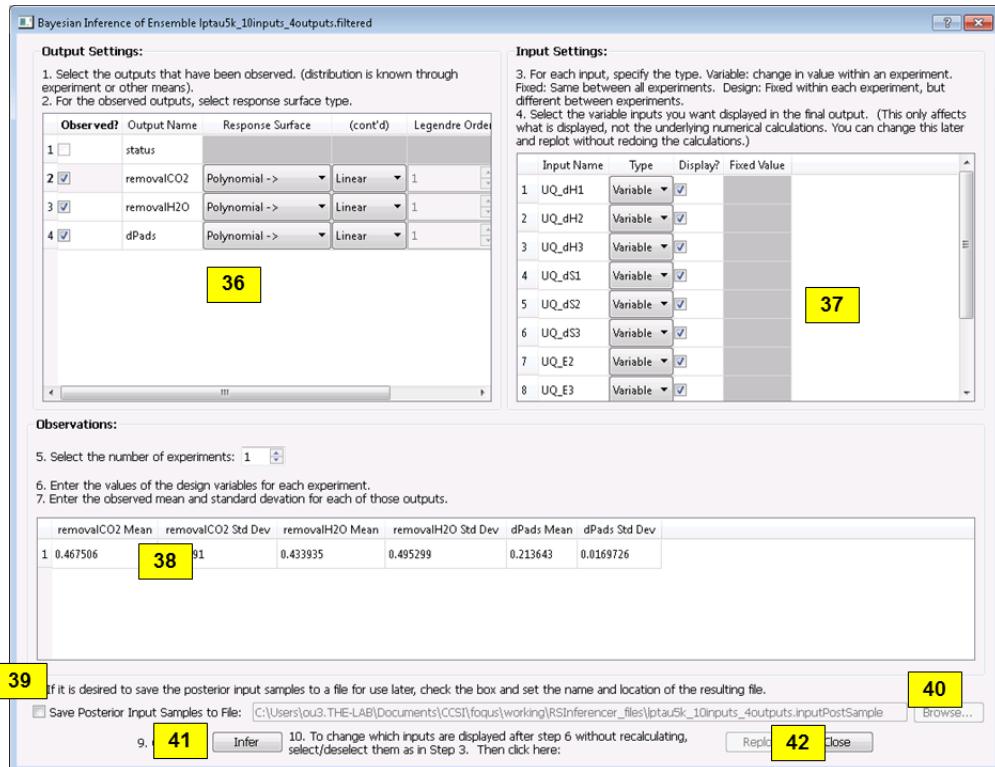


Figure 65: Bayesian Inference Dialog

36. Inference uses a response surface to approximate the input-to-output mapping. In **Output Settings**, select the observed outputs and select the response surface type that works best with each observed output. As in item (32), further selections may be required based on the response surface chosen. The simulation ensemble is used as the training data for generating the response surfaces.
37. The user can specify which inputs are fixed, design (fixed per experiment, but changes between experiments), or variable using the **Input Settings Table**. In addition, the user can specify which inputs are displayed in the resulting plots of the posterior distributions. By default, once inference completes, all inputs will be displayed in the plots. To omit specific inputs, clear the checkboxes from the **Display** column of the table. Finally, in **Expert** mode, this table can also be used to modify the input prior distributions. The default prior is the input distribution specified in the simulation ensemble. To change the prior distribution type, use the drop-down list in the **PDF** column and enter corresponding values for the PDF parameters. To change the range of a uniform prior, scroll all the way to the right to modify **Min/Max**.
38. The **Observations** section enables the user to add experimental data in the form of observations of certain output variables. At least one observation is required. Currently, the observation noise model is assumed to be a normal distribution. Other distributions may be

supported in the future. To specify the observation noise model, enter the mean (and standard deviation, if standard inference is selected) for each output observation. For convenience, the **Mean** and **Standard Deviation** fields have been populated with the statistics from the ensemble uncertainty analysis. If any inputs are selected as design inputs, their values will also be required here.

39. **Save Posterior Input Samples to File** checkbox, when selected, saves the posterior input samples as a PSUADE sample file (format described in Section 4.3). This file characterizes the input uncertainty as a set of samples, which can be re-used in the **Simulation Ensemble Setup** dialog, to evaluate the outputs corresponding to these posterior input samples.
40. If saving posterior samples to a file, click **Browse** to set the name and location of where this file is saved.
41. Click **Infer** to start the analysis. (Note: If the inference returns an invalid posterior distribution (i.e., one with no samples), it usually means the prior distributions or that the observation data distributions are not prescribed appropriately. In this case, it is recommended that the user experiment with different priors and/or data distribution means and/or standard deviations.)
42. Inference calculations often take a very long time. If inference has run to completion, use **Replot** to generate new plots (e.g., to only display a subset of the input posterior graphs) from the cached inference results.

Simulation Ensemble Setup Dialog

The **Simulation Ensemble Setup** dialog (shown in Figure 66) is used to create a new simulation ensemble. This is done by: (1) setting up distribution parameters and generating samples, or (2) loading samples from a file. This dialog is displayed when selecting **Add New** on the UQ window (Figure 61).

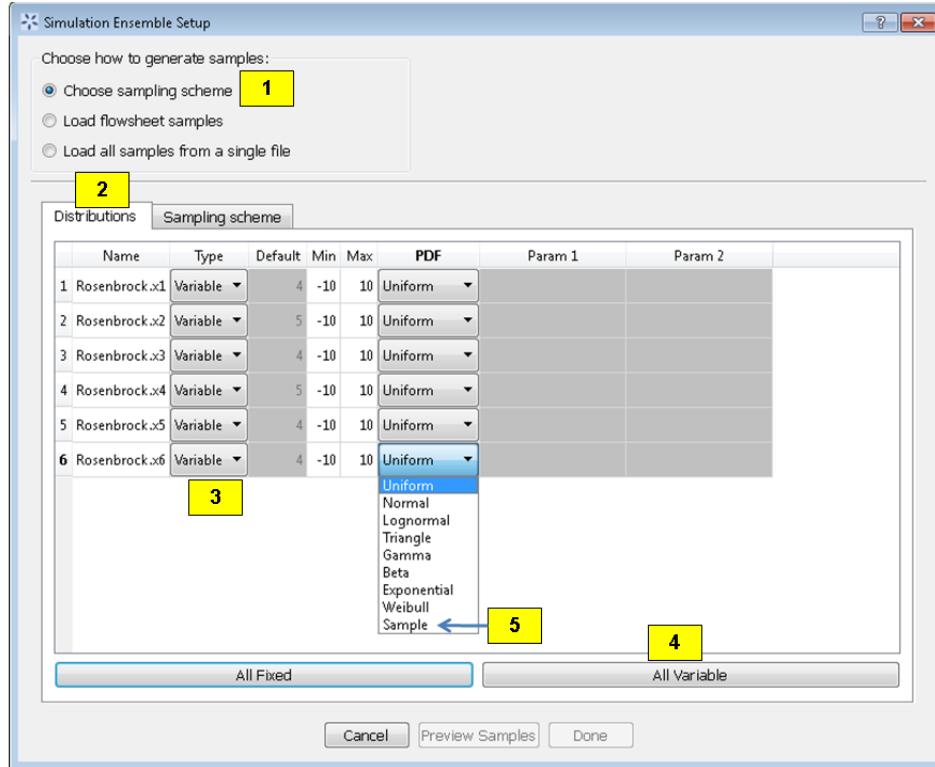


Figure 66: Simulation Ensemble Setup Dialog, Distributions Tab

1. Choose how to generate samples. There are three options: (1) Choose sampling scheme (default), (2) Load flowsheet samples, or (3) Load all samples from a single file. The option 3 is explained in item (11).
2. If Choose Sampling Scheme is selected, the Distributions tab is displayed. The user specifies the input uncertainty information.
3. The Distributions Table is pre-populated with input variable information gathered from the flowsheet node. Under the Type column drop-down list, the user can select “Fixed” or “Variable”. Selecting “Fixed” means that the input is fixed at its default value for all the samples. Changing the type to “Variable” means that the input is uncertain; therefore, its value varies between samples. With any fixed input, the only parameter that can be changed is the Default value (i.e., all samples of this input are fixed at this default value). With any variable input, the Min/Max values, as well as the probability distribution function (PDF), for that input can be changed. Some PDFs have their own parameters (e.g., mean and standard deviation for a normal distribution), which are required in the columns right of the distribution column. See the PSUADE manual for more details on the different PDFs.
4. All Fixed and All Variable are convenient ways to set all the inputs to variable or fixed.
5. Note: A “Sample” PDF refers to sampling with replacement (i.e., input samples would be randomly drawn, with replacement, from a sample file). If the selected distribution for any input is “Sample”, then the following parameters are required: (1) the path of the sample file (which must conform to the sample format specified in Section 4.3); (2) the output index that designates which output is to be used.

6. In the **Sampling scheme** tab (Figure 67), specify the sampling scheme, the sample size, and perform sample generation.

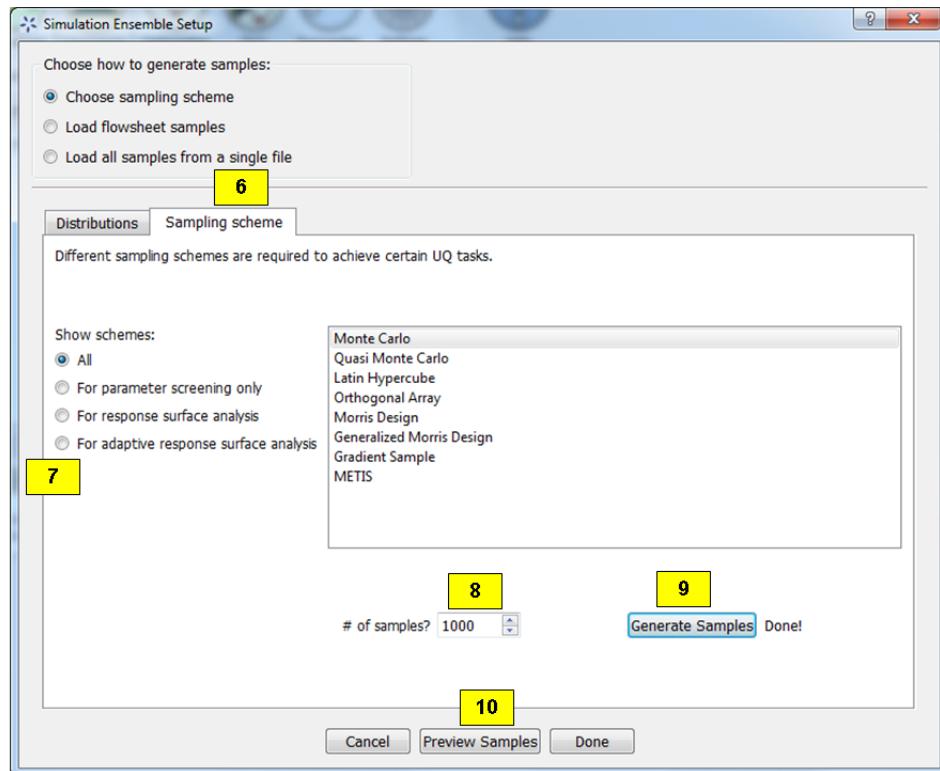


Figure 67: Simulation Ensemble Setup Dialog, Sampling Scheme Tab

7. Each radio button displays a different list of sampling schemes on the right. The radio buttons serve as a guide to help in the selection of the appropriate sampling schemes for target analyses. A sampling scheme must be selected from the list on the right to proceed.
8. Set the number of samples to be generated from the # of samples spinbox.
9. When all parameters are set, click **Generate Samples**. This generates the values for all the input variables, based on the sampling scheme selected.
10. Once samples have been generated, click **Preview Samples** to view the samples that were generated. This displays the sample values in table form, as well as graphically as a scatter plot.
11. From item (1), if the user elects to load all samples from a single file, click **Browse** to select the file containing the samples (Figure 68). This file must conform to the PSUADE full file format, the PSUADE sample format, or CSV file (all formats described in Section 4.3). Note: This is the only place where all the formats are supported. Once the file is loaded, the file name displays in the text box. These samples can now be used in the same way as an ensemble that was newly generated (as described above).

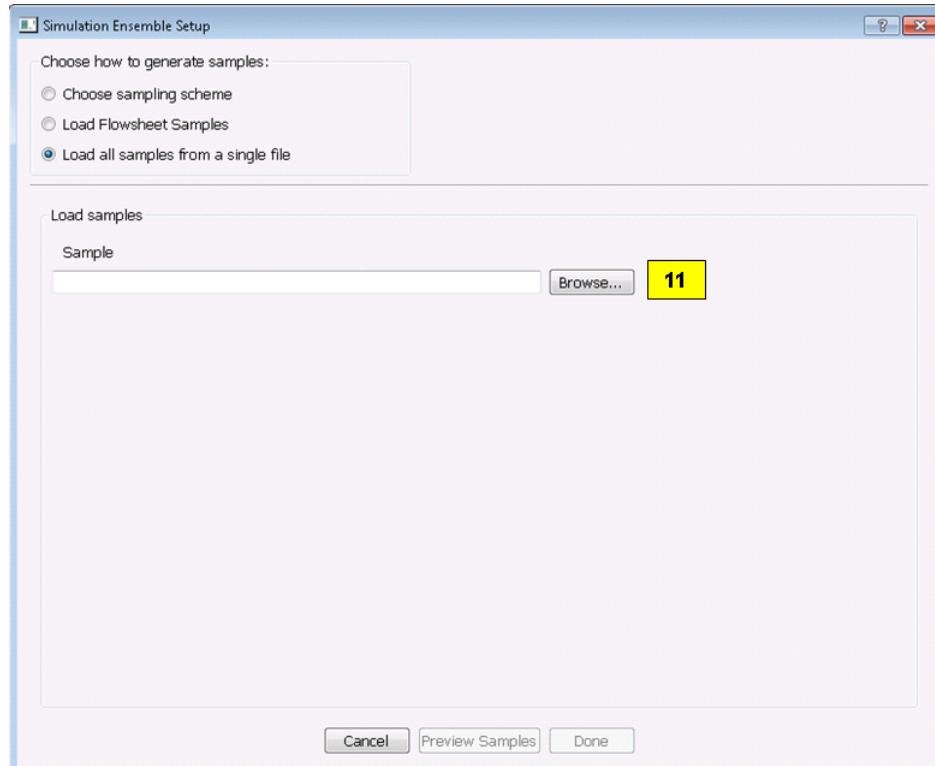


Figure 68: Simulation Ensemble Setup Dialog, Load Samples Option

UQ Tutorial

This section contains five tutorials that illustrate the use of FOQUS UQ to facilitate the UQ workflow discussed above.

Each tutorial will refer to example files that are typically located at:
C:\Program Files (x86)\foqus\foqus_2015.10.00\examples.

Simulation Ensemble Creation and Execution

In this tutorial, a simulation ensemble is created and run.

1. From the FOQUS main screen, click the **Session** button and then select **Open Session** to open a session. Browse to the examples folder, go into the UQ subfolder, and then select the “Rosenbrock_novectors.foqus” problem (Figure 69).

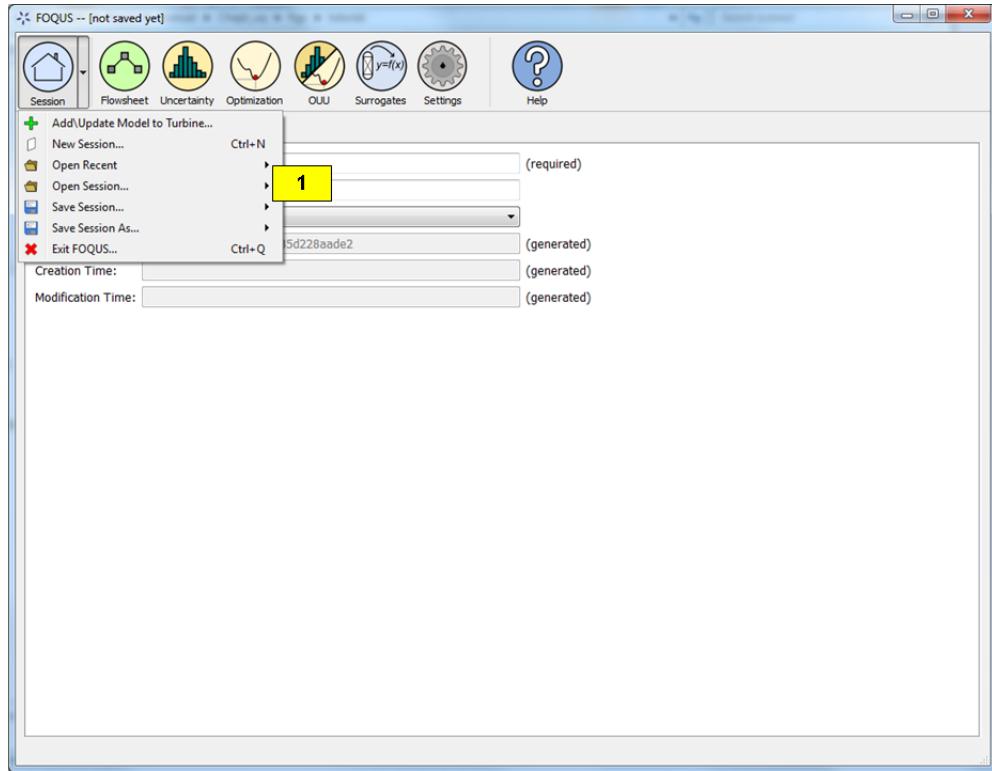


Figure 69: Home Screen

2. Opening this file loads a session that has a flowsheet with one node (Figure 70). See Section 2.8 for a detailed example of creating a flowsheet.

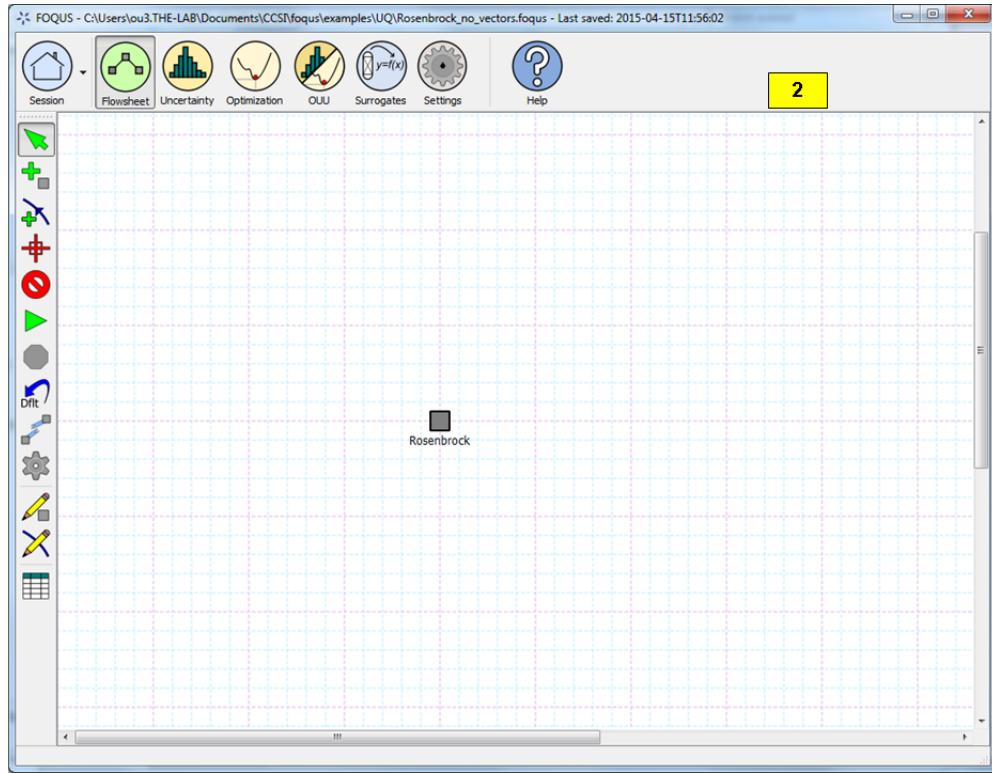


Figure 70: Flowsheet for Rosenbrock Example

3. Click the Uncertainty button (Figure 71).

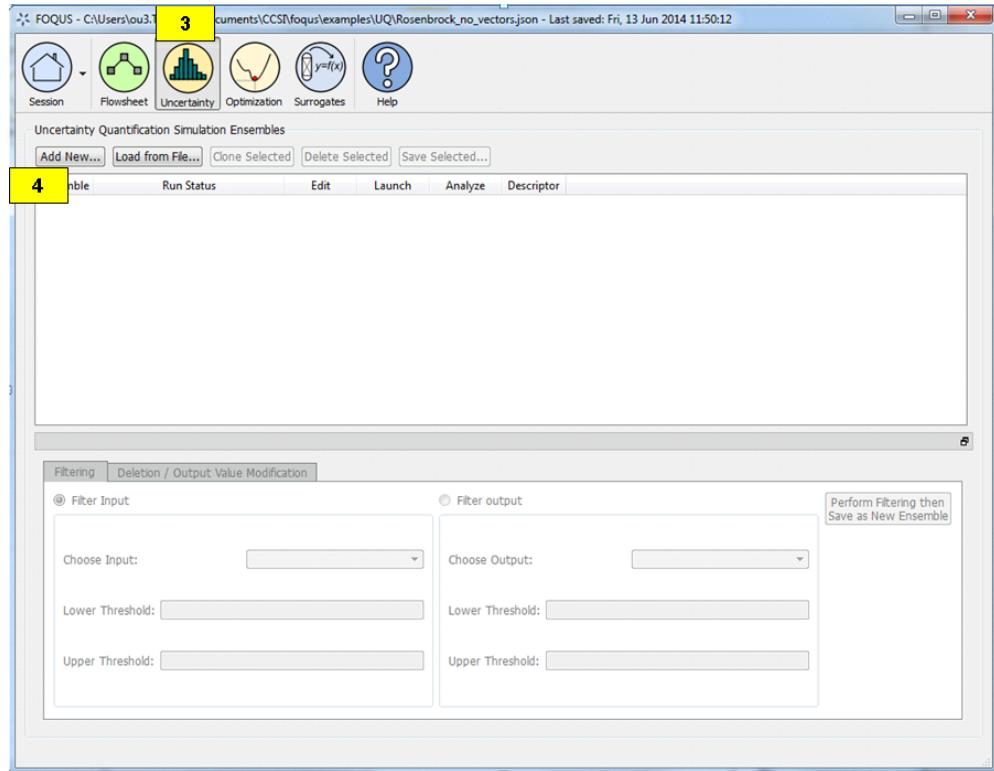


Figure 71: Uncertainty Quantification Screen

4. Click Add New to create a new simulation ensemble.
5. The Add New Ensemble dialog displays (Figure 72). The “Use flowsheet” option should be enabled.
6. *This item describes additional features and is provided for information only. It is not intended to be followed as part of the step-by-step tutorial.*

An alternative is to use an emulator by selecting “Use emulator.” This alternative is preferred if the actual simulation model is too computationally expensive to be practical for a large number of samples. This option enables the user to trade off accuracy for speed by training a response surface to approximate the actual simulation model. If this option is selected (Figure 73), the user needs to provide a training data file containing a small simulation ensemble generated from the actual simulation model. This training data file is to be in the PSUADE full file format (Section 4.3).

- Click **Browse** and select the training data file with which to train the response surface. The inputs, outputs and response surface type is read from the training data and populated accordingly on this dialog box.
- **Select Output(s) of Interest.** To select multiple outputs, the user can use Shift + Click to select a range, or use Ctrl + Click to select/deselect individual outputs.

7. Click **OK**.

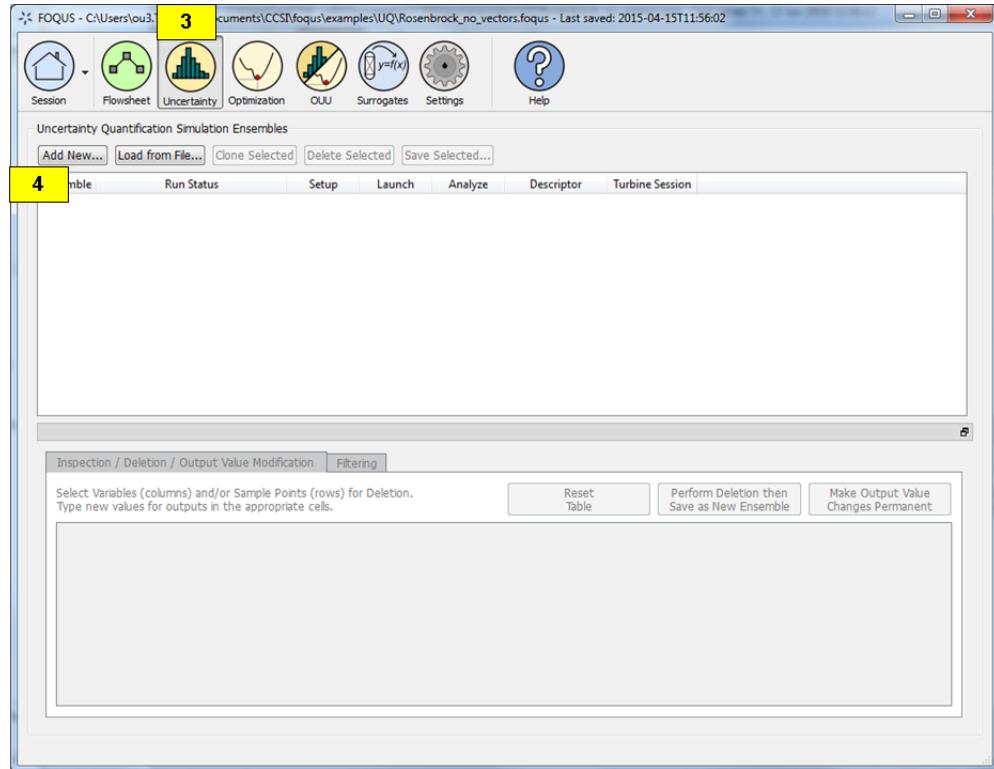


Figure 72: Add New Ensemble Dialog, Flowsheet Option

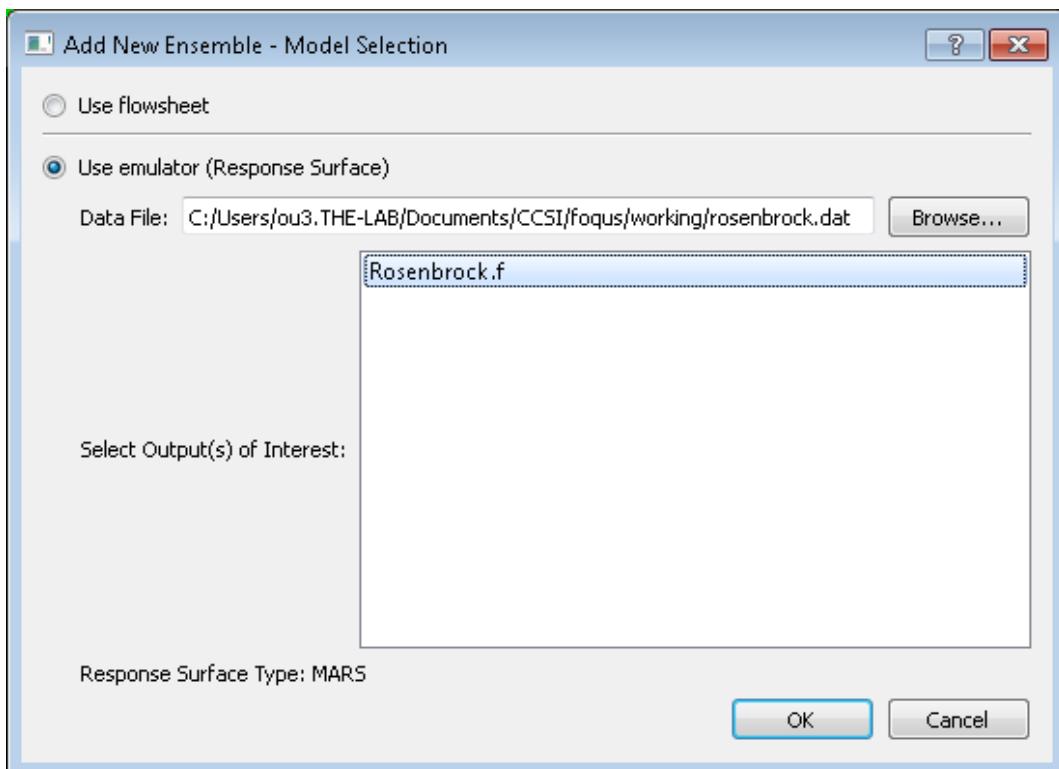


Figure 73: Add New Ensemble Dialog, Emulator Option

8. This displays the **Simulation Ensemble Setup** dialog box (Figure 74) that prompts the user for options specific to the creation of input samples.
9. Within the **Distributions** tab, the **Distributions Table** has all the inputs from the flowsheet node, each displayed in its own row.
 - (a) Click the **All Variable** button.
 - (b) Change the **Type** of “x2” to “fixed.”
 - (c) Enter 5 into the **Default** column for “x2.”

Subsequently, other cells in the row are enabled or disabled according to the type selection.

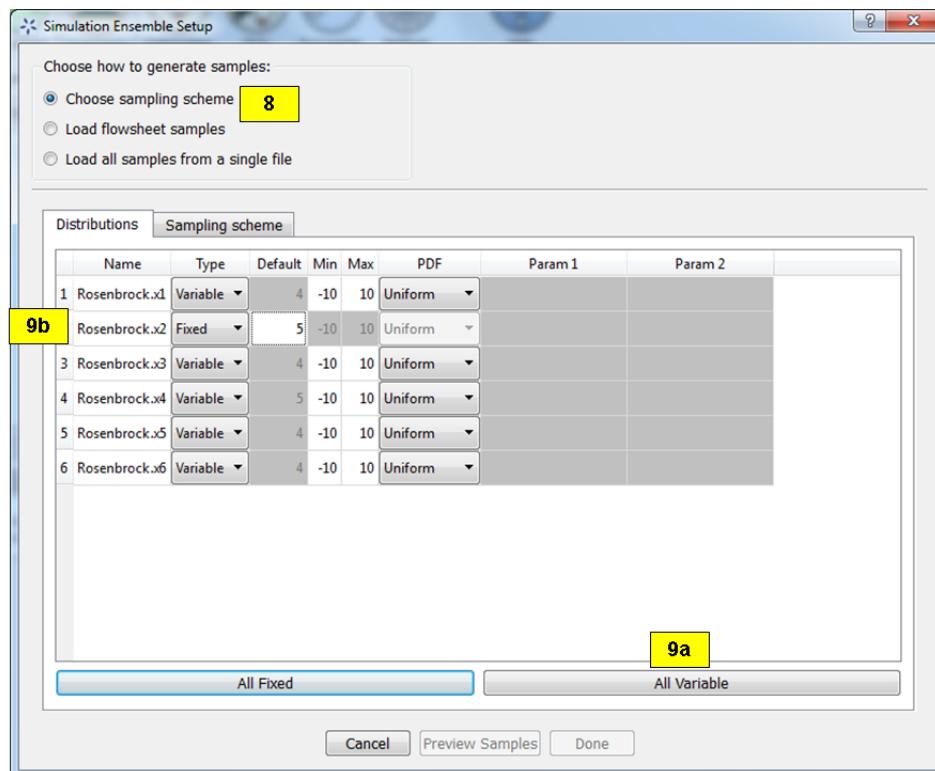


Figure 74: Simulation Ensemble Setup Dialog, Distributions Tab

In this dialog, extra options that are available related to simulation ensemble setup are discussed.

- Change the PDF of “x6” by exploring the drop-down list in the **PDF** column of the **Distributions Table**. The drop-down list is denoted by box (9c) in Figure 75. If any of the parametric distributions are selected (e.g., “Normal”, “Lognormal”, “Weibull”), the user is prompted to enter the appropriate parameters for the selected distribution. If non-parametric distribution “Sample” is selected, the user needs to specify the name of the sample file (a CSV or PSUADE sample format is located in Section 4.3) that contains samples for the variable “x6.” The user also needs to specify the output index to indicate which output in the sample file to use. The resulting simulation ensemble would contain “x6” samples that are randomly drawn (with replacement) from the samples in this file.

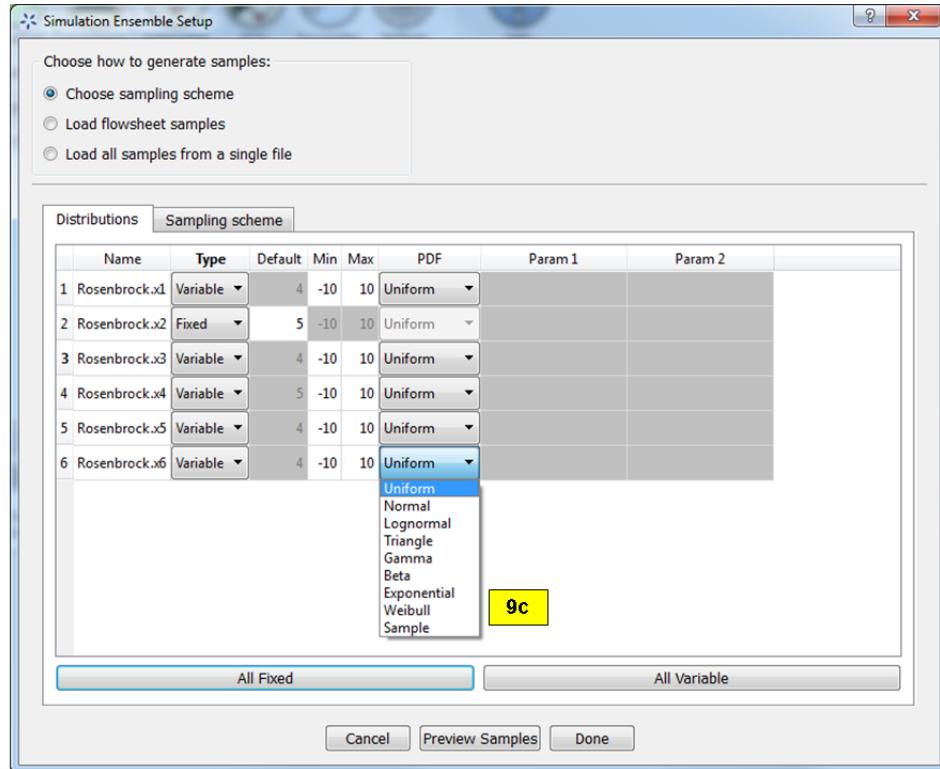


Figure 75: Simulation Ensemble Setup Dialog, Distributions Tab, PDF Selection

- Alternatively, select **Choose sampling scheme** (box (8) of Figure 74), and try selecting “Load all samples from a single file.” With this selection, a new dialog box (Figure 68) prompts the user to browse to a PSUADE full file, a PSUADE sample file, or CSV file (all formats are described in Section 4.3) that contains all the samples for all the input variables in the model.

Both of these options offer the user additional flexibility with respect to characterizing input uncertainty or generating the input samples directly.

10. Once complete, switch to the **Sampling Scheme** tab (Figure 76).

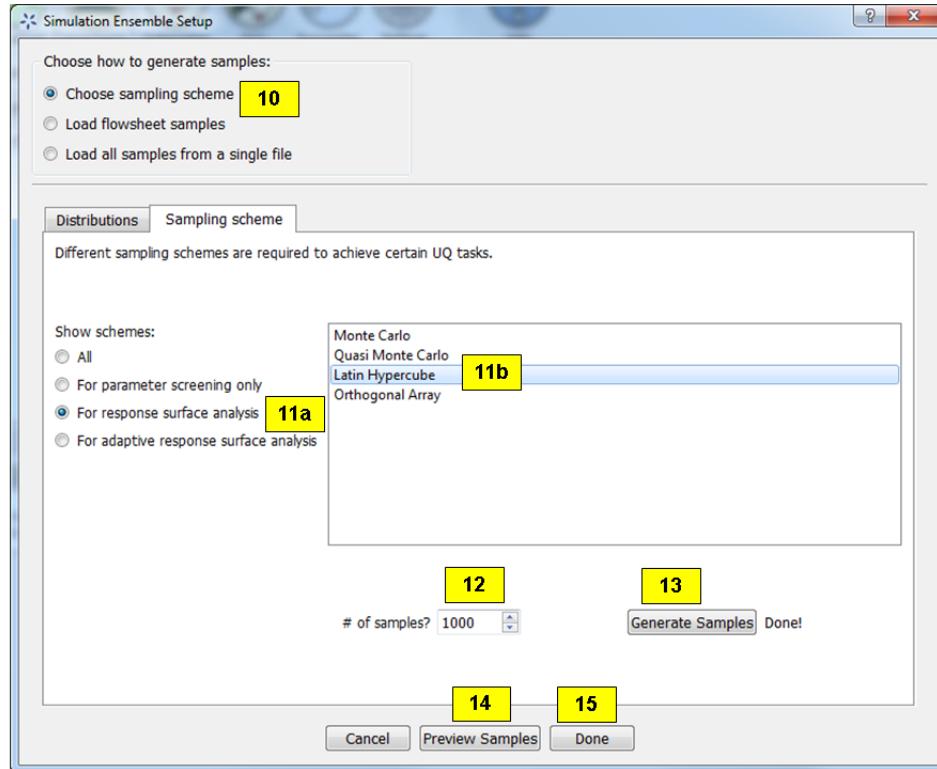


Figure 76: Simulation Ensemble Setup Dialog, Sampling Scheme Tab

11. Select a sampling scheme with the assumption that the user is unsure which sampling scheme to use, but wants to perform some kind of response surface analysis. This example helps the user find a suitable one.
 - (a) Click **For response surface analysis**. Note the list on the right changes accordingly.
 - (b) Select “Latin Hypercube” from the list on the right.
12. To generate 500 samples, change the value in “# of samples.” Some sampling schemes may impose a constraint on the number of samples. If the user has entered an incompatible sample size, a pop-up window displays with guidance on the recommended samples size.
13. Click **Generate Samples** to generate the sample values for all the variable input parameters. On Windows, if the user did not install PSUADE in its default location (C:\Program Files (x86)\psuade_project 1.7.1\bin\psuade.exe) and the user did not update the PSUADE path in FOCUS settings (refer to Section 2.2.2), then the user is prompted to locate the PSUADE executable in a file dialog.
14. Once the samples are generated, the user can examine them by clicking **Preview Samples**. This displays a table of the values, as well as the option to view scatter plots of the input values. The user can also select multiple inputs at once to view them as separate scatter plots on the same figure.
15. When finished, click **Done**.
16. The simulation ensemble should be displayed in the **Simulation Ensemble Table**. If the user would like to change any of the parameters and regenerate a new set of samples, simply

click the **Revise** button.

17. Next, calculate the output value for each sample. Click **Launch**. The user should see the progress bar quickly advance, displaying the status of completed runs (Figure 77).

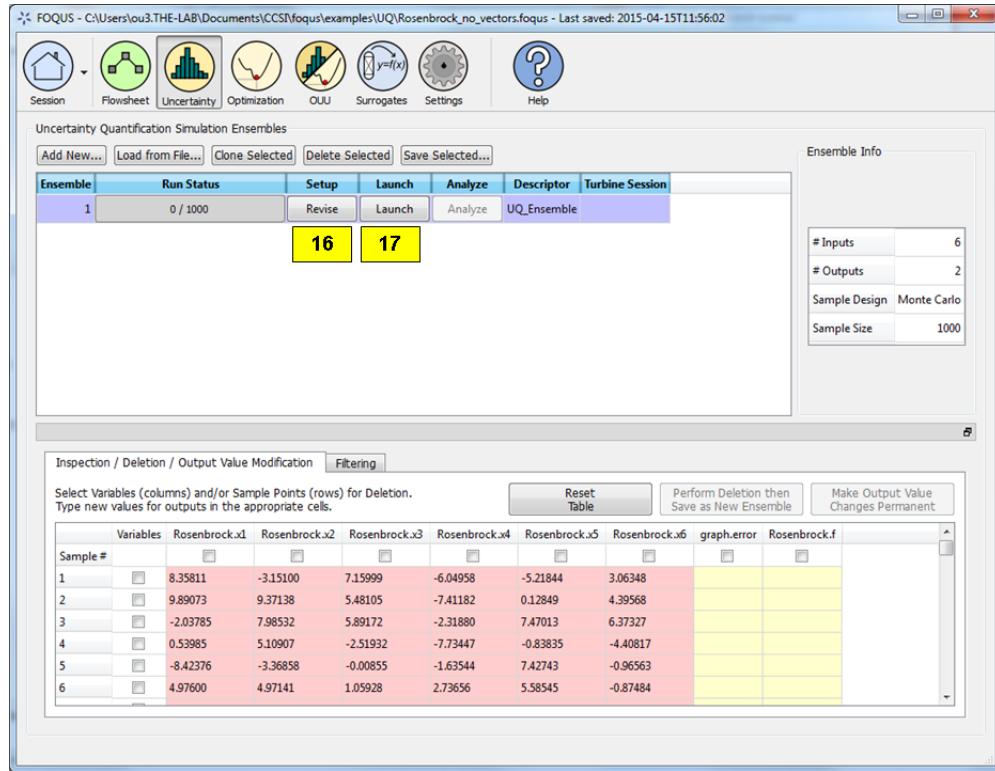


Figure 77: Simulation Ensemble Added

18. Next, look at the output.

- (a) Click **Analyze** for “Ensemble 1” (Figure 78).

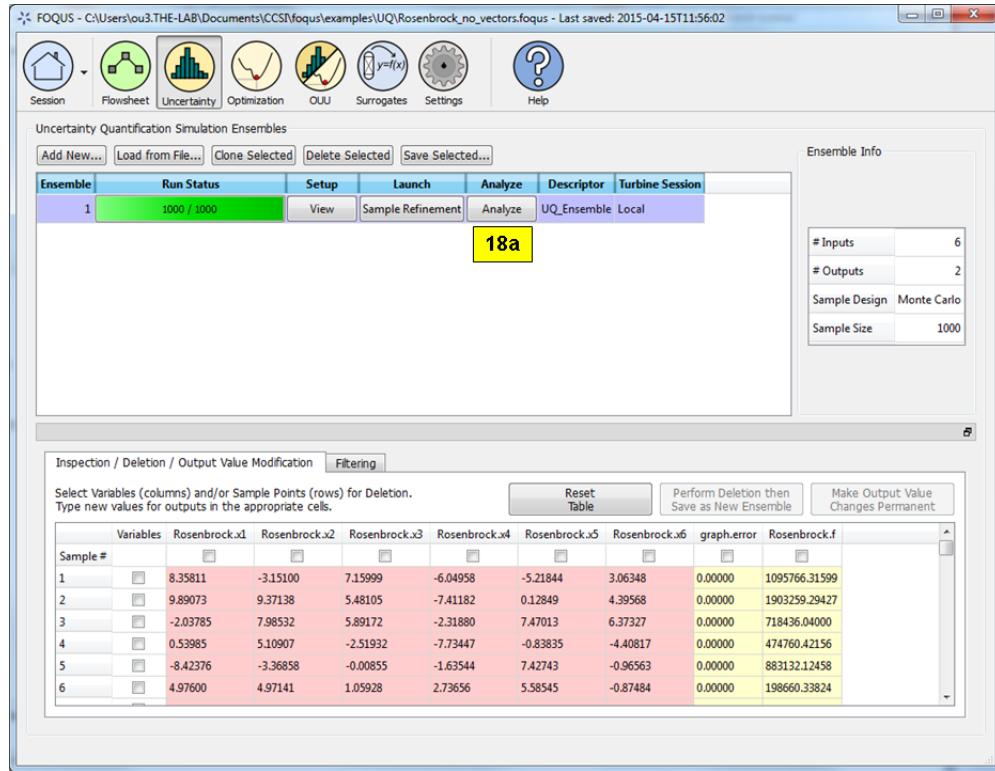


Figure 78: Simulation Ensemble Evaluation Complete

(b) Step 1 of “Analysis” (bottom page), the user selects Ensemble Data (Figure 79).

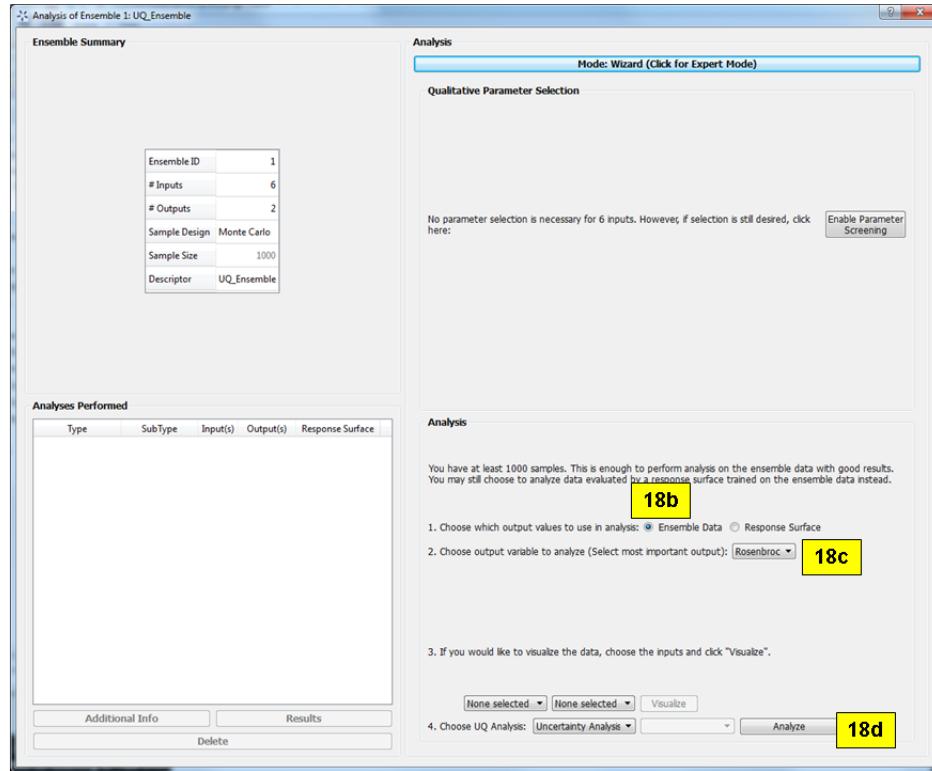


Figure 79: Simulation Ensemble Analysis

- (c) Step 2 of “Analysis” is to select “Rosenbrock.f” (Figure 79).
- (d) Step 3 of “Analysis” is to keep the analysis method as “Uncertainty Analysis” and then click **Analyze**. The user should see two graphs displaying the probability and cumulative distributions plots (Figure 80).

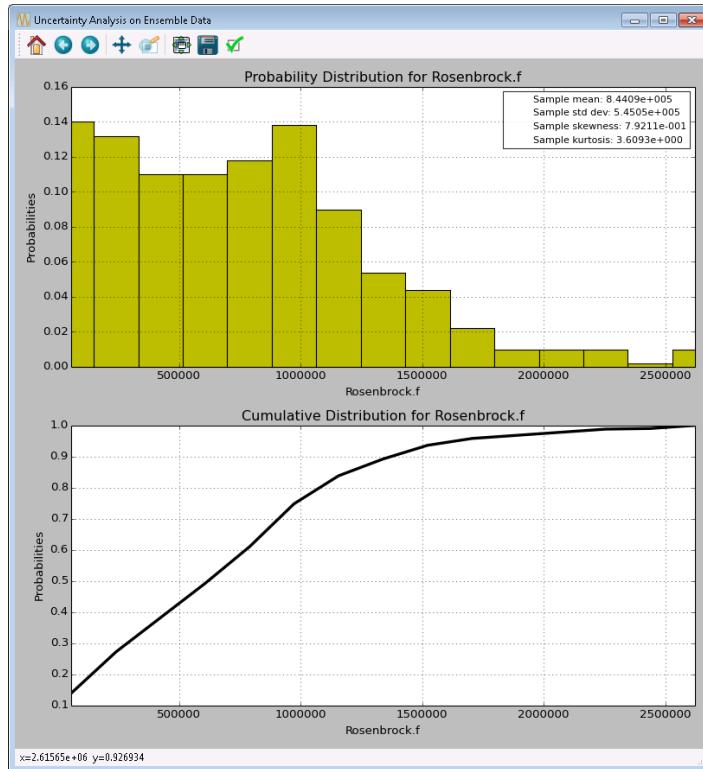


Figure 80: Uncertainty Analysis Results

Prior to this, the “Rosenbrock” example was selected to illustrate the process of creating and running a simulation ensemble because simulations complete quickly using this simple model. But from this point on, the adsorber subsystem of the A650.1 design is used as a motivating example to better illustrate how one would apply UQ within the context of CCSI.

A quick recap on our motivating example: The A650.1 design consists of two coupled reactors: (1) the two-stage bubbling fluidized bed adsorber and (2) moving bed regenerator, in which the output (outlet of sorbent stream) from one reactor is the input (inlet) for the other. The performance of the entire carbon capture system is obtained by solving these two reactors simultaneously, accounting for the interactions between the reactors. However, it is also necessary to study the individual effects of the adsorber and the regenerator without the side effects of their coupling since the two reactors display distinct characteristics under different operating conditions. Thus, the Process Design/Synthesis Team has given us a version of the A650.1 model that can be run in two modes: (1) coupled and (2) decoupled. In this section, analysis results are presented from running the A650.1 model using the decoupled mode and examining the adsorber in isolation from the regenerator.

Data Manipulation

In this tutorial, instructions to change the data before analysis are described. Current capabilities include sample filtering, input/output variable deletion, and output value modification.

Filtering

Filtering involves selecting out samples whose values of a certain input or output fall into a certain range. Typically, when runs are returned from the Turbine gateway, there could be simulations that failed to converge in Aspen, thus the simulation samples corresponding to these failed runs should be excluded from analysis. Follow the steps below to filter out the samples due to failed runs:

1. Click **Load from File** on the UQ window (Figure 81).

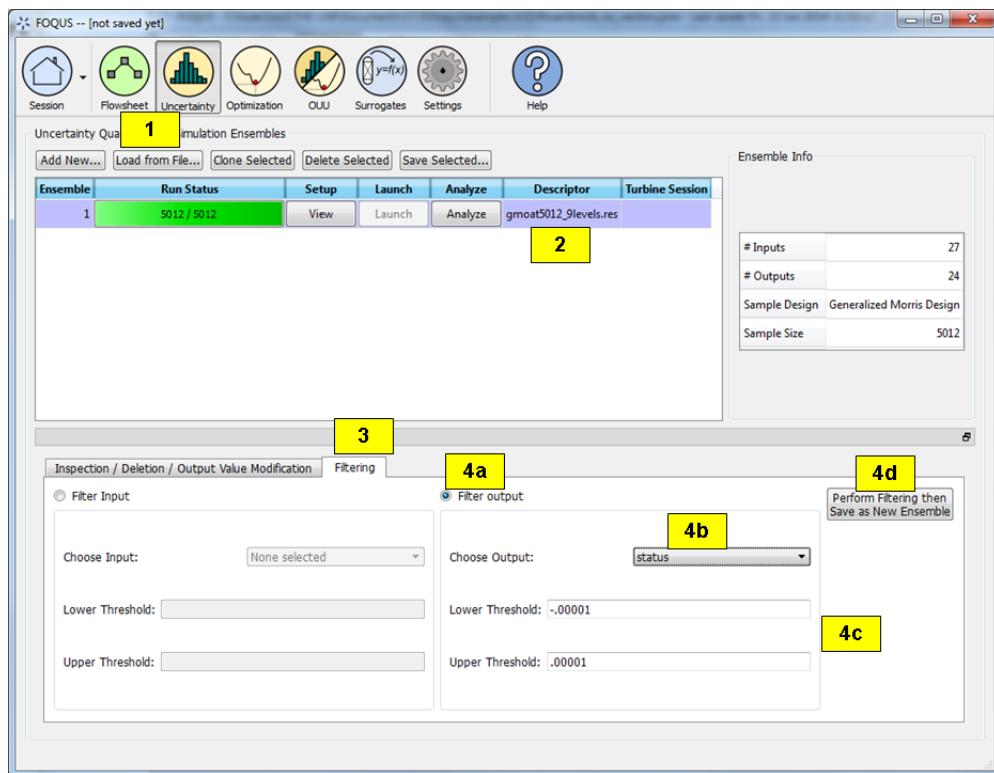


Figure 81: Data Manipulation, Filtering Tab

2. Select the file “gmoat5012_9levels.res” in the examples\UQ folder. This file is an actual simulation ensemble that has already been run on the Turbine gateway. To find this file, the user may need to change the file filter to “All files.”
3. Select the **Filtering** tab.
4. Next, filtering the loaded simulation ensemble based on output values is performed. In particular, the user should keep only the samples in which the output parameter status is “0.”
 - (a) Select **Filter Output**.
 - (b) Select “status” from the **Choose Output** drop-down list.
 - (c) Enter **-.00001** and **.00001** as **Lower Threshold** and **Upper Threshold** respectively, and then click return within “Lower threshold” and “Upper threshold.”
 - (d) Click **Perform Filtering then Save as New Ensemble**.

5. Once filtering is complete, a new row should be added to the simulation table (Figure 82). This ensemble contains only those samples that have a status value of “0.” Analysis can now be performed on this new ensemble because this ensemble contains only the valid simulations (i.e., those with output status value of 0), in which Aspen calculations have properly converged.

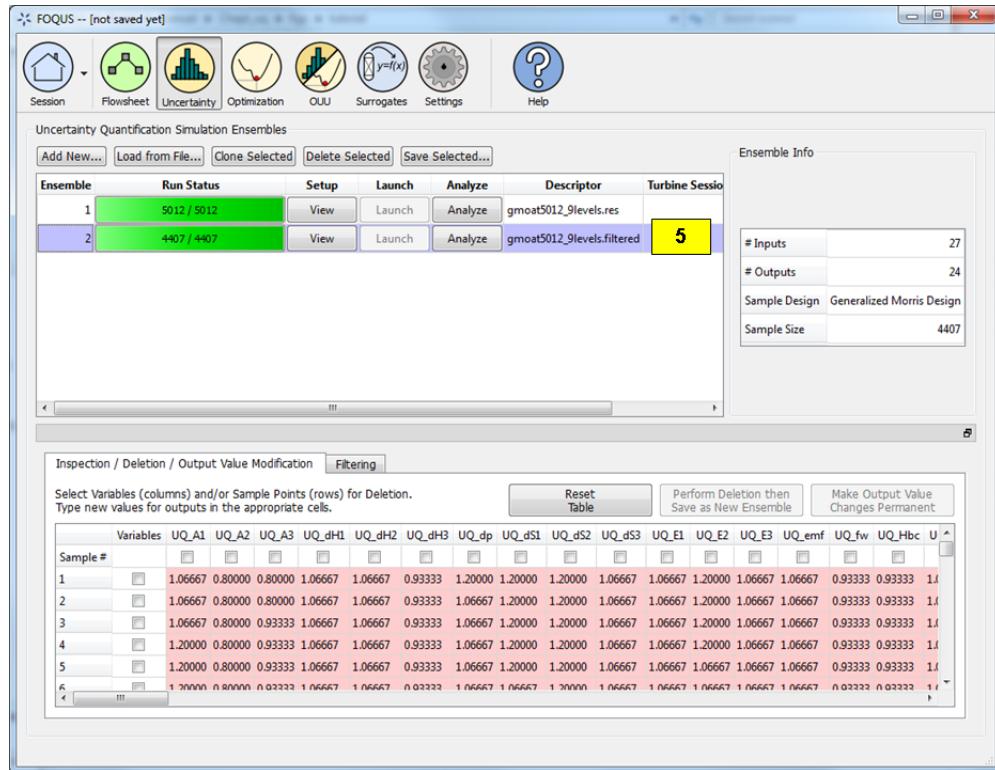


Figure 82: Data Manipulation, Filtering Results

Variable Deletion

If an input or output variable is to be removed from consideration for analysis, this can be done in the **Inspection/Deletion/Output Value Modification** tab. Delete the status output from the previous filtering as it is no longer needed for further analysis.

1. Verify that the ensemble that resulted from filtering is selected. If not, select that ensemble.
2. Click the **Inspection/Deletion/Output Modification** tab.
3. Scroll to the right of the table to the outputs, which are colored yellow.
4. Select the checkbox corresponding to the “status” output (the first output).
5. Click **Perform Deletion then Save as New Ensemble**.

The results are illustrated in Figure 83. Note: The output count has decreased by one for the new ensemble. The user can verify that the “status” output was removed in the new ensemble

by viewing this in the **Inspection/Deletion/Output Value Modification** tab again. Deletion of an input can be performed similarly by selecting its checkbox and clicking the **Perform Deletion then Save as New Ensemble** button.

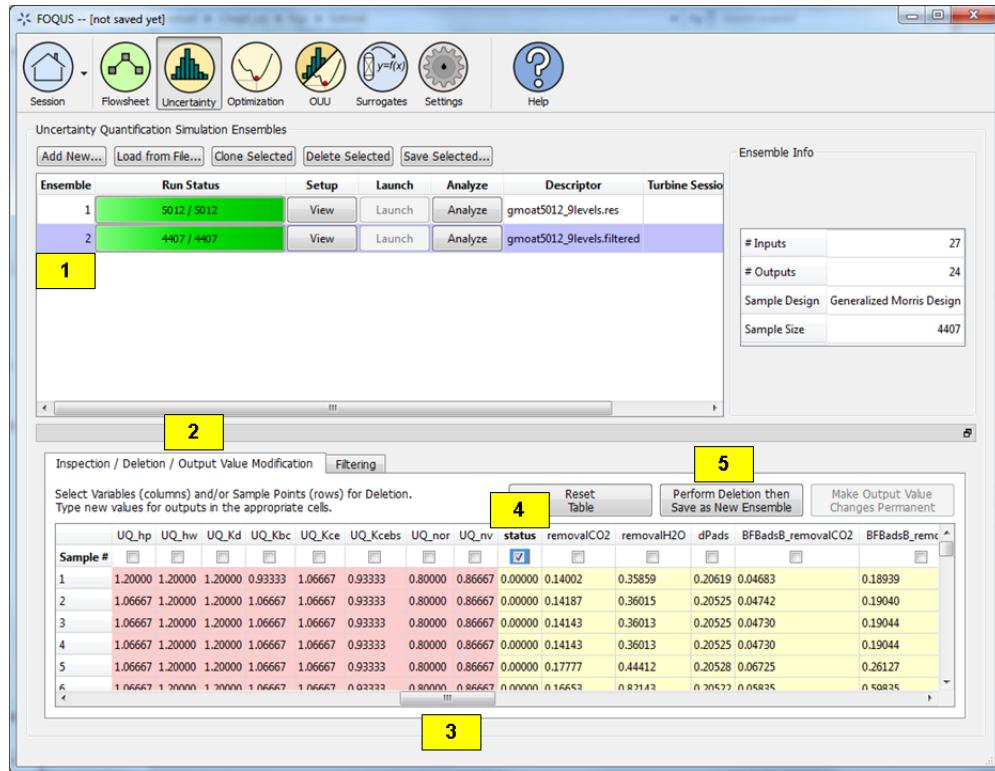


Figure 83: Data Manipulation, Inspection/Deletion

Output Value Modification

To change the value of an output for a sample or several samples, follow steps below:

1. Select an ensemble.
2. Click the **Inspection/Deletion/Output Value Modification** tab.
3. Scroll to the right to the outputs.
4. Click on a cell for one of the outputs and enter a new value. Do the same for another cell. Notice that the modified cells turn green. This indicates the cells that have been modified.
5. Click **Make Output Value Changes Permanent** to permanently change the values. The modified cells will turn yellow, indicating the permanent change. If the user wishes to reset the table and start over before making changes permanent, click the **Reset Table**.

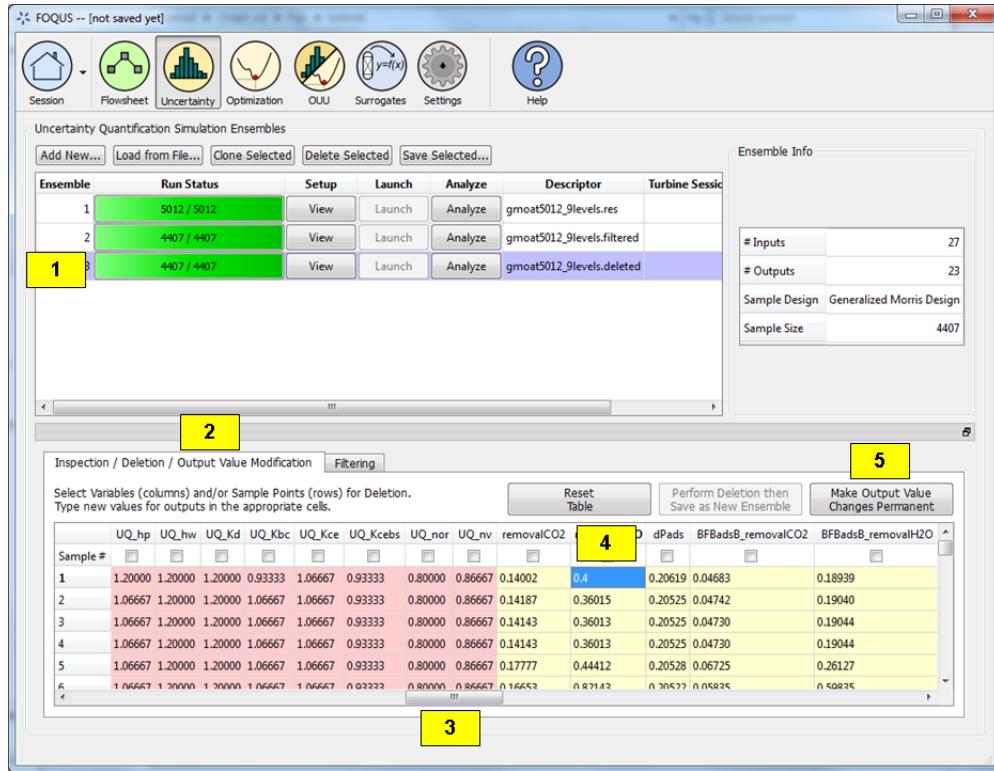


Figure 84: Data Manipulation, Value Modification

Single-Output Analysis

From the Single-Output Analysis Screen, the user can perform analyses that are specific to a particular output of interest. Here, the “removalCO2” output parameter is discussed.

Parameter Selection

For simulation models that have a large number of input parameters, it is common practice to down-select to a smaller subset of the most important input parameters that are most relevant to the output of interest. This is done so subsequent detailed studies can be performed more efficiently. By using a smaller set of inputs, a smaller set of samples may be needed.

1. From the UQ window, load the file “gmoat5012_9levels.filtered” in examples\UQ. (This file contains the same set of samples that resulted from data filtering. They are included here to make each demo self-inclusive.)
2. Click Analysis. A new page is displayed (Figure 85).

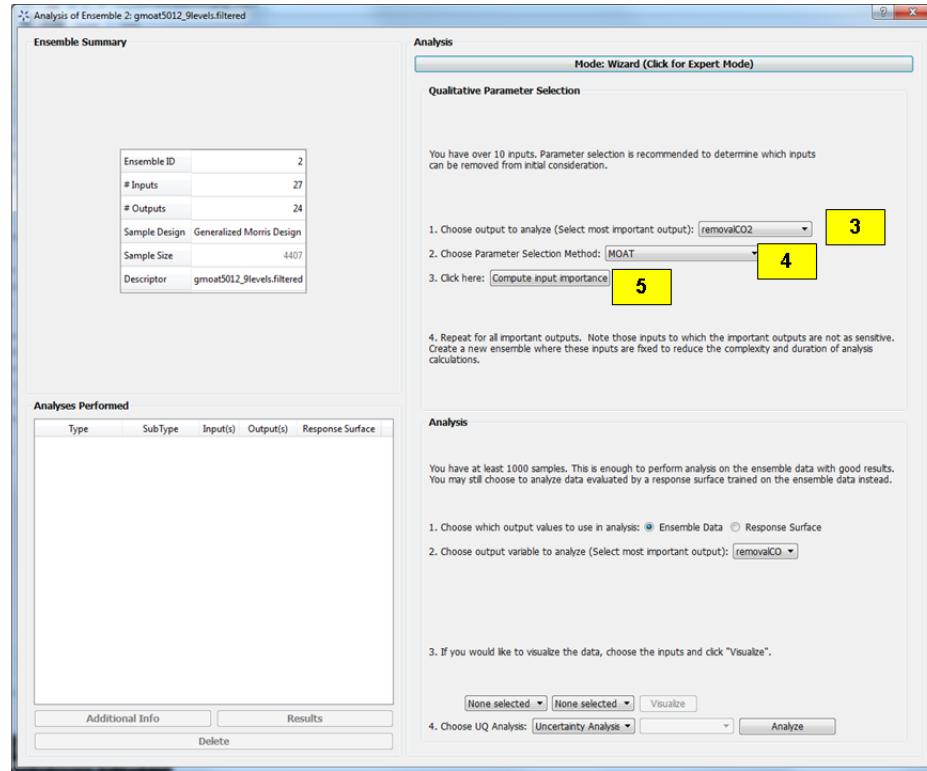


Figure 85: Analysis Dialog, Parameter Selection

3. Under the Qualitative Parameter Selection section, select “removalCO2” as the output.
4. Select “MOAT” as the method to be used.
5. Click **Compute input importance**. A graph should appear with the results (Figure 86).

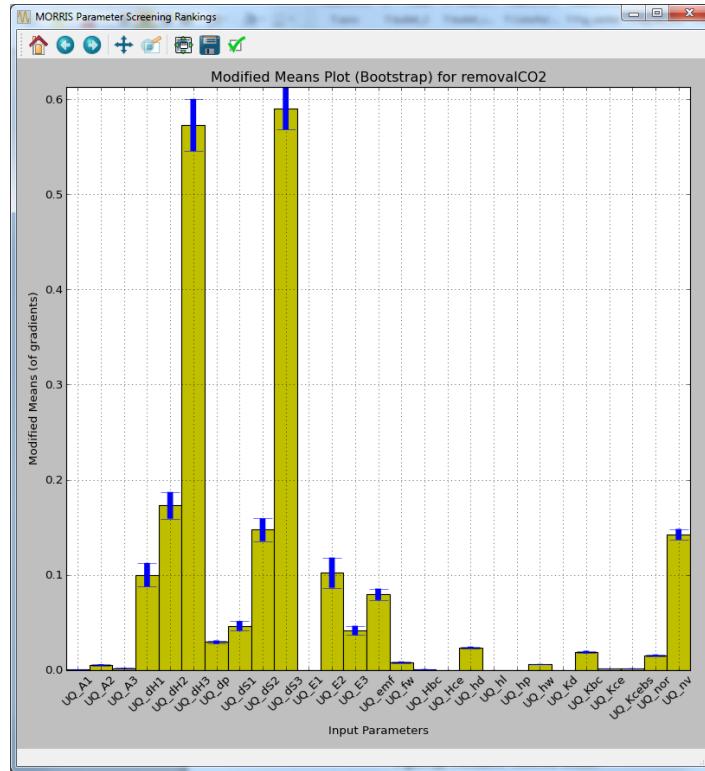


Figure 86: Parameter Selection Results

The bars in the plot represent the importance of a particular input in determining the value of the output. For example, the values of dH3 and dS3 are very important in determining the value of removalCO₂, whereas Hce and hp have no affect (the y-axis displays the average changes in the model output as a result of changing the inputs in their respective ranges. For example, from Figure 86, changing dH2 in its range results in an average change in CO₂ removal as much as about 57 percent with a margin of +/- 3 percent). Thus, it would be safe to exclude any inputs that have negligible bar lengths from analysis. Next, down-select the ten most important inputs based on these results. See Section 4.2.2 for details. Change the number of samples and scheme as desired and then generate new samples. Click **Launch** to run these samples to obtain another simulation ensemble that can be analyzed.

Ensemble Data Analysis

If the user is interested in the output uncertainty of removalCO₂ based on the uncertainties from the ten most important input parameters, perform uncertainty analysis, which would compute the probability distribution and sample statistics of “removalCO₂.”

1. Load “lptau20k_10inputs_4outputs.filtered” from the examples\UQ folder. Assume this is the file that the user would receive after running the cloned simulation ensemble in which the user has down-selected the ten most important inputs, set the **Sampling Scheme** to “Quasi-

Monte Carlo (LPTAU)”, set the sample size to 20K, and performed data filtering to retain only the samples with the status output equal to “0.”

2. Click Analyze. A new page displays (Figure 87).
3. Select “Ensemble Data” to indicate that analysis is to be directly performed on the raw sample data.
4. Select “removalCO2” as the output variable to analyze.
5. Select “Uncertainty Analysis” and then click Analyze.

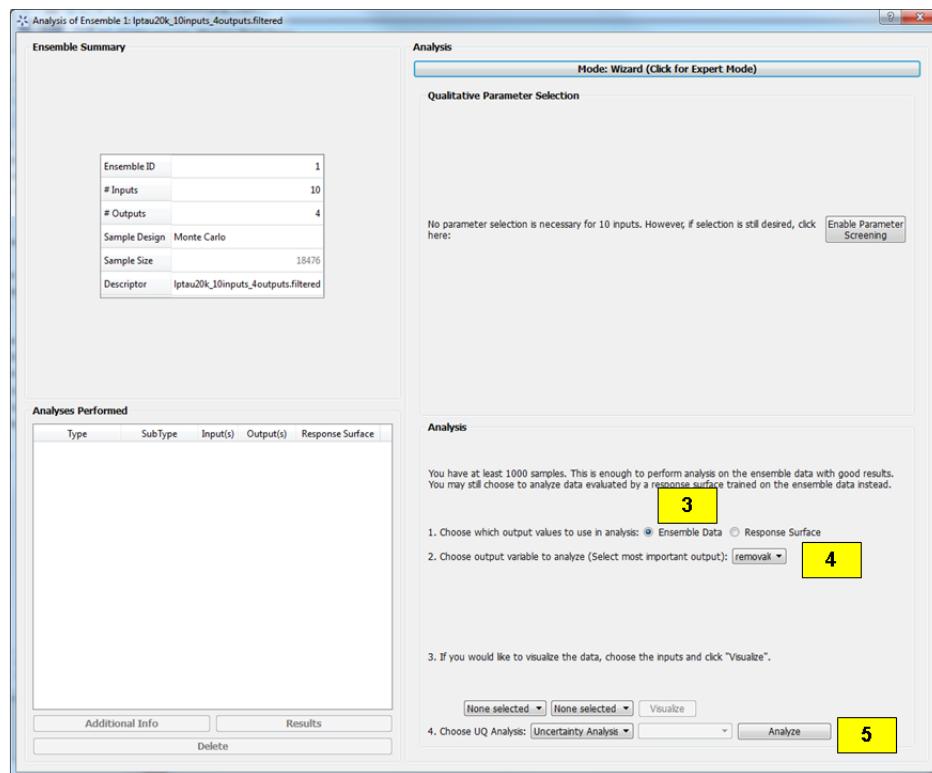


Figure 87: Analysis Dialog, Ensemble Data Uncertainty Analysis

Once uncertainty analysis is complete, results display (Figure 88) illustrating the probability distribution function (PDF), cumulative distribution function (CDF), and the sufficient statistics (e.g., mean, standard deviation) of “removalCO2” (top left corner of the PDF plot). This is used to evaluate if the output uncertainty is acceptable. If the output uncertainty is too great or the user prefers the system to operate within a higher percentage of capture, pursue further analyses to understand the relationships between the inputs and outputs, and investigate what can be done to reduce the output uncertainties by reducing the input uncertainties.

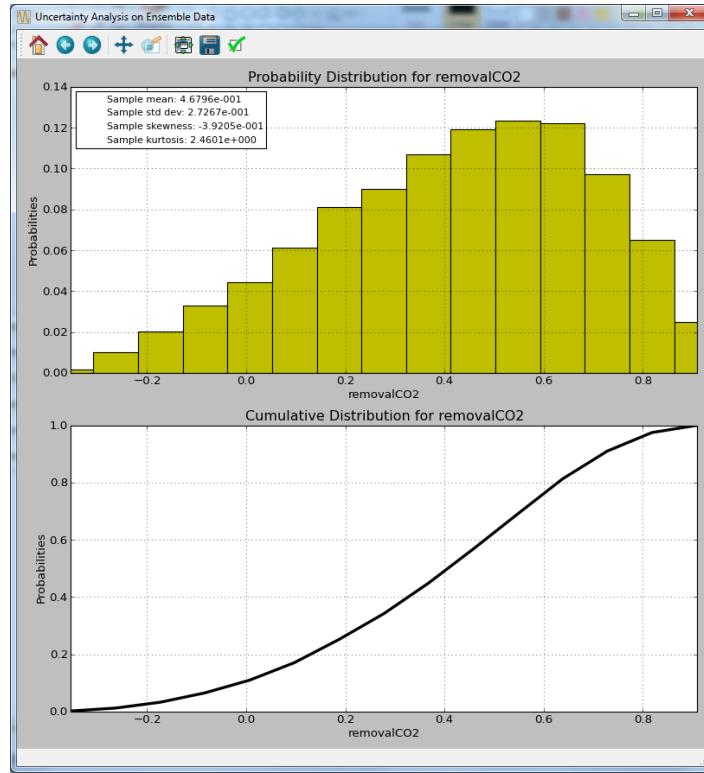


Figure 88: Ensemble Data Uncertainty Analysis Results

Next, the user may apply variance-based sensitivity analysis to quantify each inputs contribution to the output variance:

6. From the bottom of the “Analysis” section, select “Sensitivity Analysis.”
7. There are three options for sensitivity analysis: (1) first-order, (2) second-order, and (3) total-order. First-order analysis examines the effect of varying an input parameter alone. Second-order analysis examines the effect of varying pairs of input parameters. Total-order analysis examines all interactions’ effect of varying an input parameter alone and as a combination with any other input parameters. For this demonstration, select “Total-order” and click **Analyze**. The total sensitivity indices display in a graph. Note: If the simulation ensemble has more than ten inputs, “Total-order” is disabled (since any reasonable sample size is not sufficient). Additionally, since quantitative sensitivity analysis in general requires large ensembles with many samples (thousands or more), ensemble sensitivity analysis (without the use of response surfaces) is often less practical and accurate than response surface based analyses. The result is illustrated in Figure 89.



Figure 89: Ensemble Data Total-order Sensitivity Analysis Results

These results confirm that “removalCO2” is more sensitive to “dH3” and “dS3” than other inputs. (The y-axis displays an approximate percentage of output variance attributed to each individual parameter. Since total sensitivity includes higher order interaction terms with other parameters, the sum of these total sensitivity indices usually exceeds 1.)

Ensemble Data Visualization

1. In this release, ensemble data visualization is only available in “Expert” mode. At the top of the “Analyze” page, toggle the bar to expert mode and select “removalCO2” as the output. Next, to “Visualize Data,” choose an input (e.g., “UQ_dH1”) and click **Visualize** for a 2-D scatter plot of “removalCO2” versus that input (Figure 90).

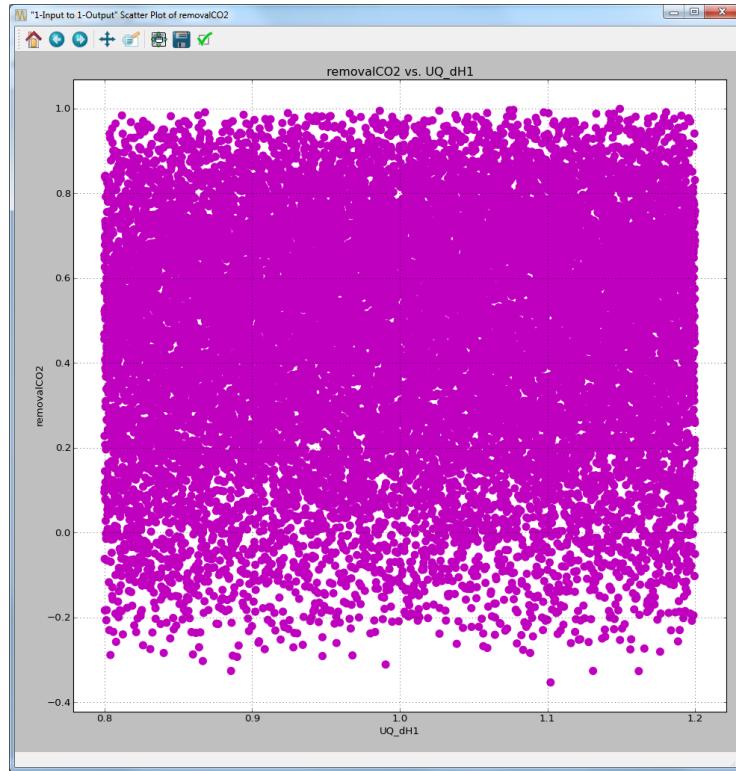


Figure 90: Ensemble Data Visualization of One Input

2. Next, select a second input (e.g., "UQ_dH2") and click **Visualize** for a 3-D scatter plot of "removalCO2" versus the two inputs. (Note: The input selections must be unique for the **Visualize** button to be enabled.) Figure 91 shows the results.

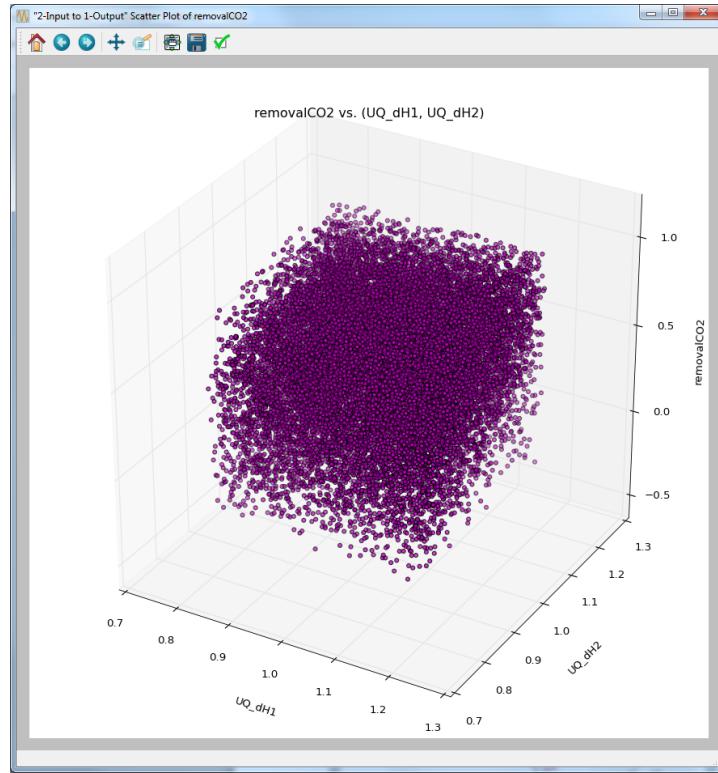


Figure 91: Ensemble Data Visualization of Two Inputs

The plot in Figure 91 can be rotated by clicking and dragging.

Response Surface Based Analysis

For simulation models that are expensive to run, response surface analysis can be a resourceful option. To construct a response surface, a space-filling sampling design is desired. For example, quasi-Monte Carlo (LPTAU) or Latin hypercube sampling schemes are recommended. Additionally, there are several possibilities for curve fitting methods. If the sample size is relatively small, polynomial regression or Gaussian process (if installed as part of PSUADE) is preferred. Alternatively, if the sample size is large enough (one hundred or more), cubic splines (if installed) may also be feasible.

Response Surface Model Validation

To proceed with response surface based analysis, the user needs to find a suitable response surface with which to approximate the input-to-output mapping. Validation is performed to see how well a particular response surface can predict a subset of the withheld data.

1. Load “lptau100_10inputs_4outputs.dat” from the examples\UQ folder. Note: This is an extremely small simulation ensemble, as this is used to highlight the differences (in validation results) between a good response surface and a bad one.

2. Click **Analyze** for the current ensemble. A new dialog page displays (Figure 92).
3. Under “Analysis” (bottom section), under Step 1, select “Response Surface.”

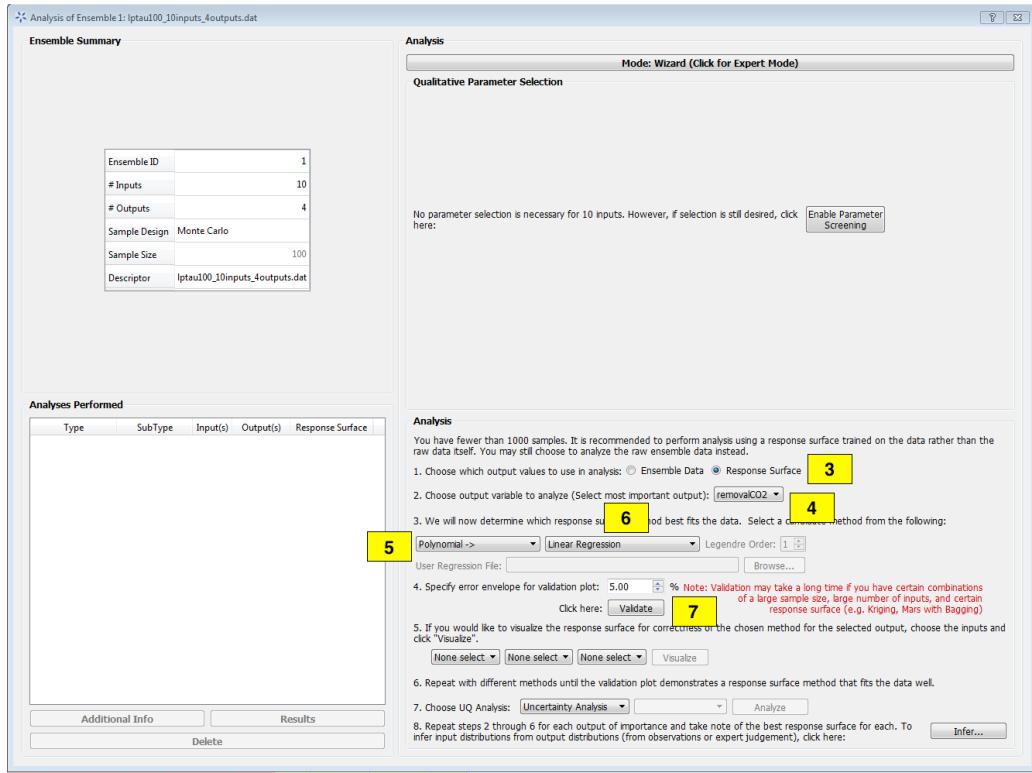


Figure 92: Analysis Dialog, Response Surface Validation of Linear Model

4. Under Step 2, select “removalCO2” as the output for analysis.
5. Under Step 3, select “Polynomial” for response surface method.
6. There are multiple types of polynomial response surfaces, with increasing complexity as the user navigates down the list. For now, select “Linear” in the next drop-down list.
7. Insert 5.00 as the error envelope for the validation plot. Click **Validate**. The result is illustrated in Figure 93.

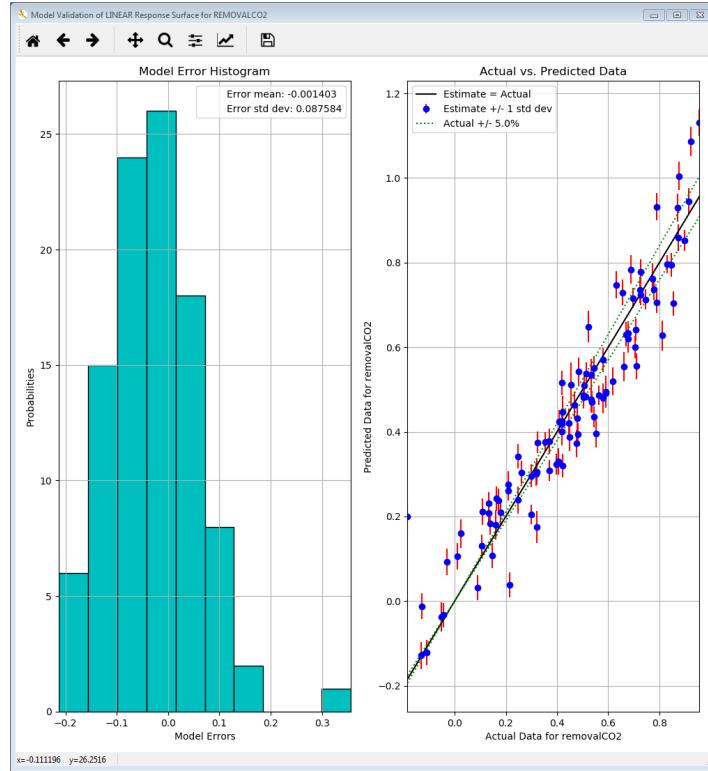


Figure 93: Linear Response Validation Results

The cross-validation results for the linear regression model are displayed as a histogram of errors to the left and a plot of predicted values versus actual values to the right. The histogram displays the cross validation error distribution, which provides the user information on what the errors are like overall. If this distribution is not centered on zero, there may be a systematic bias in the response surface model. If the distribution is too wide, it is not a good fit. As for the plot of predicted values versus actual values, the more closely the points are to the diagonal, the better the fit. Most response surface models, with the exception of MARS, also provide uncertainty information about the response surface. The vertical error bars on the left plot reflect the uncertainty in the linear response's predictions.

In summary, these two figures should provide sufficient information for the user to judge how good the fit is. As is apparent in the figures, the linear model consistently overestimates and thus is an ill-suited response surface to model our data. In general, the user may use a few response surface methods to see which method gives the best fit.

Response Surface Based Uncertainty Analysis

These capabilities are similar to those for ensemble data analysis. The difference is that the results are now derived from a much larger ensemble that is computed from the response surface. With the 100 samples from the ensemble data, a response surface is trained and is used to generate

100K samples internally to compute the results for uncertainty and sensitivity analyses. (Note: Validation must be performed before these analyses are available.)

After the response surface validation step, select “Uncertainty Analysis” to be the UQ analysis in Step 7 of “Analysis” (Figure 92). Click Analyze and a distribution representing the output uncertainty will be displayed (Figure 94).

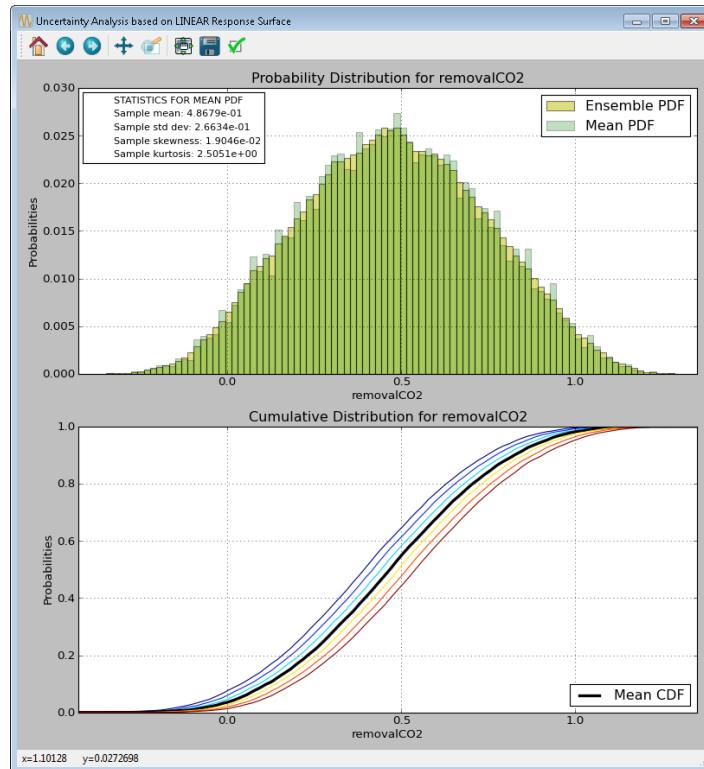


Figure 94: Response Surface Based Uncertainty Analysis Results

Compare the response surface based uncertainty results (Figure 94) to the results from ensemble data analysis (Figure 88). The two main differences are easily seen.

- **Two PDFs on top plot:** A response surface (in this case, linear regression) is used to predict the output values corresponding to the input samples. From the validation step (left plot of Figure 93). Note: There is error associated with the response surfaces predictions. This error is propagated in uncertainty analysis, in the form of standard deviations around the predicted output values (i.e., the means).

Accordingly, two histograms are presented: The “mean PDF” represents the output probability distribution computed from the response surface’s predicted output values only, without consideration for the uncertainties surrounding these predicted values. The “ensemble PDF” represents the output probability distribution that encompasses the uncertainties surrounding these predicted values. In most cases, the ensemble PDF should have a larger spread because

it is accounting for more uncertainties (i.e., those that stem from the approximations inherent in the response surface).

- **Multiple cumulative distribution functions (CDFs) on bottom plot:** The “mean CDF” is constructed from a cumulative sum on the mean PDF in the top plot. Since each predicted output value (i.e., the mean) has an associated standard deviation, this information is used to construct other PDFs that correspond to output values that are +/- 1, 2, and 3 standard deviations from the mean. These PDFs are then converted to CDFs and shown as colored lines. These colored lines provide an uncertainty “envelope” around the mean CDF.

Response Surface Based Mixed Epistemic-Aleatory Uncertainty Analysis

In “Expert Mode”, the user can perform more advanced uncertainty analysis that handles both epistemic and aleatory uncertainties. To do so, the user will need to designate the uncertainty type (epistemic or aleatory) for each uncertain input. In general, epistemic uncertainties are reducible uncertainties that arise due to lack of knowledge, such as simplifying assumptions in a mathematical model. Therefore, epistemic uncertainty is often characterized by upper and lower bounds. On the other hand, aleatory uncertainties are irreducible uncertainties that represent natural, physical variability in the phenomenon under study. As such, aleatory uncertainties are often characterized by distributions. Hence, the user is required to provide a PDF for each aleatory input. (In FOCUS, with the exception of mixed epistemic-aleatory uncertainty analysis, all uncertain inputs are treated as aleatory inputs.)

To perform mixed epistemic-aleatory uncertainty (Figure 95), switch to “Expert Mode” by clicking the **Mode** button that toggles between the analysis modes. After response surface validation, select “Uncertainty Analysis” in the first **Choose UQ Analysis** drop-down list, then “Epistemic-Aleatory” in the secondary drop-down list, for the UQ analysis. In the input table, designate the parameter **Type** (“Epistemic”, “Aleatory” or “Fixed”) and the corresponding information for each input. Once complete, click **Analyze**.

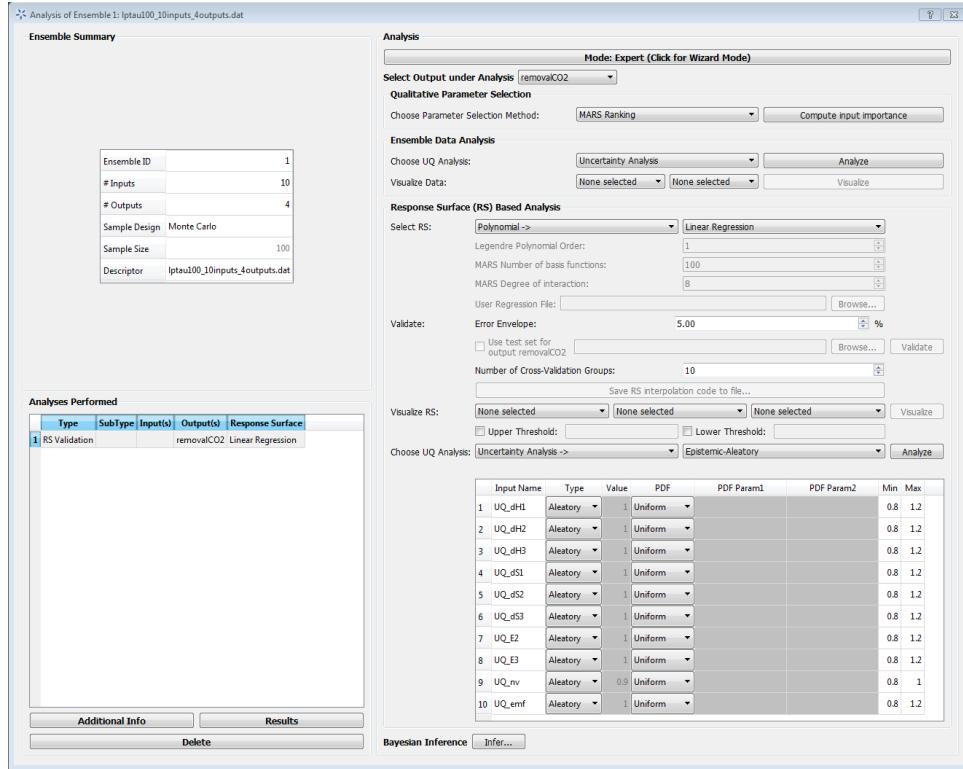


Figure 95: Response Surface Based Mixed Epistemic-Aleatory Uncertainty Analysis

The results of mixed epistemic-aleatory uncertainty analysis is a plot (Figure 96) containing multiple CDFs. In the mixed analysis, the epistemic inputs are sampled according to their lower and upper bounds. Each sample point spawns a response surface based uncertainty analysis, in which the epistemic inputs are fixed at their sampled value and the aleatory input uncertainties are propagated to generate a CDF that represents the output uncertainty. A slider is provided for the user to extract the probability range corresponding to a particular value of the output.

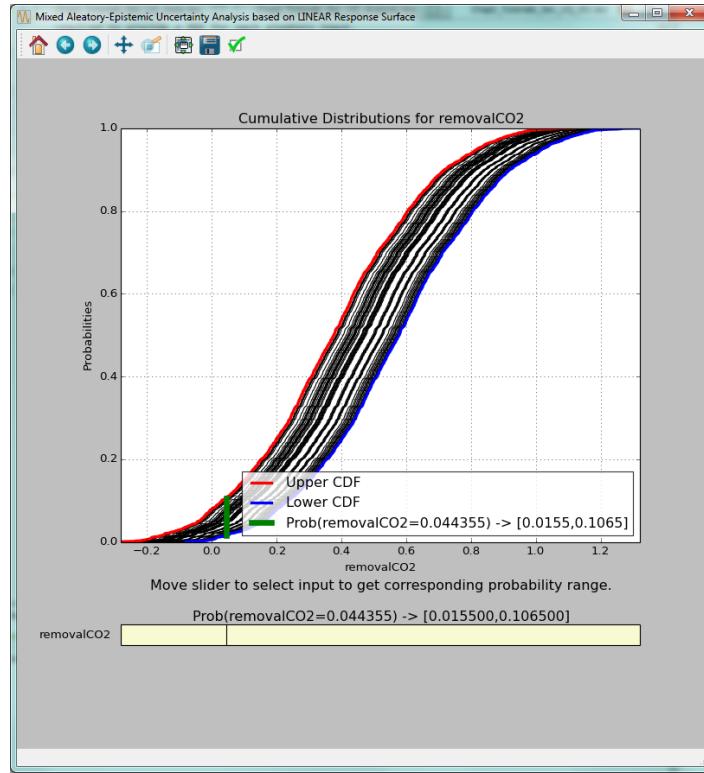


Figure 96: Response Surface Based Mixed Epistemic-Aleatory Uncertainty Analysis Results

Response Surface Based Sensitivity Analysis

For quantitative sensitivity analysis, follows these steps:

1. In the **Choose UQ Analysis** drop-down list (Step 6 of “Analysis”), select “Sensitivity Analysis.”
2. In the next drop-down list, select “First-order” and click **Analyze**. (This analysis may take a long time depending on the sample size and the response surface used.)

Prediction errors are associated with the response surface’s predictions of the output values (left plot of Figure 93). Earlier, it was observed that the response surface error contributed to the output uncertainty, leading to a larger spread in the output PDF (top plot of Figure 94). In Figure 97, the response surface error contributed to uncertainty (shown as blue error bars) surrounding each input’s contribution to the output variance (shown as yellow bars).

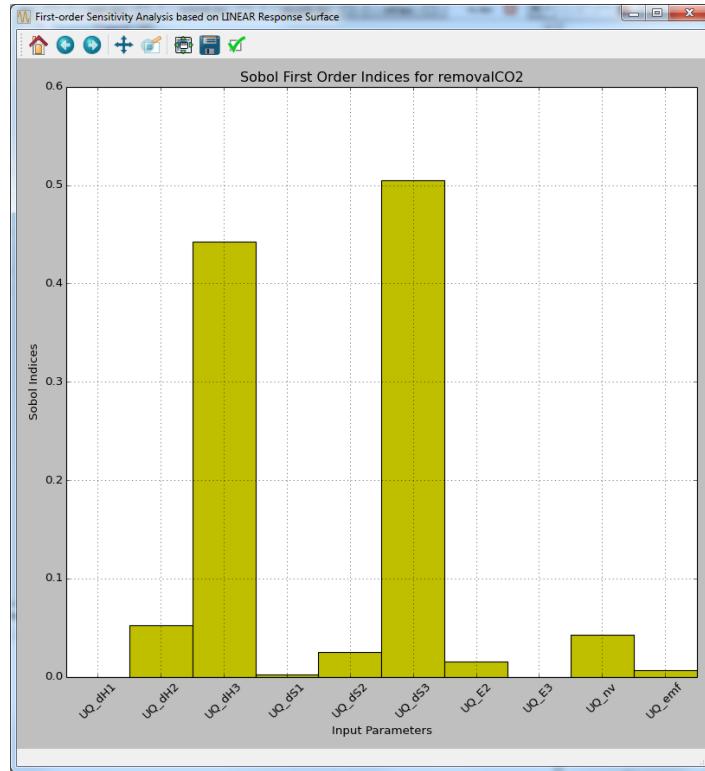


Figure 97: Response Surface Based First-order Sensitivity Results

Response Surface Based Visualization

The response surface that has been validated can also be visualized.

1. Select one input next to “Visualize Response Surface.”
2. Click Visualize to display a 2-D line plot that displays “removalCO2” versus the selected input.

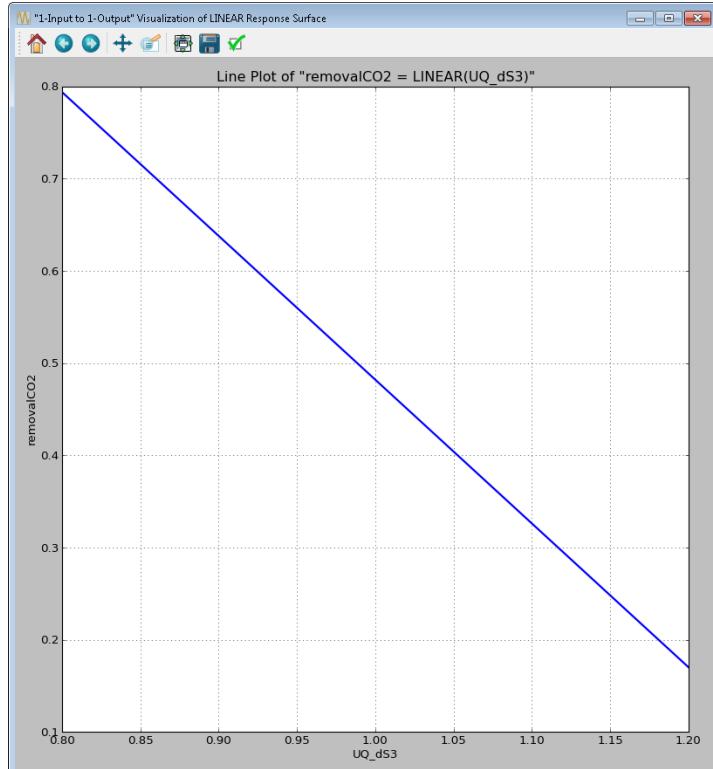


Figure 98: 1-D Response Surface Visualization

3. Select another input next to the first one for a 2-D response surface visualization.
4. Click Visualize to display a figure with a 3-D surface plot and a 2-D contour plot (Figure 99).

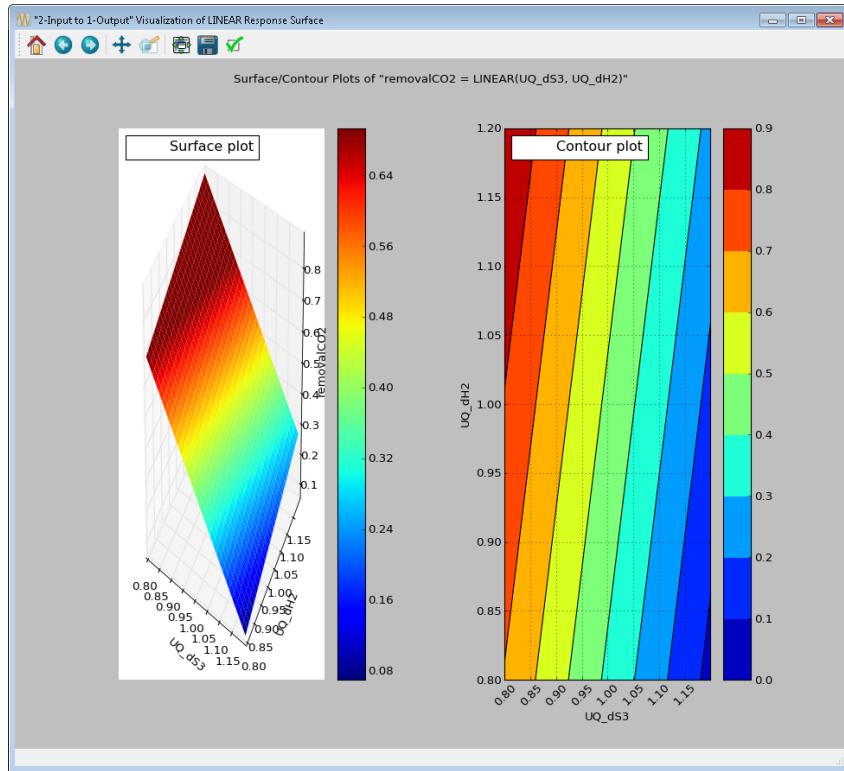


Figure 99: 2-D Response Surface Visualization

5. Select another input next to the second one for a 3-D response surface visualization.
6. Click **Visualize** to display a 3-D isosurface plot. Move the slider to see the points in the 3-D input space that fall within the small range of “removalCO2” (Figure 100).

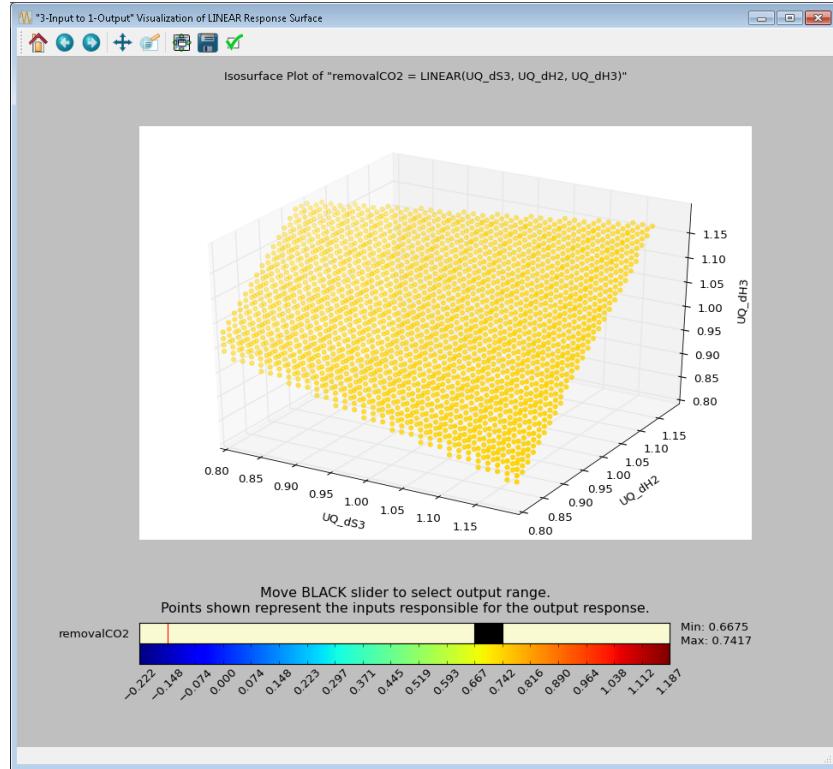


Figure 100: 3-D Response Surface Visualization

Bayesian Inference

For each output variable, the user specifies an observed value (from physical experiments) with the associated uncertainties (in the form of standard deviation), if applicable. Whether standard inference or SolventFit is selected, the tool will launch a Markov Chain Monte Carlo (MCMC) algorithm to compute the posterior distributions of the uncertain input parameters. These input posterior distributions represent a refined hypothesis about the input uncertainties in light of what was previously known (in the form of input prior distributions) and what was observed currently (in the form of noisy outputs).

1. Load the file “lptau5k_10inputs_4outputs.filtered” from the examples\UQ folder.
2. Click Analyze for the current ensemble and a new dialog box displays (Figure 101).

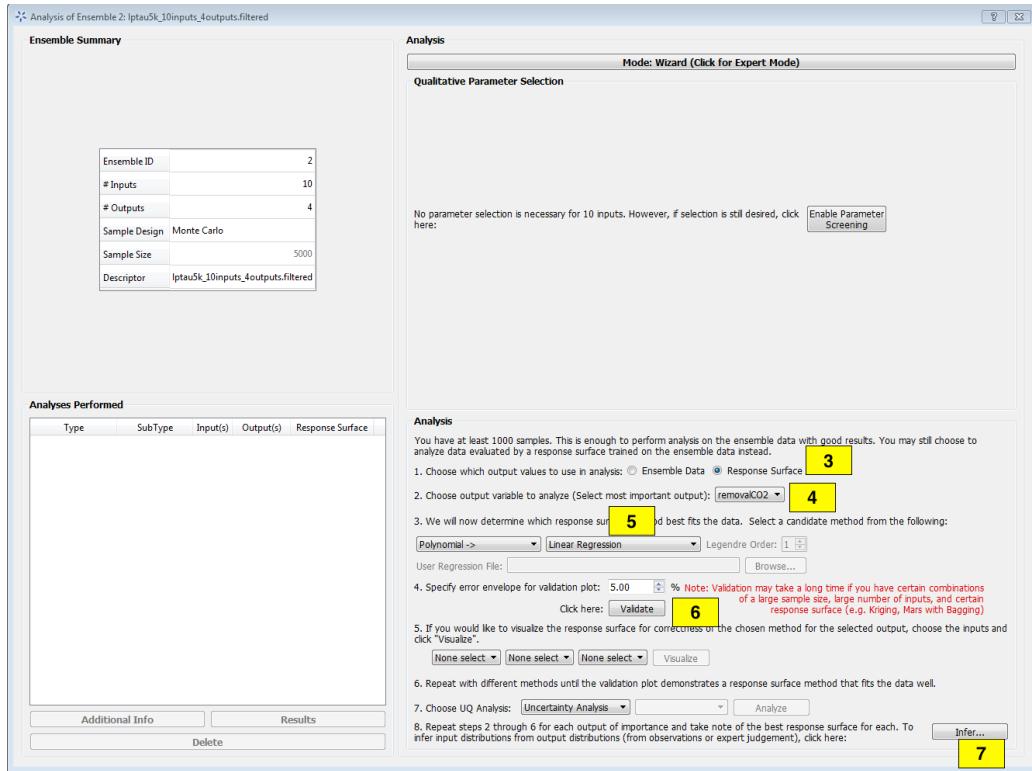


Figure 101: Analysis Dialog, Bayesian Inference

3. Select “Response Surface” in the “Analysis” section.
4. Select “Output variable to analyze” to be “removalCO2.”
5. Select “Linear Regression” as the response surface.
6. Insert 5.00 as the error envelope for the validation plot. Click Validate. The GUI allows the user to proceed with Bayesian inference after one input has been validated; however, the user may want to validate all outputs since they are all used in the inference.
7. Once validation is completed, click Infer at the lower right corner, which displays a new dialog box (Figure 102).
8. In the **Output Settings** table (on the left), select the second, third, and fourth outputs as the observed outputs. The user can experiment with using different response surface models (for example, linear polynomials) to approximate the mapping from inputs to each of the outputs.
9. In the **Input Settings** table (on the right), designate input types (variable, design, or fixed) and if necessary, switch to Expert Mode to revise the prior distribution on the input parameters. The prior distribution represents knowledge that the user possesses about the inputs before observational data (from experiments) has been incorporated into this knowledge. If the user does not have any updated knowledge about the simulation ensemble, it is OK to leave the table as is.
10. In the **Observations** table (in the middle), select the number of experiments from which the user can get observational data. In essence, if the user has N observations, then N should be set as the number of experiments. The table will then populate columns for design

inputs (if any) and observed outputs. Currently, only normal distribution is supported as the noise model for observations. Enter the mean and standard deviation for each of these observations. For convenience, the mean and standard deviation values are prepopulated with the results from uncertainty analysis. These values have been provided as a sanity check for the user, in case the observation for a particular output is way out of range from these distributions.

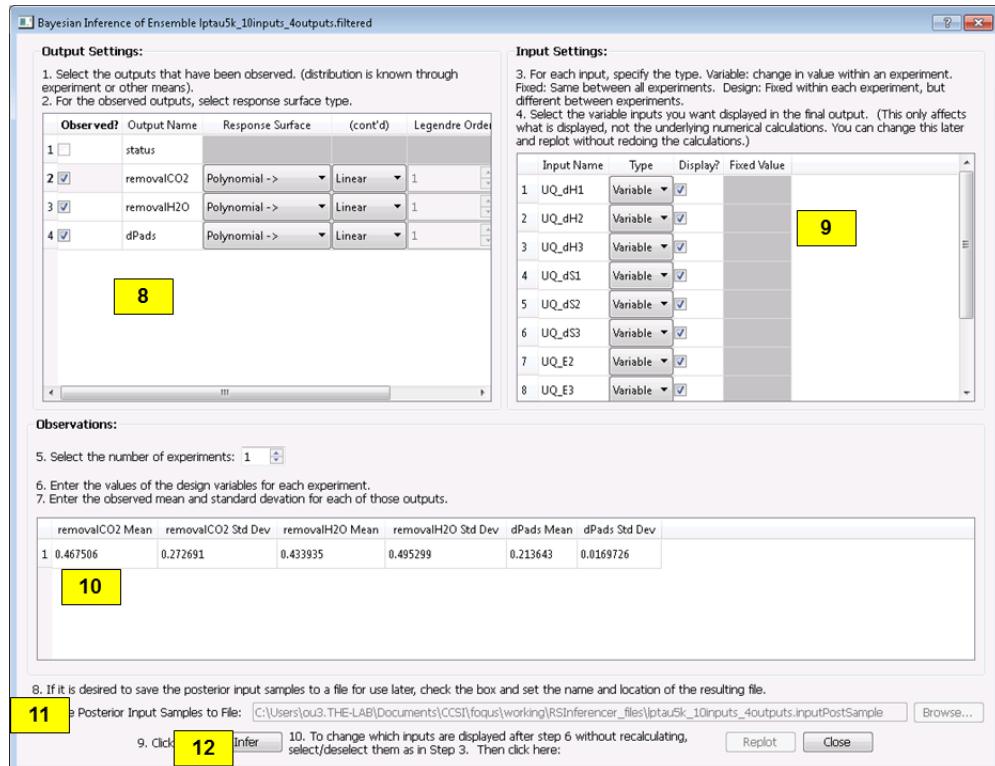
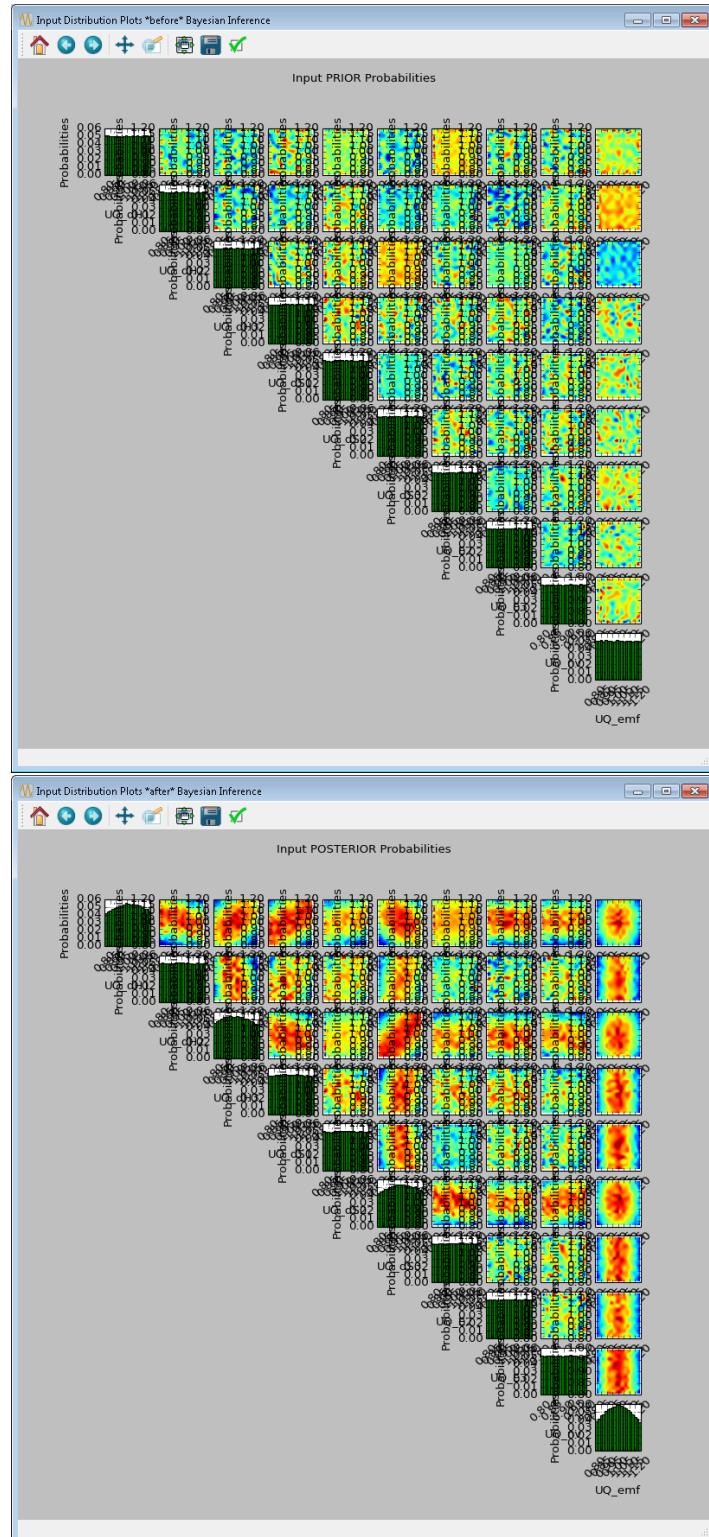


Figure 102: Bayesian Inference Dialog for Standard Inference

11. To save an input sample drawn from the posterior distribution, select the Save Posterior Input Samples to File checkbox and select a location and file name to store the sample.
12. Click **Infer** to start the analysis. Inference can take a long time; thus, a stop feature has been implemented. Once inference starts, the **Infer** button changes to **Stop**. To stop inference calculations, click **Stop** which changes the button back to **Infer**, allowing the user to restart the calculations from scratch. If inference is allowed to run its course, its results are interpolated to produce heat maps (off-diagonal subplots in Figure 103) for visualization. This interpolation step can take a few minutes and while it is running, **Infer** is disabled.

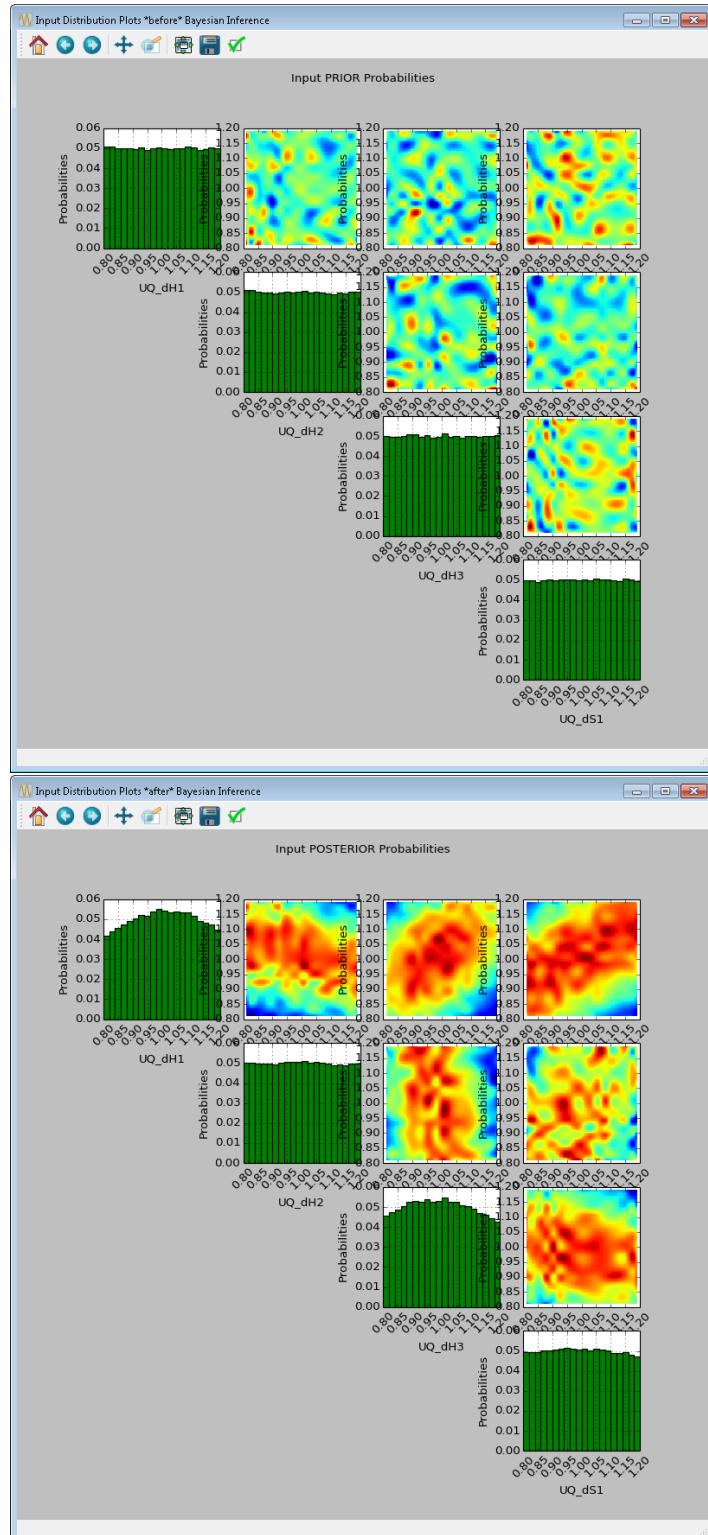
**Figure 103: Bayesian Inference Results**

Once the inference and interpolation steps are complete, two windows will be displayed: a multi-

plot figure of the prior distributions and another multi-plot figure of the posterior distributions. If the user has selected the **Save Posterior Input Samples to File** checkbox, then a sample file will also be written to the designated file location.

In the resulting prior and posterior plots (Figure 103), the univariate input distributions are displayed as histograms on the diagonal. The bivariate input distributions (between pairs of inputs) are displayed as heat maps in the off-diagonal subplots. On these heat maps, the regions in red reflect the input space with higher probability. In the posterior plots, the red regions represent inputs that are more likely to have generated the specified observations on the outputs. By comparing the prior and the posterior figures, the user can see the "before" and "after" impact of inference on our knowledge of the input uncertainty.

To zoom in on any one of the subplots, left-click; to zoom out, right-click. To display a subset of these subplots, clear the checkbox for the inputs to be omitted (from the first column of the Input Prior Table) and click **Replot** (Figure 104).

**Figure 104: Bayesian Inference Replot Results**

File Formats

Most UQ capabilities within FOCUS rely on PSUADE. As such, different UQ components require input files in PSUADE formats. CSV (comma-separated values) files are also compatible. The specific requirements are explained in Sections 4.2 and 6.

PSUADE Full File Format

The following is an example of the full PSUADE file format. Comments in red do not appear in the file and are only for instructional purposes.

```

PSUADE_IO (Note : inputs not true inputs if pdf ~=U) Start data block
5 2 6
      5 inputs, 2 outputs, and 6 samples
1 1
      Sample index, run value (0 if sample point has not been calculated.)
      Value to the first input of sample 1
      Value to the second input of sample 1
      Value to the third input of sample 1
      Value to the fourth input of sample 1
      Value to the fifth input of sample 1
      Value to the first output of sample 1 (9.99e+34 if undefined)
      Value to the second output of sample 1 (9.99e+34 if undefined)
2 1
      Sample point 2. Run value is true (outputs calculated).
      -9.5477386934673336e-02
      8.5427135678391997e-02
      -9.7989949748743721e-01
      -4.8743718592964824e-01
      3.5175879396984966e-02
      9.7708275149071300e-01
      8.6655187317087226e-02
3 1
      Sample point 3. Run value is true (outputs calculated).
      -6.9849246231155782e-01
      -5.9798994974874375e-01
      -9.6984924623115576e-01
      2.5125628140703515e-02
      8.1909547738693478e-01
      -6.4229247835711212e-02
      2.8546752874255432e-01
4 1
      Sample point 4. Run value is true (outputs calculated).
      2.1608040201005019e-01
      7.2864321608040195e-01
      4.9748743718592969e-01
      5.6783919597989962e-01
      6.7839195979899491e-01
      -4.7115433927748318e-01
      -3.5869634004753126e-01
5 1
      Sample point 5. Run value is true (outputs calculated).
      5.6783919597989962e-01
      5.4773869346733672e-01

```

```

-2.2613065326633164e-01
 3.8693467336683418e-01
-1.7587939698492461e-01
 6.8926859881410230e-03
-2.7551395275787588e-01
6 0                               Sample point 6. Run value is false (outputs not calculated).
-7.2864321608040195e-01
 2.1608040201005019e-01
 8.3919597989949746e-01
 3.5175879396984966e-02
 2.3618090452261309e-01
 9.999999999999997e+34   Output not calculated.
 9.999999999999997e+34   Output not calculated.

PSUADE_IO      End data block
PSUADE         Start informational block
INPUT          Start input information block
  dimension = 5      Number of inputs
  variable 1 A0    = -1.00000e+00  1.00000e+00  Input name & range
  variable 2 A1    = -1.00000e+00  1.00000e+00
  variable 3 A2    = -1.00000e+00  1.00000e+00
  variable 4 A3    = -1.00000e+00  1.00000e+00
  variable 5 A4    = -1.00000e+00  1.00000e+00
END            End input information block
OUTPUT         Start output information block
  dimension = 2      Number of outputs
  variable 1 Y1     Output name
  variable 2 Y2
END            End output information block
METHOD          Start sampling method information block
  sampling = LH      Latin Hypercube sampling
  num_samples = 6    Number of samples
END            End sampling method block
APPLICATION      Start application block
  driver = NONE     Name of driver program for calculating outputs (NONE for no driver)
END            End application block
ANALYSIS         Start analysis method information block
  analyzer output_id = 1
  analyzer rstype = MARS  Default response surface type
  diagnostics 1
END            End analysis method information block
END            End information block

```

This file format is accepted when:

- The user loads an existing ensemble by clicking the **Load from File** button from the main UQ screen (Figure 61).

- The user creates a new ensemble by clicking the **Add New** button from the main UQ screen (Figure 61) and selecting the **Load all samples from a single file** radio button in the user's selection of sample generation (Figure 68).
- The user performs optimization under uncertainty from the main OUU screen (Figure 109) and selects the **Load Model From File** radio button for the user's model; for this file, the user does not need to specify the first block (i.e., the PSUADE.IO block).

This file format is written when:

- The user saves an existing ensemble by clicking the **Save Selected** button from the main UQ screen (Figure 61).

PSUADE Sample File Format

The following is an example of the sample file format. Comments in red do NOT appear in the file and are only for instructional purposes.

```

PSUADE_BEGIN      Start data block
5 2              5 samples, 2 inputs
1 4.0 -1.0      Sample index, input values for sample point 1
2 3.0 2.0      Sample index, input values for sample point 2
3 5.0 1.0      Sample index, input values for sample point 3
4 2.0 1.5      Sample index, input values for sample point 4
5 3.0 3.0      Sample index, input values for sample point 5
PSUADE_END      End data block

```

This file format is accepted when:

- The user creates a new ensemble by clicking the **Add New** button from the main UQ screen (Figure 61) and selecting the **Load all samples from a single file** radio button in the user's selection of sample generation (Figure 68).
- The user creates a new ensemble by clicking the **Add New** button from the main UQ screen (Figure 61) and selecting the **Choose sampling scheme** radio button in the user's selection of sample generation (Figure 66); in the **Distributions** tab, if the user designates an input variable's PDF to be of type "Sample", the "Param 1" field will generate a **Select File** button that prompts for the sample file representing the input's PDF.
- Similar to above, when the user enters Expert Mode within the Analysis dialog; within Expert Mode (Figure 95), the user can change the input distribution before performing response surface based analysis.
- The user performs optimization under uncertainty from the main OUU screen (Figure 109); if any of the variables are designated as random variables, the **UQ Setup** tab will be displayed

and any prompt for loading existing sample (e.g., “Load existing sample for Z3” or “Load existing sample for Z4”) will require this file format. (Currently, the **UQ Setup** tab is missing from the Figure because no variables have been designated as random.)

This file format is written when:

- The user wants to save the results of inference by clicking **Save Posterior Input Samples to File** within Bayesian Inference (Figure 65), which is accessible from the Analysis screen of UQ (Figure 63).

CSV (Comma Separated Values) File Format

The following is an example of the CSV file format. Comments in red do not appear in the file and are only for instructional purposes. CSV files can be easily generated using Excel and exporting in the .csv format

```
A0,A1,A2,A3,A4,Y1,Y2      Input variable names, then output variable names (if any)
-0.959,0.105,0.216,-2.16e-10,-0.256,-1.698,-0.783 Values for the
first sample (Output values are not required if not calculated)
-0.095,0.085,-0.980,-0.487,0.035,0.978,0.087 Values for the second sample
-0.698,-0.598,-0.970,0.025,0.819,-0.064,0.285
0.216,0.729,0.497,0.568,0.678,-0.471,-0.359
0.568,0.548,-0.226,0.387,-0.176,6.89e-03,-0.276
```

Variable names are specified in the first line, with input names and then output names. Output names can be specified, even if there is no data available for them yet. Data is only required for inputs. In addition, the variable names line is not required in those places where a PSUADE sample file is acceptable.

This file format is accepted when:

- The user loads an existing ensemble by clicking the **Load from File** button from the main UQ screen (Figure 61). Variable names are required.
- The user creates a new ensemble by clicking the **Add New** button from the main UQ screen (Figure 61) and selecting the **Load all samples from a single file** radio button in the user’s selection of sample generation (Figure 68).
- The user creates a new ensemble by clicking the **Add New** button from the main UQ screen (Figure 61) and selecting the **Choose sampling scheme** radio button in the user’s selection of sample generation (Figure 66); in the **Distributions** tab, if the user designates an input variable’s PDF to be of type “Sample”, the “Param 1” field will generate a **Select File** button that prompts for the sample file representing the input’s PDF.

- Similar to above, when the user enters Expert Mode within the Analysis dialog; within Expert Mode (Figure 95), the user can change the input distribution before performing response surface based analysis.
- The user performs optimization under uncertainty from the main OUU screen (Figure 109); if any of the variables are designated as random variables, the **UQ Setup** tab will be displayed and any prompt for loading existing sample (e.g., “Load existing sample for Z3” or “Load existing sample for Z4”) will require this file format. (Currently, the **UQ Setup** tab is missing from the Figure because no variables have been designated as random.)

5. SOLVENTFIT

The SolventFit module is an uncertainty quantification tool for full Bayesian calibration of an Aspen Plus solvent process model to experimental data. SolventFit may provide improved predictions with uncertainty bounds by accounting for uncertainty in model parameters and deficiencies in the model form. The result is a posterior distribution of parameters allowing for predictions with uncertainty. uses a custom BSS-ANOVA-based response surface for the outputs. Like the Bayesian inference module, the **SolventFit** algorithm [10] utilizes Markov Chain Monte Carlo (MCMC) to compute the posterior distributions, and uses a custom BSS-ANOVA-based response surface (emulators) that serves as a fast approximations to the actual simulation model.

SolventFit Instructions

To use **SolventFit**, the user will need to install R, as well as the R packages “MCMCpack”, “abind” and “MASS”. Please refer to the installation instructions in Section 8.3. Once R is installed, the user will also need to set the path to the RScript executable within FOCUS.

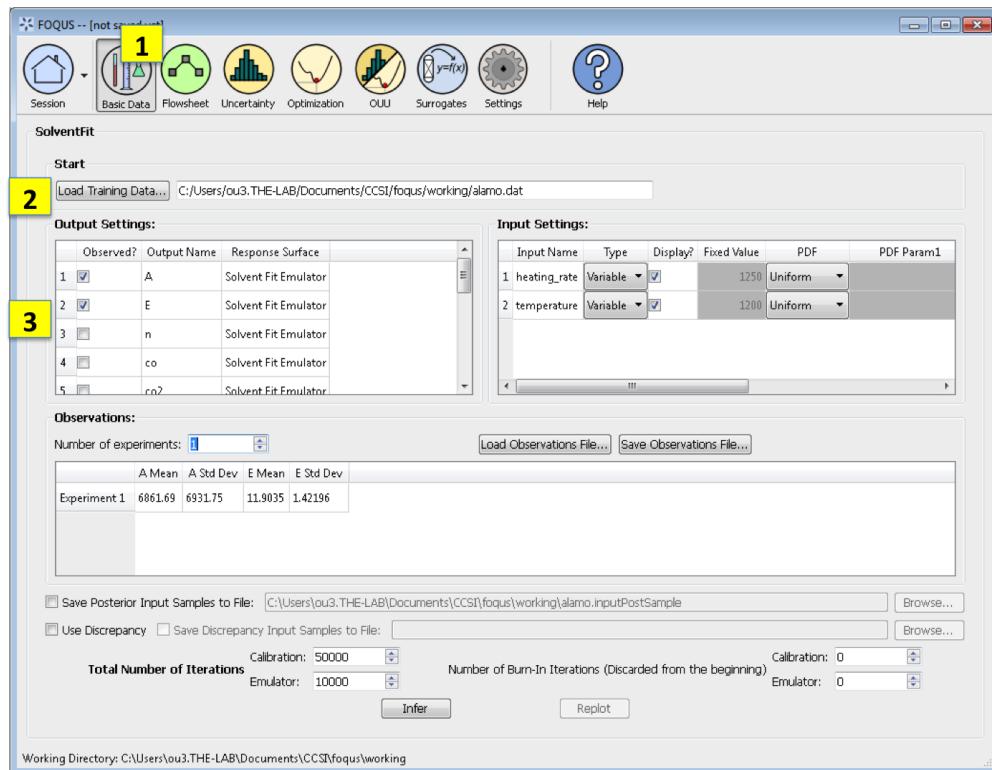


Figure 105: SolventFit Home Screen

1. **Basic Data.** From the FOCUS main screen, click the **Basic Data** button and select **SolventFit** to enter a SolventFit session.
2. **Load Training Data** loads the file of design and variable inputs and their relevant simulation outputs (from Aspen or other computer simulation code). This file usually has extension .txt, .dat, or .csv.

3. **Output Settings** lists the available outputs for analysis. The Output Name column lists the name of each output. The user can select/deselect outputs for analysis using the checkboxes in the Observed column. The response surface in SolventFit will be prepopulated with “SolventFit Emulator” because SolventFit uses its own custom BSS-ANOVA response surface model. The simulation ensemble is used as the training data for generating the response surfaces.

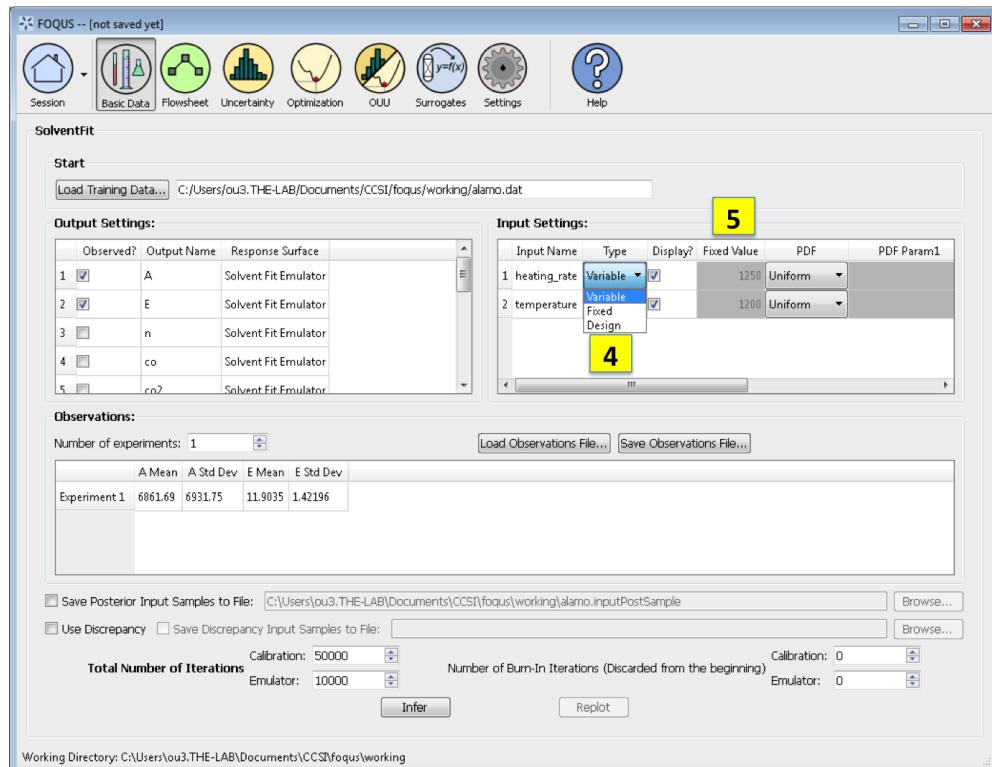


Figure 106: SolventFit Input Settings

4. **Input Settings** is populated with input variable information from the training data. Under the column, **Type**, the user can specify which inputs are fixed, design, or variable using the drop-down menu in the **Input Settings Table**. Selecting “Fixed” means that the input is fixed at its default value for all design points. Changing the type to “Variable” means that the input is a calibration parameter and is uncertain; therefore, its value varies between samples. Changing the type to “Design” means that the input is an experimental input with preselected values. In addition, the user can specify which inputs are displayed in the resulting plots of the posterior distributions. To omit specific inputs, clear the checkboxes from the **Display** column of the table. (The default is that once inference is completed, all inputs will be displayed in the plots.)
5. **Fixed Value.** With any fixed input, the only parameter that can be changed is the default value (i.e., all samples of this input are fixed at this default value).
6. **PDF.** With any variable input, the minimum/maximum values, as well as the probability distribution function (PDF), for that input can be changed. The default prior is specified to be Uniform. To change the prior distribution type (e.g., Normal, Lognormal, or Gamma),

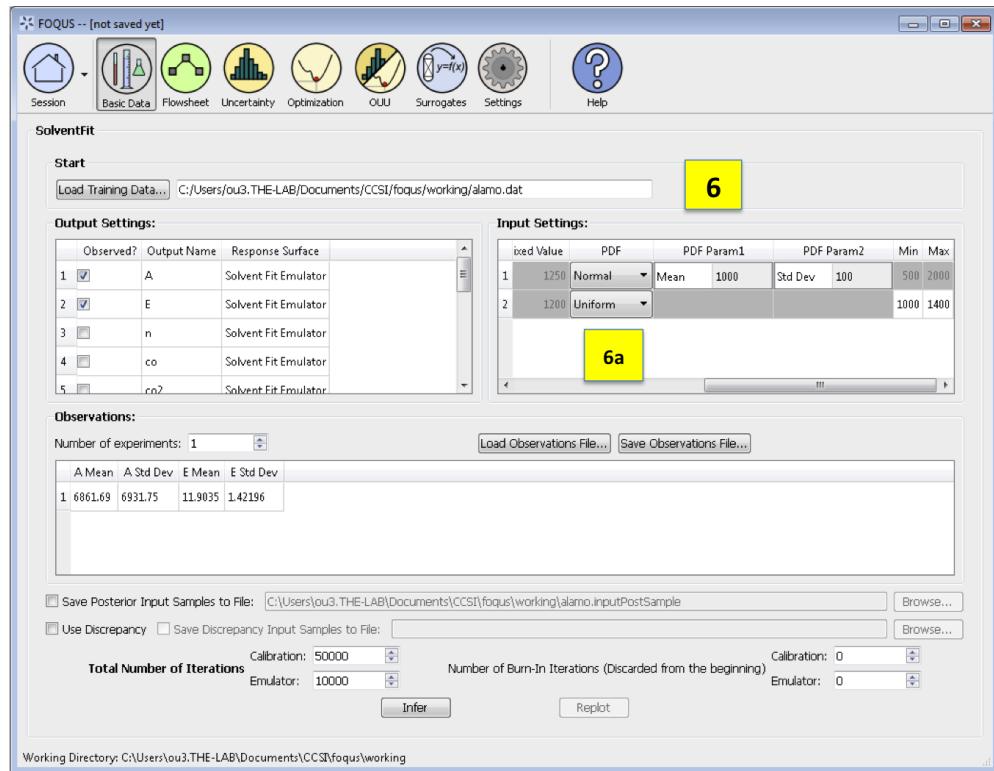


Figure 107: SolventFit Prior PDF for Inputs

use the drop-down list in the **PDF** column (box 6a) and enter corresponding values for the PDF parameters. To change the range of a uniform prior, scroll all the way to the right to modify **Min/Max**.

7. **Observations** section enables the user to add experimental data in the form of observations of certain output variables. At least one observation is required; the **number of experiments** may be changed using the pull down menu. For each observation, enter the mean and standard deviation (enter zero if there is no information about the noise) for all of the outputs. If any inputs are selected as design inputs, their values will also be required here. Currently, the observation noise model is assumed to be a normal distribution. Alternatively, the user can import the file of experiments using the **Load Observation File** button (7a). The user can also export the observations using the **Save Observation File** button (7b).
8. **Number of Iterations** are the number of iterations that the Markov Chain Monte Carlo (MCMC) is run for emulation and calibration. The default number of samples is set at 10000 for emulation and 50000 for calibration. Also the number of “burn-in” samples (number of initial samples to be thrown out) for both emulation and calibration may be changed from its default of 0 using the relevant button (8a).
9. **Use Discrepancy.** Check this box if the discrepancy should be estimated in the calibration model. It is usually good practice to include the discrepancy in the calibration analysis.
10. **Save Posterior Input Samples to File** checkbox, when selected, saves the posterior input samples as a PSUADE sample file (format described in Section 4.3). This file characterizes the input uncertainty as a set of samples. In addition, the user can save the discrepancy

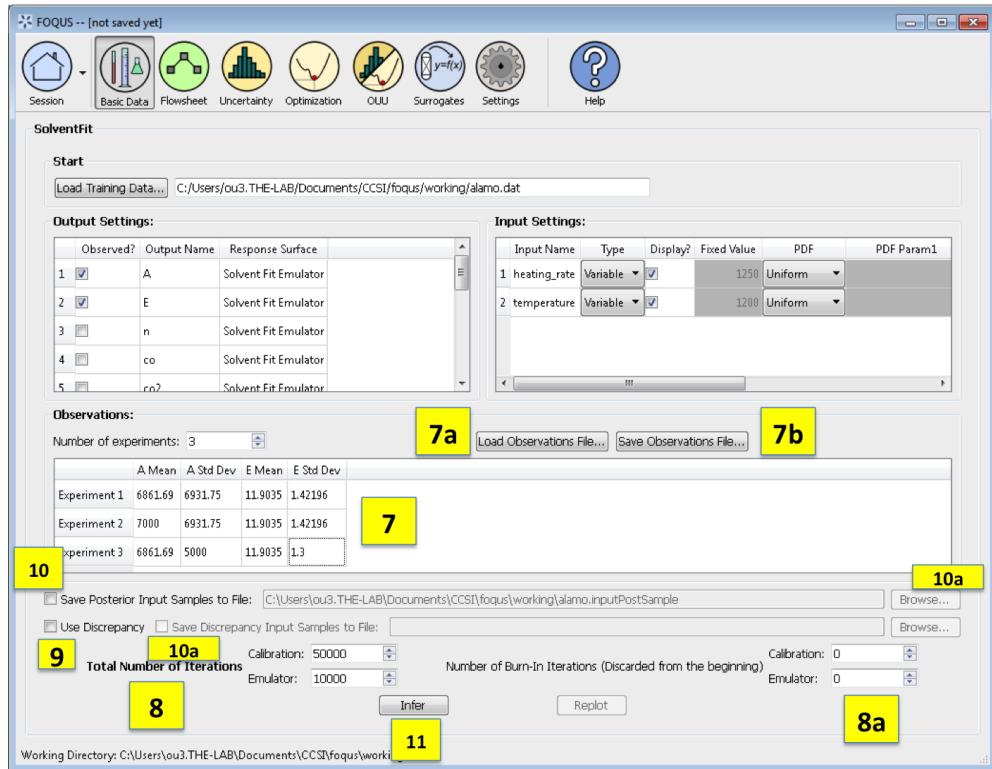


Figure 108: SolventFit Inference Screen

samples to a file by selecting the checkbox Save Discrepancy Input Samples to File (10a). If saving posterior and/or discrepancy samples to a file, click Browse to set the name and location of where this file is saved (10b).

11. Click **Infer** to start the analysis. (Note: If the inference returns an invalid posterior distribution (i.e., one with no samples), it usually means the prior distributions or that the observation data are not prescribed appropriately. In this case, it is recommended that the user experiment with different priors and/or data distribution means and/or standard deviations.)
12. The plotted results for SolventFit are posterior distributions of the selected variable inputs; they are similar to the plots from the Bayesian Inference in the Uncertainty Quantification module in Figure 104, see Section 4.2 on Bayesian Inference for more details.

6. OPTIMIZATION UNDER UNCERTAINTY

The FOCUS OUU module supports several variants of optimization under uncertainty. This chapter first presents the mathematical formulations of these variants. Subsequently, details of the OUU graphical user interface will be discussed.

OUU Variables

Suppose a simulation model is available for an OUU study. Let this simulation model be represented by the following function:

$$Y = F(Z_1, Z_2, Z_3, Z_4), \quad (6.1)$$

which is characterized by four types of variables:

1. Design/Decision/Optimization variables

- Notation: Z_1 with dimension n_1
- Definition: Design variables are continuous variables that may be bounded or unbounded. They are generally the set of optimization variables in a single-stage optimization or the set of outer optimization variables in the two-stage optimization.

2. Recourse/Operating variables

- Notation: Z_2 with dimension n_2
- Definition: Operating variables are optimization variables in the inner optimization for a given scenario (or realization) of the uncertain variables in a two-stage optimization.

3. Discrete uncertain variables

- Notation: Z_3 with dimension n_3
- Definition: Discrete variables are uncertain variables that have an enumerable set of states (called scenarios) such that each state is associated with a finite probability and the sum of probabilities for all the scenarios is equal to 1.

4. Continuous uncertain variables

- Notation: Z_4 with dimension n_4
- Definition: Continuous uncertain variables are associated with a joint probability distribution function from which a sample can be drawn to compute the basic statistics.

OUU Objective Functions

In the presence of uncertainties, OUU seeks to find the optimal solution in some statistical sense. For example, an optimization goal may be to find the design settings that minimizes the statistical mean of the system response. Other popular objective functions are:

1. a linear combination of statistical mean and standard deviation of some selected output,
2. probability of exceeding the best value is smaller than some percentage at any point in the design space (this is analogous to conditional value at risk).

Note that these metrics are defined in the design variable space - that is, at each iteration of an OUU algorithm, the selected metric will be computed for the decision point under consideration. Since the calculation of these statistical metrics requires a sample (possibly large), OUU can benefit from parallel computing capabilities (e.g., the Turbine gateway).

Mathematical Formulations

FOQUS supports two types of OUU methods: single-stage OUU and two-stage OUU. The main difference between single-stage and two-stage OUU is the presence of the recourse (or operational) variables. Strictly speaking, since recourse variables are generally hidden (they are only needed in the inner stage and their values are not used in the outer stage of two-stage OUU), the distinction between single-stage and two-stage OUU is not clear. Nevertheless, for the sake of clarity, we will describe details of each formulation separately. The current OUU does not support linearly or nonlinearly-constrained optimization.

Single-Stage Formulation

In this formulation, there is no recourse variable:

$$Y = F(Z_1, Z_3, Z_4) \quad (6.2)$$

and the optimization problem becomes:

$$\min_{Z_1} \Phi_{Z_3, Z_4} [F(Z_1, Z_3, Z_4)] \quad (6.3)$$

where $\Phi_{Z_3, Z_4}[F(Z_1, Z_3, Z_4)]$ is the statistical metric (one of the three options given above).

For example, if the objective function is the statistical mean, then the formulation becomes:

$$\min_{Z_1} \mathbf{E}_{Z_3, Z_4} [F(Z_1, Z_3, Z_4)] \approx \min_{Z_1} \sum_{j=1}^{n_3} \pi_j \left(\int F(Z_1, Z_3, Z_4) P(Z_4) dZ_4 \right) \quad (6.4)$$

where, again, n_3 is the number of scenarios for the discrete uncertain variables, π_j is the probability of the j -th scenario, and $P(Z_4)$ is the joint probability of the continuous uncertain variables.

Two-Stage Formulation

In this formulation all four types of variables are present. The objective function is given by:

$$\min_{Z_3, Z_4} \Phi_{Z_3, Z_4} \left[\min_{Z_2} F(Z_1, Z_2, Z_3, Z_4) \right]. \quad (6.5)$$

If the objective function is the statistical mean, the formulation becomes:

$$\begin{aligned} & \min_{Z_1} \mathbf{E}_{Z_3, Z_4} \left[\min_{Z_2} F(Z_1, Z_2, Z_3, Z_4) \right] \\ & \approx \min_{Z_1} \sum_{j=1}^{n_3} \pi_j \left(\int \left[\min_{Z_2} F(Z_1, Z_2, Z_3, Z_4) \right] P(Z_4) dZ_4 \right) \end{aligned} \quad (6.6)$$

Let

$$G(\mathbf{Z}_1, \mathbf{Z}_3, \mathbf{Z}_4) = \min_{Z_2} F(Z_1, Z_2, Z_3, Z_4). \quad (6.7)$$

Then the two-stage equation can be rewritten as:

$$\begin{aligned} & \min_{Z_1} \mathbf{E}_{Z_3, Z_4} [G(Z_1, Z_3, Z_4)] \\ & \approx \min_{Z_1} \sum_{j=1}^{n_3} \pi_j \left(\int G(Z_1, Z_3, Z_4) P(Z_4) dZ_4 \right) \end{aligned} \quad (6.8)$$

which is a single-stage OUU with respect to the G function.

OUU User Interface

The OUU module enables the user to perform optimization under uncertainty studies on a flowsheet. From the OUU tab, the user can set up the different types of optimization parameters, select from the different OUU options, and run the optimization. This screen is shown in Figure 109.

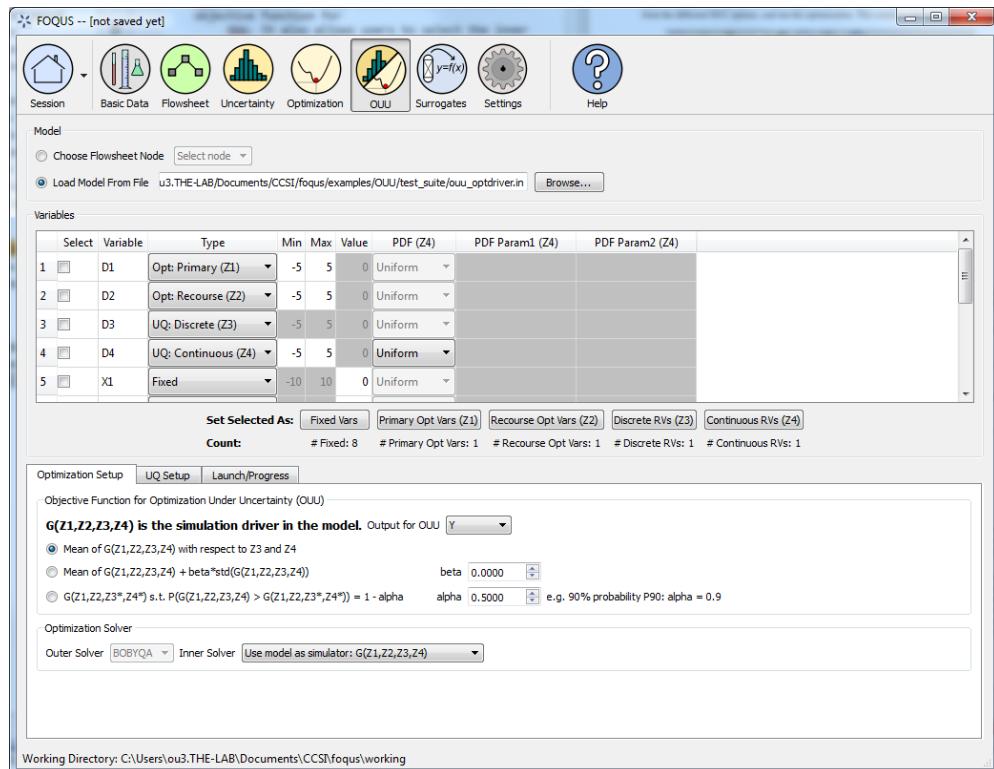


Figure 109: Optimization Under Uncertainty Screen

1. **Model** provides two options for setting up the model: (1) select a node from the flowsheet that has already been instantiated; or (2) load the model from a file in the PSUADE full file format (with the `opt_driver` variable set to the simulation executable.)
2. **Variables** displays all variables defined in the model that can be used in this context. Each available variable can be set to either one of the 5 types:

- fixed (the parameter's value is fixed throughout the optimization process)
 - primary (parameter for the outer optimization)
 - recourse (parameter for the inner optimization)
 - discrete (categorical uncertain parameter that contributes to scenarios)
 - continuous (continuous uncertain parameter with a given probability distribution)
3. **Optimization Setup** allows users to select the objective function for OUU. It also allows users to select the inner optimization solver. There are two options for the inner solver: (1) the simulation model provided by users is an optimizer itself, and (2) the simulation provided by users needs to be wrapped around by another optimizer in FOCUS.
 4. **UQ Setup** allows users to set up the continuous uncertain parameters. There are two options: (1) FOCUS can generate a sample internally, or (2) a user-generated sample can be loaded into FOCUS. The sample size should be larger than the number of continuous uncertain parameters. Optionally, response surface can be turned on to enable the statistical moments to be computed more accurately even with small samples. Users can also select a smaller subset of the sample for building response surfaces and evaluate the response surfaces with the larger samples.
 5. **Launch/Progress** has the ‘Run OUU’ button to launch OUU runs.

OUU Tutorial

This section walks through a few examples of running OUU.

Example 1: OUU with Discrete Uncertain Parameters Only

This example has only discrete uncertain parameters and the objective function is computed from the mean estimation with the scenarios from a sample file.

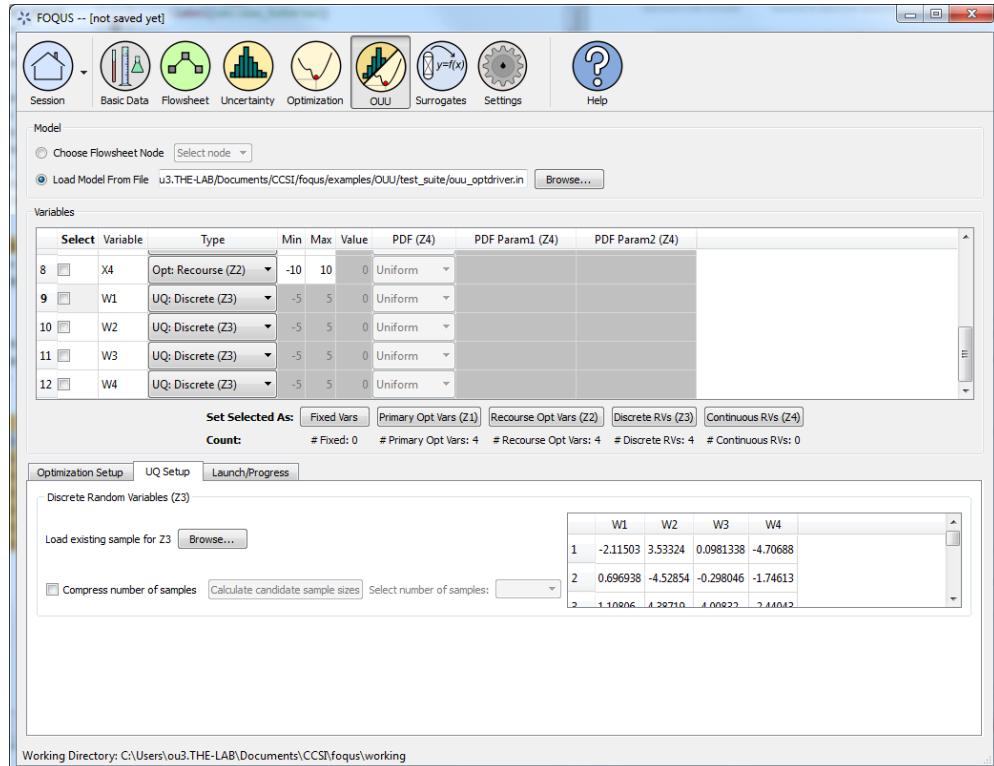


Figure 110: OUU Example with Discrete Uncertain Parameters

1. Start FOCUS and click the ‘OUU’ icon.
2. Under ‘Model’, browse and load examples/OUU/test_suite/ouu_optdriver.in.
3. Under ‘Variables’, set variable 1 – 4 as Z_1 , variable 5 – 8 as Z_2 , and variable 9 – 12 as Z_3 .
4. Under ‘Optimization Setup’, select the first objection function (default) and select ‘use model as optimizer’ as the ‘Inner Solver’.
5. Under ‘UQ Setup’ and ‘Discrete Random Variables’, browse the examples/OUU/test_suite directory and load the x3sample.smp sample file (see Figure 110).
6. Go to ‘Launch/Progress’ page, click ‘Run OUU’ and see OUU in action.

Example 2: OUU with Continuous Uncertain Parameters Only

This example has only continuous uncertain parameters and the objective function is computed from the mean estimation with a Latin hypercube sample of size 200 for Z_4 .

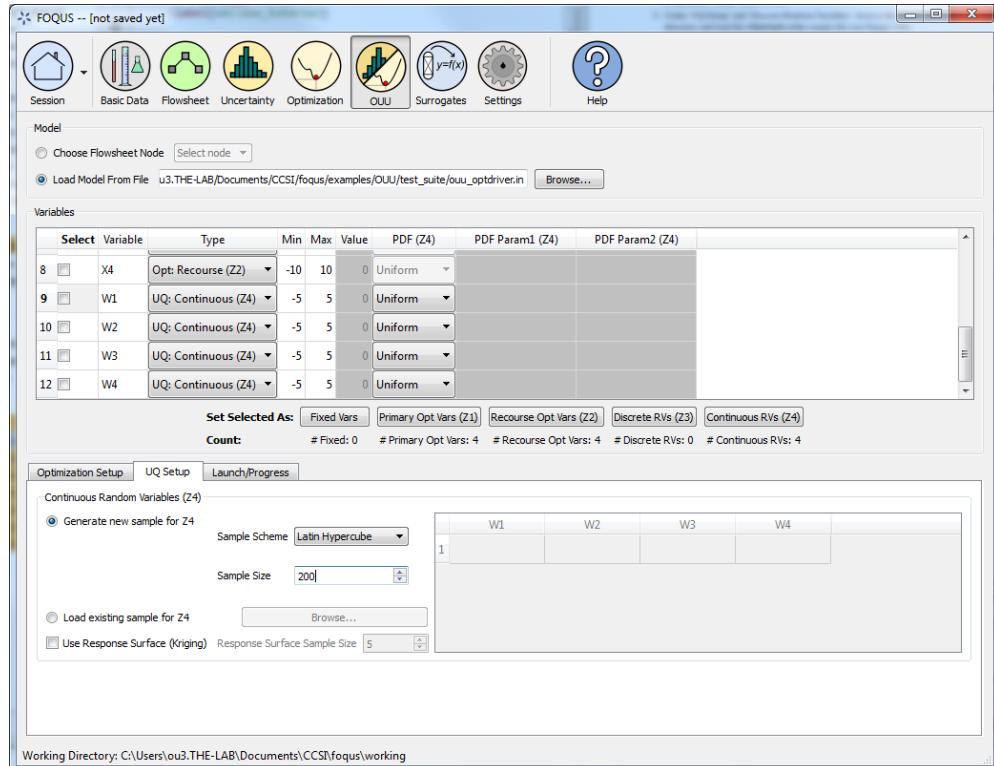


Figure 111: OUU Example with Continuous Uncertain Parameters

1. Start FOCUS and click the ‘OUU’ icon.
2. Under ‘Model’, browse and load `examples/OUU/test_suite/ouu_optdriver.in`.
3. Under ‘Variables’, set variable 1 – 4 as Z_1 , variable 5 – 8 as Z_2 , and variable 9 – 12 as Z_4 .
4. Under ‘Optimization Setup’, select the first objection function (default) and select ‘use model as optimizer’ as the ‘Inner Solver’.
5. Under ‘UQ Setup’ and ‘Continuous Random Variables’, select ‘Generate new sample for Z_4 ’, set ‘Sample Scheme’ to ‘Latin Hypercube’ and set sample size to 200 (see Figure 111).
6. Go to ‘Launch/Progress’ page, click ‘Run OUU’ and see OUU in action.

Example 3: OUU with Continuous Uncertain Parameters and Response Surface

This example is similar to Example 2 except that response surfaces will be used on the Z_4 sample (that is, the Z_4 sample will be used to construct response surfaces and the means will be estimated from a large sample evaluated on the response surfaces).

1. Start FOCUS and click the ‘OUU’ icon.
2. Under ‘Model’, browse and load `examples/OUU/test_suite/ouu_optdriver.in`.

3. Under ‘Variables’, set variable 1 – 4 as Z_1 , variable 5 – 8 as Z_2 , and variable 9 – 12 as Z_4 .
4. Under ‘Optimization Setup’, select the first objection function (default) and select ‘use model as optimizer’ as the ‘Inner Solver’.
5. Under ‘UQ Setup’ and ‘Continuous Random Variables’, select ‘Generate new sample for Z_4 ’, set ‘Sample Scheme’ to ‘Latin Hypercube’ and set sample size to 200.
6. Under ‘UQ Setup’ and ‘Continuous Random Variables’, check the ‘Use Response Surface’ box (see Figure 111).
7. Go to ‘Launch/Progress’ page, click ‘Run OUU’ and see OUU in action.

Example 4: OUU with Discrete and Continuous Uncertain Parameters

This example has both discrete and continuous parameters. The discrete scenarios will be loaded from a sample file. A Latin hypercube sample will be generated for the continuous variables.

1. Start FOCUS and click the ‘OUU’ icon.
2. Under ‘Model’, browse and load `examples/OUU/test_suite/ouu_optdriver.in`.
3. Under ‘Variables’, set variable 1 – 4 as Z_1 , variable 5 – 8 as Z_2 , variable 9 as Z_3 , and variable 10 – 12 as Z_4 .
4. Under ‘Optimization Setup’, select the first objection function (default) and select ‘use model as optimizer’ as the ‘Inner Solver’.
5. Under ‘UQ Setup’ and ‘Discrete Random Variables’, browse the `examples/OUU/test_suite` directory and load the `x3sample4.smp` sample file.
6. Under ‘UQ Setup’ and ‘Continuous Random Variables’, select ‘Generate new sample for Z_4 ’, set ‘Sample Scheme’ to Latin hypercube and set ‘Sample Size’ to 100.
7. Go to ‘Launch/Progress’ page, click ‘Run OUU’ and see OUU in action.

Example 5: OUU with Mixed Uncertain Parameters and Response Surface

This example is similar to Example 4 except that response surfaces will be used to estimate the means for the continuous uncertain variables.

1. Start FOCUS and click the ‘OUU’ icon.
2. Under ‘Model’, browse and load `examples/OUU/test_suite/ouu_optdriver.in`.
3. Under ‘Variables’, set variable 1 – 4 as Z_1 , variable 5 – 8 as Z_2 , variable 9 as Z_3 , and variable 10 – 12 as Z_4 .
4. Under ‘Optimization Setup’, select the first objection function (default) and select ‘use model as optimizer’ as the ‘Inner Solver’.
5. Under ‘UQ Setup’ and ‘Discrete Random Variables’, browse the `examples/OUU/test_suite` directory and load the `x3sample4.smp` sample file.

6. Under ‘UQ Setup’ and ‘Continuous Random Variables’, select ‘Generate new sample for Z_4 ’, set ‘Sample Scheme’ to Latin hypercube and set ‘Sample Size’ to 100.
7. Under ‘UQ Setup’ and ‘Continuous Random Variables’, check the ‘Use Response Surface’ box.
8. Go to ‘Launch/Progress’ page, click ‘Run OUU’ and see OUU in action.

Example 6: OUU with User-provided Samples and Response Surface

This example is similar to Example 4 except that a sample for Z_4 will be used (instead of the Latin hypercube sample generated internally).

1. Start FOQUS and click the ‘OUU’ icon.
2. Under ‘Model’, browse and load `examples/OUU/test_suite/ouu_optdriver.in`.
3. Under ‘Variables’, set variable 1 – 4 as Z_1 , variable 5 – 8 as Z_2 , variable 9 as Z_3 , and variable 10 – 12 as Z_4 .
4. Under ‘Optimization Setup’, select the first objection function (default) and select ‘use model as optimizer’ as the ‘Inner Solver’.
5. Under ‘UQ Setup’ and ‘Discrete Random Variables’, browse the `examples/OUU/test_suite` directory and load the `x3sample4.smp` sample file.
6. Under ‘UQ Setup’ and ‘Continuous Random Variables’, check ‘Load existing sample for Z_4 ’ and load the Z_4 sample `examples/OUU/test_suite/x4sample4.smp`.
7. Go to ‘Launch/Progress’ page, click ‘Run OUU’ and see OUU in action.

Example 7: OUU with Large User-provided Samples and Response Surface

This example is similar to Example 5 except that a sample for Z_4 is provided (instead of generated internally).

1. Start FOQUS and click the ‘OUU’ icon.
2. Under ‘Model’, browse and load `examples/OUU/test_suite/ouu_optdriver.in`.
3. Under ‘Variables’, set variable 1 – 4 as Z_1 , variable 5 – 8 as Z_2 , and variable 9 – 12 as Z_4 .
4. Under ‘Optimization Setup’, select the first objection function (default) and select ‘use model as optimizer’ as the ‘Inner Solver’.
5. Under ‘UQ Setup’ and ‘Continuous Random Variables’, check ‘Load existing sample for Z_4 ’ and load the Z_4 sample `examples/OUU/test_suite/x4sampleLarge.smp` (10000 sample points).
6. Under ‘UQ Setup’ and ‘Continuous Random Variables’, check ‘Use Response Surface’ and set ‘Sample Size’ to 100.
7. Go to ‘Launch/Progress’ page, click ‘Run OUU’ and see OUU in action.

7. HEAT INTEGRATION

Plug-in Models

Heat Integration

The Heat Integration Node performs heat integration calculations for the entire process in the meta-flowsheet. This node transfers simulation results to a GAMS-based heat integration program, solves the program, and transfers part of important heat integration results from GAMS to the graphical interface. The detailed heat integration results can be found in \gams\HeatIntegration.lst. All input and output variables in the nodes connected to the Heat Integration Node (with the edge pointing to the heat integration node) are automatically transferred to heat integration. However, only the variables with heat integration tags are considered and processed in heat integration. Heat integration tags are described in the Heat Integration Tutorial in Section 7.2.

The Heat Integration Node Editor widget is illustrated in Figure 112 and Figure 113. To specify a node as a Heat Integration Node, expand the **Model** section, expand the **Model** drop-down list, and then select “heat_integration.” Some input variables control the performance of heat integration. Output variables display heat integration results, which are used as final results or inputs for steam cycle calculations.

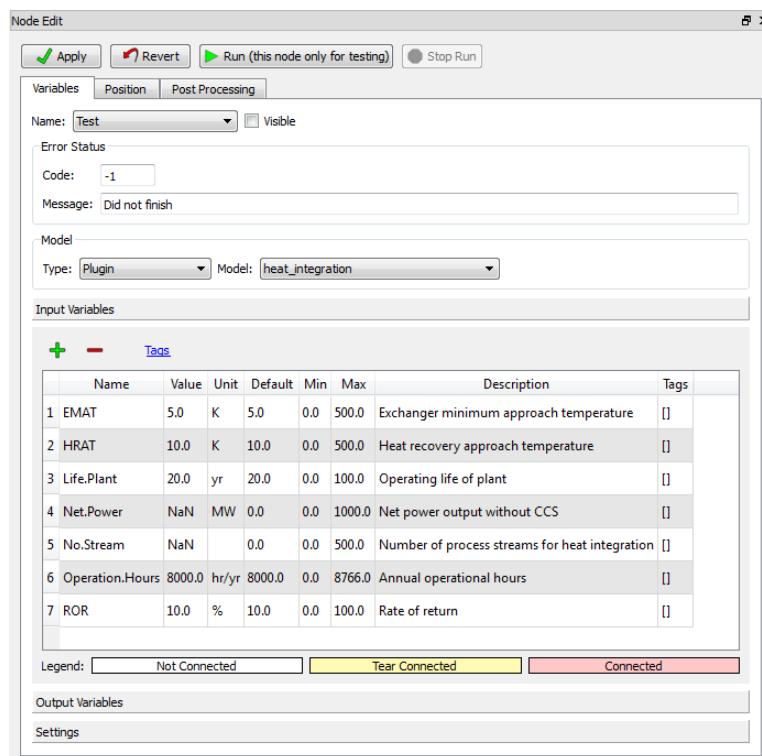


Figure 112: Heat Integration Node Editor (Input Variables)

The Heat Integration Node **Input Variables** are listed below:

- Exchanger minimum approach temperature (**EMAT**) is the minimum temperature difference between hot streams and cold streams for heat exchanger area calculations. The value of EMAT is usually not larger than that of HRAT (see below). The default value of EMAT is set to 5K. When a small value of HRAT is chosen (e.g., 3-5K), the value of EMAT should also be small (e.g., 1-3K).
- Heat recovery approach temperature (**HRAT**) is the minimum temperature difference between hot streams and cold streams for utility consumption calculations. Smaller HRAT usually leads to lower utility consumptions but higher capital cost of the heat exchanger network. An appropriate value should be selected for HRAT in order to achieve the best economic performance. The typical value of HRAT in industrial applications is 10K, which is also the default value of HRAT. The user may choose other HRAT values depending on their applications (e.g., in power plants, HRAT can be as low as 3-5K for higher heat recovery).
- **Life.Plant** is the operating life of the plant. The default value is 20 years. The user can change the value depending on detailed applications.
- **Net.Power** is the net power output of a power plant without carbon capture and storage (CCS). This is an optional input variable; therefore, it has no default value. The user only needs to assign a value to it when the steam cycle node is also present in the flowsheet. The net power output is used to calculate the amount of heat recovered for steam cycle via the heat exchanger network.
- **No.Stream** is the number of process streams (for heat integration) in the plant. This is an optional input variable without default value. When the value of “No.Stream” is not assigned, the number of process streams is calculated by the Python program via heat integration tags (see the Heat Integration Tutorial in Section 7.2).
- **Operation.Hours** is the annual operational hours of the plant. For power, refinery, and chemical industries, the typical value is 7500-8500 hours. The default value here is 8000 hours.
- **ROR** is the rate of return for the project. The default value is 10 percent. The user should select appropriate values for ROR based on their own projects. The variables **Life.Plant**, **No.Stream**, **Operation.Hours**, and **ROR** are used for the cost calculation.

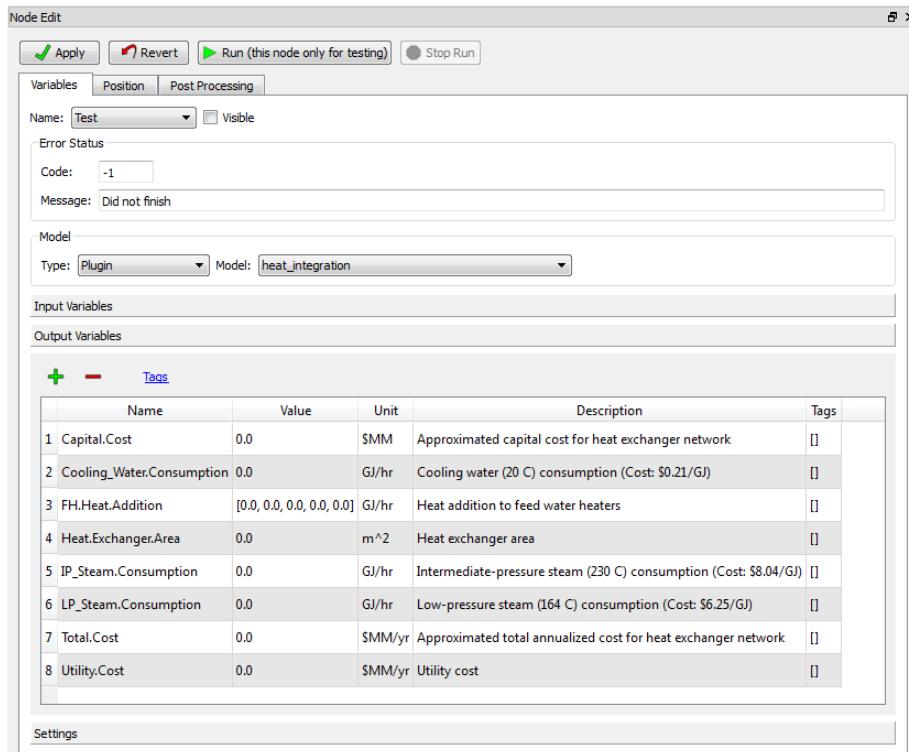


Figure 113: Heat Integration Node Editor (Output Variables)

The **Output Variables** of the Heat Integration Node are listed below:

- **Capital.Cost** is the minimum approximated capital cost for the heat exchanger network (in million U.S. dollars). The capital cost is calculated from the heat exchanger area (see below).
- **Cooling.Water.Consumption** is the minimum cooling water consumption rate (represented by energy rate GJ/hr) predicted by heat integration. Cooling water is an outside utility with the temperature of 20°C.
- **FH.Heat.Addition** is the amount of heat recovered to feed water heaters in the steam cycle. This variable is only valid when steam cycle calculations are also presented. If more heat is added to the steam cycle, higher net power output is expected. The value of “FH.Heat.Addition” is a vector with five elements. The first to fifth element of the vector represents the amount of heat added to the first-stage to the fifth-stage of the feed water heater in the steam cycle. The temperature of the feed water increases from the first-stage to the fifth-stage; therefore, heat added to the feed water heaters achieves a higher efficiency power generation from the first-stage to the fifth-stage heater. The temperature range for the five feed water heaters are 34-65°C, 65-96°C, 96-128°C, 128-160°C, and 160-195°C. These values are obtained from a supercritical power plant simulation model in Thermoflex.
- **Heat.Exchanger.Area** is the minimum heat exchanger area (in square meters), which is used for the capital cost calculation.

- **IP Steam Consumption** is the minimum intermediate-pressure steam consumption rate (in GJ/hr). Note: Both intermediate- and low-pressure steam are treated as a utility in heat integration calculations here; however, they are actually extracted from the steam cycle in the power plant. Therefore, minimizing steam consumption is equivalent to maximizing the net power output. Intermediate-pressure steam is extracted from the crossover of the pressurized intermediate-pressure turbine (PIPT) and the intermediate-pressure turbine (IPT) with the temperature of 230°C.
- **LP Steam Consumption** is the minimum low-pressure steam consumption rate (in GJ/hr). Low-pressure steam is extracted from the crossover of IPT and low-pressure steam turbine (LPT) with the temperature of 164°C.
- **Total.Cost** is the minimum approximated total annualized cost for the heat exchanger network (in million U.S. dollars per year), which equals the sum of utility cost and annualized capital cost.
- **Utility.Cost** is the minimum utility cost (in million U.S. dollars per year), which equals the sum of the cost of cooling water, intermediate-pressure steam, and low-pressure steam. It can also be treated as scaled total utility consumption where the consumption rate of each utility is weighted by its cost.

Steam Cycle

The Steam Cycle Node performs steam cycle and power output calculations for a power plant with CCS (and possibly heat integration). Correlations for net power output with steam extraction and heat addition to feed water heaters, which are obtained from a supercritical power plant model in Thermoflex, are utilized to calculate net power output and net efficiency with CCS in the Steam Cycle Node. These correlations are currently hard coded in Python for this node. The users will have a choice to provide their own correlations in future versions of FOQUUS.

The Steam Cycle Node Editor widget is illustrated in Figure 114 and Figure 115.

To specify a node as a Steam Cycle Node, expand the **Model** section, click on the **Model** drop-down list, and then select “steam_cycle.” All input variables (potentially) can be contributed to power output calculations; however, not all input variables are required to have a value assigned, except net power output and net efficiency without CCS. Output variables describe effects of CCS and heat integration to net power output and net efficiency.

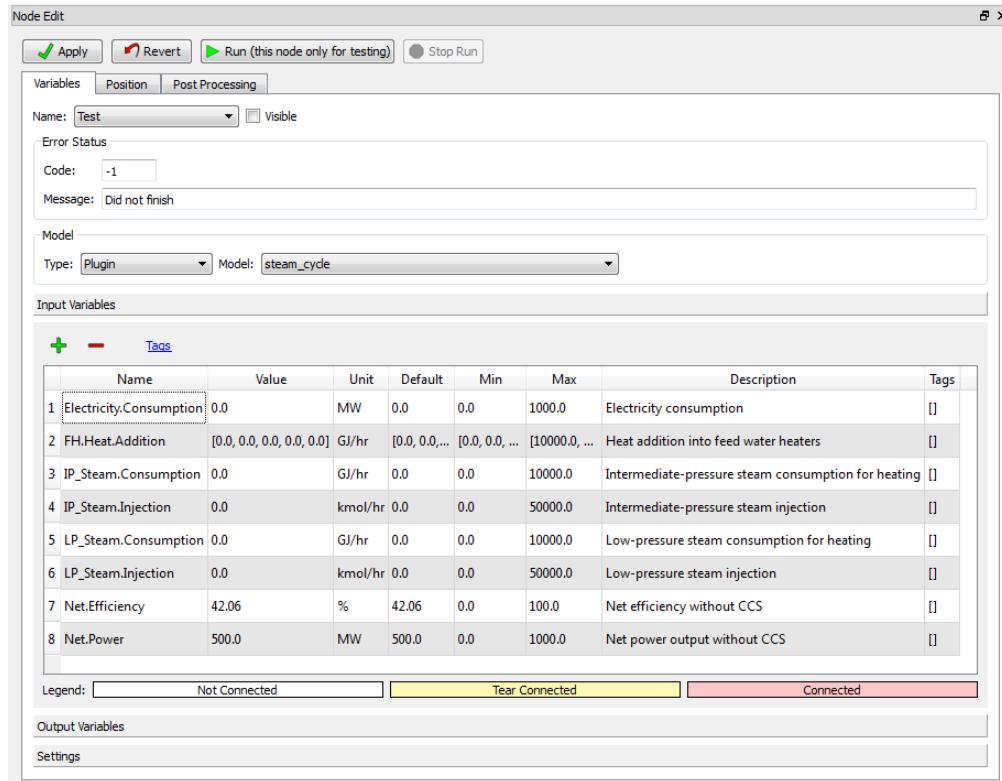


Figure 114: Steam Cycle Node Editor (Input Variables)

The **Input Variables** of the Steam Cycle Node are described below:

- **Electricity.Consumption** is the total electricity consumption in all processes other than steam cycle. The input value of this variable can be provided by the user or transferred from simulation outputs.
- **FH.Heat.Addition** is the amount of heat recovered to feed water heaters in steam cycle. The input value of this variable can be transferred from heat integration output.
- **IP_Steam.Consumption** is the intermediate-pressure steam consumption rate in heat exchangers. It is usually provided by heat integration, and sometimes it can be directly provided by simulation.
- **IP_Steam.Injection** is the intermediate-pressure steam injection rate to process streams. In some equipment, such as regenerators in the capture process, steam needs to be injected directly into the input stream to provide a large amount of heat and realize fast heat transfer. The steam injection rate is different from the steam consumption rate as it does not need heat exchangers and is not considered in heat integration. This variable is typically provided by simulation output.
- **LP_Steam.Consumption** is the low-pressure steam consumption rate in heat exchangers provided by heat integration or simulation output.

- **LP_Steam.Injection** is the low-pressure steam injection rate to process streams provided by simulation output.
- **Net.Efficiency** is the net efficiency of the power plant without CCS. Its default value is 42.06 percent, which is the efficiency of a typical supercritical pulverized coal-fired power plant without CCS. The user should change the value when another type of power plant is applied.
- **Net.Power** is the net power output of a power plant without CCS. The user must give an input to this variable to perform steam cycle calculations. Both **Net.Efficiency** and **Net.Power** provide base case values for a power plant without CCS and heat integration.

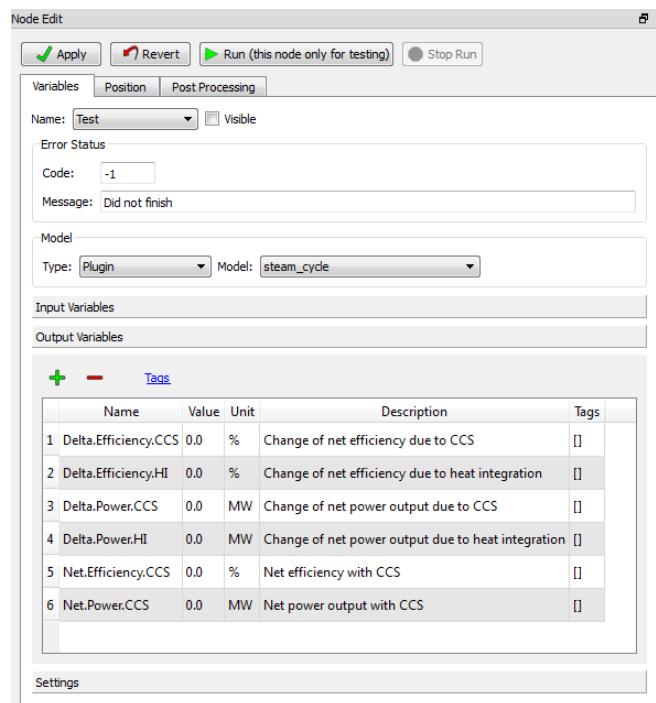


Figure 115: Steam Cycle Node Editor (Output Variables)

The **Output Variables** of the Steam Cycle Node are listed below:

- **Delta.Efficiency.CCS** is the change of the net efficiency of a power plant with CCS compared to the base case value. This variable is expected to be negative since CCS decreases the net power output to a certain degree.
- **Delta.Efficiency.HI** is the change of the net efficiency of a power plant with heat integration compared to the base case value. This variable is expected to be positive since heat integration potentially increases the net power output.
- **Delta.Power.CCS** is the change of the net power output of a power plant with CCS compared to the base case value.

- Delta.Power.HI is the change of the net power output of a power plant with CCS compared to the base case value.
- Net.Efficiency.CCS is the net efficiency of the power plant with CCS given the base case value.
- Net.Power.CCS is the net power output of the power plant with CCS assigned as the base case value.

Heat Integration Tutorial

This tutorial demonstrates the use of FOQUS to perform heat integration with process simulations. This tutorial uses four models: (1) bubbling fluidized bed (BFB) model in ACM, (2) multi-stage compressor model in ACM, (3) heat integration model in GAMS, and (4) simplified steam cycle model in Python. Heat integration and steam cycle models are included in FOQUS as plug-in models (as described in Section 7.1). Because most detailed steps for adding/editing models, building flowsheets, and running simulations have already been covered in the previous section, they are skipped here.

Example ACM and JSON files are provided with the FOQUS installer and are typically installed at: C:\Program Files (x86)\foqus\foqus_2014.10.0\examples\Heat_Integration\Model_Files. Copy the example files to a convenient location.

This tutorial is divided into 10 major steps.

GAMS Pre-Settings (Windows System Only)

If the user is working on a Windows system, the GAMS installation directory needs to be added into the system path. Navigate to the Windows **Control Panel**, select **System and Security**, select **System**. Select **Advanced system settings** in the left side bar. In the **System Properties** window, click **Environment Variables**. In the **Environment Variables** window, navigate to the **System variables** box, double-click the variable **Path**. Add the GAMS installation directory to the **Variable value** box. An example of the GAMS directory is: C:\GAMS\win64\24.2\. The user needs to modify the above path to the exact location where GAMS is installed.

If the user is using GAMS in the first time (after installation or re-installation), double-click the GAMS icon on the desktop. A prompt dialog displays, asking the user to select default solvers. Click **OK** to accept all default solver selections. Close GAMS.

Open a GAMS Project (Windows System Only)

Suppose the current directory is the user's working directory (e.g., C:\FOQUS). If the user is working on a Windows system, navigate to the directory "gams", and double-click the file "Heat-Integ.gpr." This will open the file in GAMS and make its path the current GAMS working directory. Close GAMS. All GAMS input and output files will be stored in this directory. This step is not necessary with a Linux system. All GAMS files should be closed when running heat integration.

Start a New Session

Start FOCUS. Start a new session. In the “Session Information” screen, under the **Metadata** tab, enter “BFB_CP_HI_SC” in the **Session Name** field (Figure 116). Save the session.

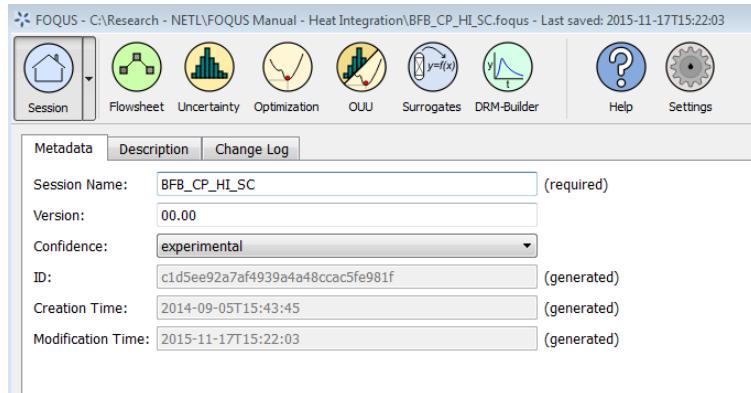


Figure 116: Start a New Session

Upload Simulation Models

The ACM BFB and compressor model files need to be uploaded to the turbine.

Choose **Add/Update Model to Turbine**. In the **Turbine Upload** dialog, upload the BFB ACM model file (Figure 117). The SimSinter configuration file for the BFB model is BFB_3ads_2rgn.json. Enter “BFB_3ads_2rgn” in the **Simulation Name** drop-down list.

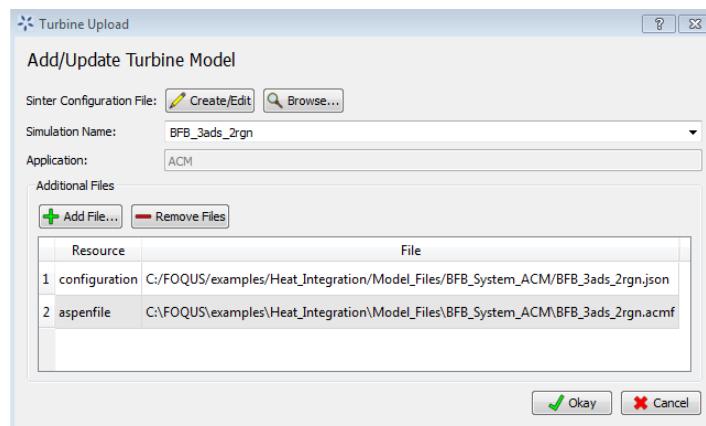


Figure 117: Upload Simulation Models

Upload the compressor ACM model as above. The SimSinter configuration file for the compressor model is CompIG.json. The simulation name for the compressor model is “Comp.”

Flowsheet Setup

Navigate to the **Flowsheet** window and build the flowsheet for this heat integration example. The flowsheet for this tutorial is shown in Figure 118. Build the flowsheet in the same way as the figure. The nodes **BFB**, **Compressor**, **Heat Integration**, and **Steam Cycle** correspond to the BFB simulation in ACM, compressor simulation in ACM, heat integration in GAMS and steam cycle calculations in Python. Models “BFB_3ads_2rgn,” “Comp,” “heat_integration,” and “steam_cycle” are assigned to nodes **BFB**, **Compressor**, **Heat Integration**, and **Steam Cycle**, respectively. Model **BFB** and **Compressor** are Turbine gateway models (see below); Model **Heat Integration** and **Steam Cycle** are plugin models. The node **Total Consumption** is used to calculate the total electricity and steam consumptions for BFB and compressor process. It is a user-specified Python calculation node and is described later. All edges should be the same directions as those in the figure.

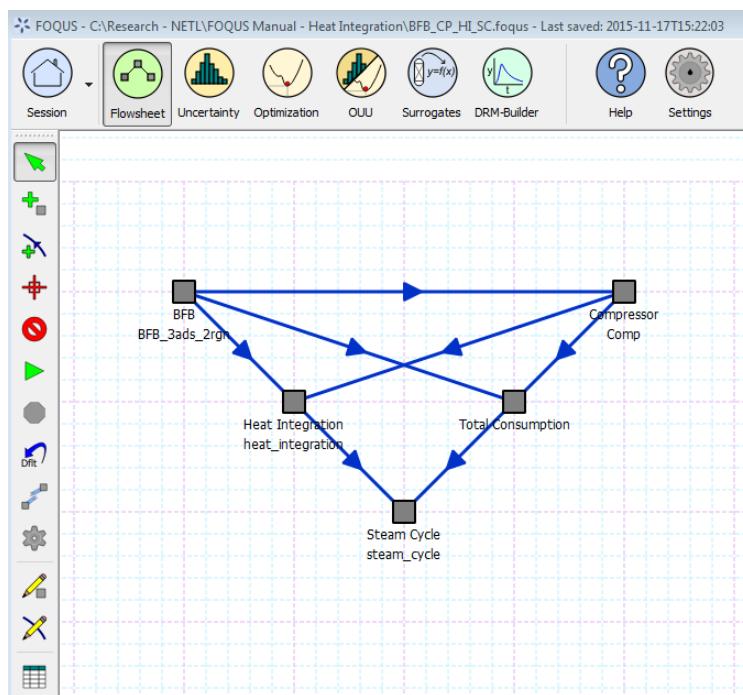


Figure 118: Flowsheet of Heat Integration Example

Edit Nodes

- **BFB and Compressor Nodes:** Figure 119 illustrates the edit of the **BFB** Node. From the “Node Edit” dialog box, select “Turbine” from the **Type** drop-down list, and “BFB_3ads_2rgn” from the **Model** drop-down list. Next, select the **Compressor Node**. Select “Turbine” from the **Type** drop-down list, and “Comp” from the **Model** drop-down list. For both the **BFB** and **Compressor Nodes**, **Heat Integration Tags** should be added to their corresponding output variables. The detailed description for heat integration tags is covered later. No other changes are required for the two nodes.

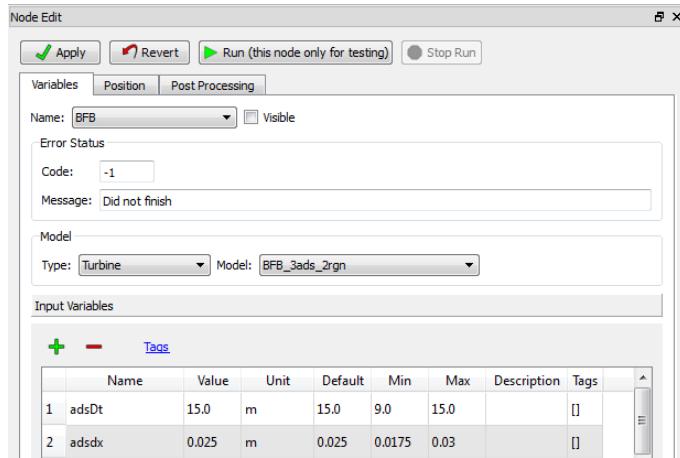


Figure 119: BFB Node Editor

- **Total Consumption Node:** The user is required to define the **Total Consumption Node**. The Node Editor for the **Total Consumption Node** is illustrated in Figures 120, 121, and 122. Within the Node Edit dialog box, choose “None” in the Type drop-down list and leave the Model drop-down list blank.

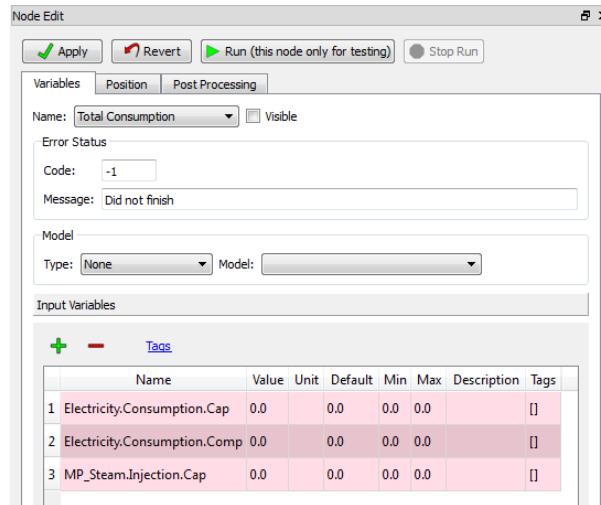


Figure 120: Total Consumption Node Editor (Input Variables)

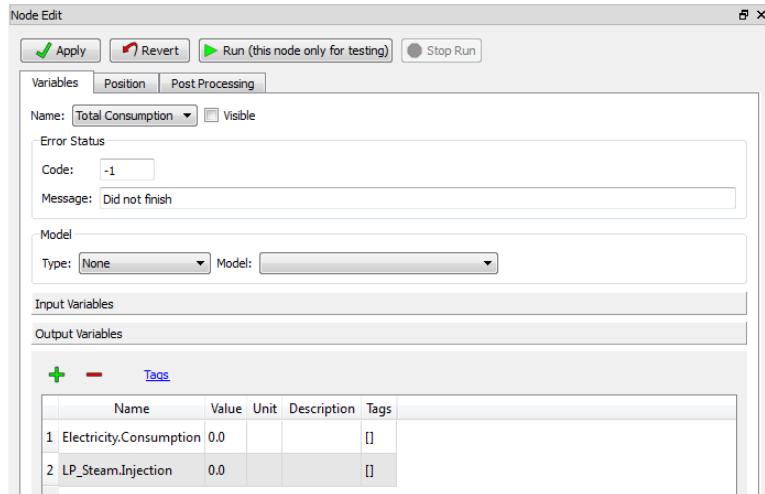


Figure 121: Total Consumption Node Editor (Output Variables)

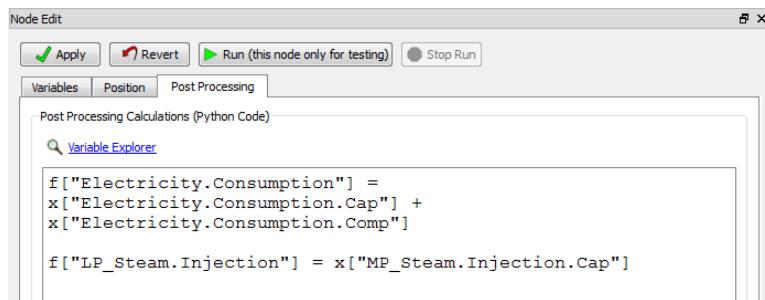


Figure 122: Total Consumption Node Editor (Python Codes)

1. In the **Input Variables** section (Figure 120): add the following three variables: (1) “Electricity.Consumption.Cap,” (2) “Electricity.Consumption.Comp,” and (3) “MP_Steam.Injection.Cap.” The value and default value should remain 0.0 for these variables. The category should be “fixed.” The user can optionally enter the unit, minimum/maximum, and description. These variables are linked to corresponding output variables of node **BFB** and **Compressor**.
2. In the **Output Variables** area of the **Node Edit** dialog box (Figure 121), add two variables: (1) “Electricity.Consumption” and (2) “LP_Steam.Injection.” The value should remain 0.0. The user can optionally enter the unit and description. These variables are linked to corresponding input variables of the **Steam Cycle Node**.
3. From the **Node Edit** dialog box (Figure 122), click the **Node Script** tab. Enter the following Python code in the dialog to perform node calculations:

```

f["Electricity.Consumption"] = \
x["Electricity.Consumption.Cap"] \
+ x["Electricity.Consumption.Comp"]
f["LP_Steam.Injection"] = \

```

```
x [ "MP_Steam.Injection.Cap" ]
```

- **Heat Integration Node:** Select “heat_integration” from the Type drop-down list in the Model section on the Node Edit dialog box. Only input variables need to be edited. Change the value of “HRAT” to 5.0 for higher heat recovery. Change the value of “EMAT” to 2.0. Enter the value of “Net.Power” as 650.33, which is the base case net power output. Leave all other fields unchanged. The Node Editor for the Heat Integration Node is shown in Figure 123.

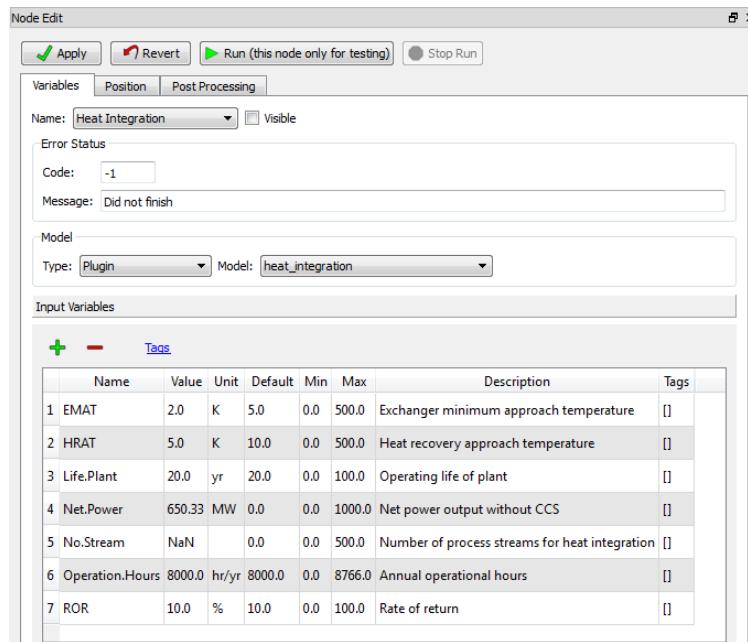
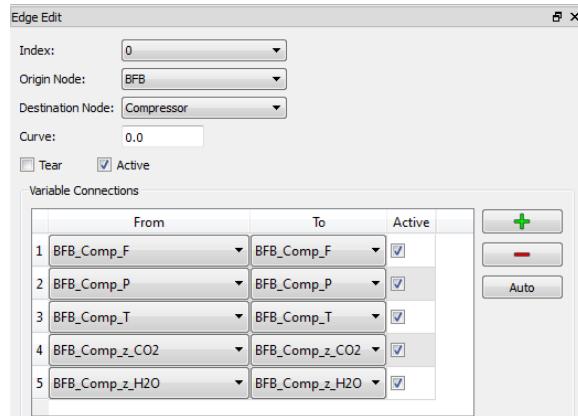
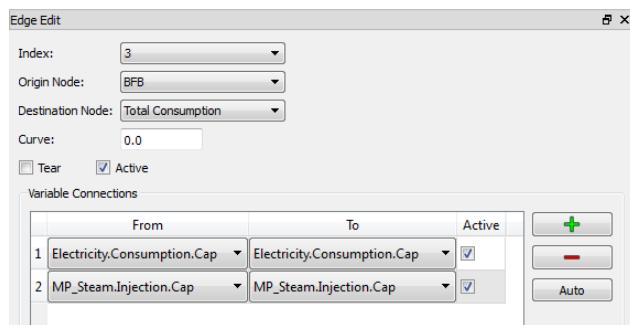
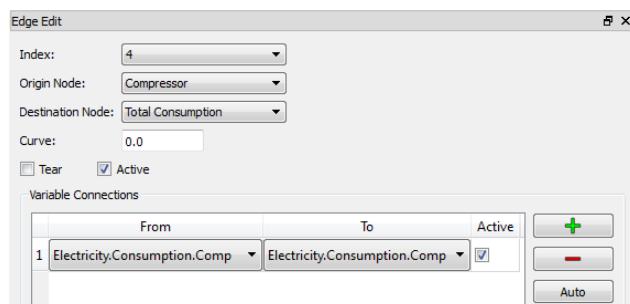


Figure 123: Heat Integration Node Editor

- **Steam Cycle Node:** Select “steam_cycle” from the Type drop-down list in the Model section on the Node Edit dialog box and leave all other fields unchanged.

Edit Edges

The user needs to specify variable connections in edges. In this example, all input and output variables that need to be connected have the same names; therefore, simply choose Auto to connect all linking variables. The editor for Edge 0 (BFB → Compressor), 3 (BFB → Total Consumption), 4 (Compressor → Total Consumption), 5 (Heat Integration → Steam Cycle), and 6 (Total Consumption → Steam Cycle) are illustrated in Figure 124, Figure 125, Figure 126, Figure 127 and Figure 128, respectively. Edge 1 (BFB → Heat Integration) and 2 (Compressor → Heat Integration) have no variable connections.

**Figure 124: Editor for Edge 0****Figure 125: Editor for Edge 3****Figure 126: Editor for Edge 4**

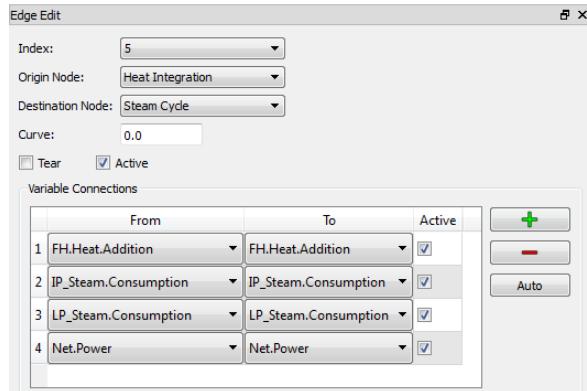


Figure 127: Editor for Edge 5

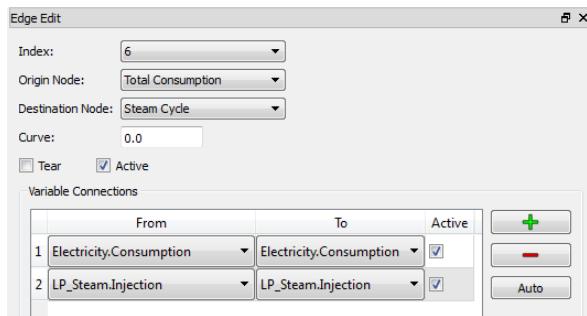


Figure 128: Editor for Edge 6

Add Heat Integration Tags

Heat integration tags are required if one variable needs to be considered in heat integration.

Four types of tags are needed for heat integration, and they identify (1) which block the variable is associated with, (2) which type of port the variable is in, (3) what type of variable it is, and (4) which kind of heat source the variable is involved in. The detailed lists of tags are provided in Tables 4, 5, 6, and 7.

Table 4: Tag 1: Block Name

Tag	Description	Note
“Block *”	* is the name of the block that the variable is associated with	

Table 5: Tag 2: Type of Port

Tag	Description	Note
“Port_Material_In”	Inlet material port	
“Port_Material_Out”	Outlet material port	
“Port_Heat_In”	Inlet heat port	
“Port_Heat_Out”	Outlet heat port	
“Blk_Var”	Block variable (not in any port)	

Table 6: Tag 3: Type of Variable

Tag	Description	Note
“T”	Temperature	
“Q”	Heat duty or heat flow rate	

Table 7: Tag 4: Type of Heat Source

Tag	Description	Note
“heater”	Simple heater or cooler with only one inlet and outlet stream	
“HX_Hot”	Hot side of heat exchanger with two inlet and outlet streams	
“HX_Cold”	Cold side of heat exchanger with two inlet and outlet streams	
“Point_Hot”	Isothermal heat source	Equipment removing heat to outside (e.g., adsorber)
“Point_Cold”	Isothermal heat sink	Equipment requiring heat from outside (e.g., regenerator)

Pick one tag from each type of tags for the variable as only one variable is considered in heat integration if all four types of tags are present.

For example, assume a variable has the following tags: “Block H1,” “Port_Material_In,” “T,” and “heater.” This means the variable is within Block H1, is the temperature of an inlet stream, and it is involved in the “heater” type heat source. The Python code

determines whether this variable is related to heat integration, and if yes, it then calculates the relevant heat integration inputs from this variable. If any one of the above tags is missing (e.g., only two or three tags are present), the variable is not included in heat integration; make sure that all four tags are properly added for heat integration variables.

In most cases, each variable should only have one tag within each different type of tags. This is true for block name, type of port, and type of variable; however, some variables may have two heat source tags. For example, the heat duty of a heat exchanger with two inlet and outlet streams is actually tagged as both “HX_Hot” and “HX_Cold.”

The minimum set of variables needed for heat integration is described below. For non-isothermal heat sources, including “heater,” “HX_Hot,” and “HX_Cold,” the user must provide the temperature of the inlet material port, temperature of outlet material port, and either equipment heat duty (block variable) or heat flow rate of inlet and outlet heat port. For isothermal heat sources, including “Point_Hot” and “Point_Cold,” the user needs to provide equipment temperature and heat duty (both of them are block variables).

In this example, some of the output variables in BFB and Compressor models are required to add heat integration tags. These variables have a description such as “Heat Integration,” so the user can easily find them.

The steps for adding heat integration tags for some related variable are illustrated below.

Take the variable “BFBadsB_Q” in the BFB model as an example (Figure 129):

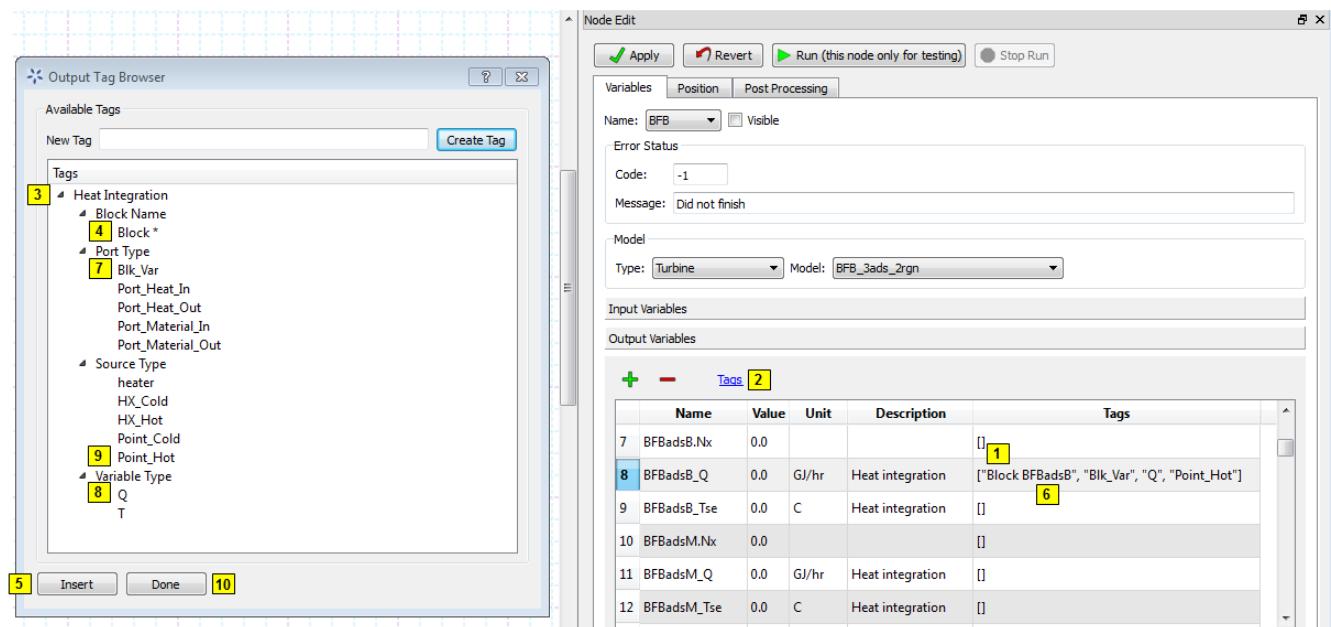


Figure 129: Procedures for Adding Heat Integration Tags

1. In the “Output Variables” dialog box, select variable “BFBadsB_Q.” Navigate to the **Tags** column and insert the cursor between the blank brackets.
2. Click **Tags** above **Name**. The Output Tag Browser displays.
3. Click ▷ on the left of **Heat Integration**. The list for four types of tags displays.
4. Click ▷ on the left of **Block Name**. The block name tag displays. Choose “Block *.”
5. Click **Insert**. The tag “Block *” is added to the tag list of variable BFBadsB_Q.
6. Navigate back to the **Tags** column in the Output Variables dialog box. Replace “*” with “BFBadsB.” Place the cursor after the entire tag (after the close quote).
7. Click ▷ on the left of **Port Type** and five port type tags are displayed. Select “Blk_Var” and click **Insert**.
8. Click ▷ on the left of **Variable Type** and two variable type tags are displayed. Select “Q” and click **Insert**.
9. Click ▷ on the left of **Source Type** and five source type tags are displayed. Select “Point_Hot” and click **Insert**.
10. Click **Done** to close the Output Tag Browser.

All four tags for variable BFBadsB_Q are now added. The user can also add four tags in a simultaneous way expand all tag types, press Ctrl and then select “Block *”, “Blk_Var”, “Q” and “Point_Hot” at the same time, and then click **Insert**. Heat integration tags for all other variables can be added in the same way.

The heat integration tags for BFB output variables are shown in Figures 130, 131 and, 132; the tags for Compressor output variables are shown in Figures 133, 134, and 135.

Output Variables					
	Name	Value	Unit	Description	Tags
8	BFBadsB_Q	143.5216...	GJ/hr	Heat integration ["Block BFBadsB", "Blk_Var", "Q", "Point_Hot"]	[]
9	BFBadsB_Tse	80.14054...	C	Heat integration ["Block BFBadsB", "Blk_Var", "T", "Point_Hot"]	[]
10	BFBadsM.Nx	641.1413...			[]
11	BFBadsM_Q	165.0984...	GJ/hr	Heat integration ["Block BFBadsM", "Blk_Var", "Q", "Point_Hot"]	[]
12	BFBadsM_Tse	83.63660...	C	Heat integration ["Block BFBadsM", "Blk_Var", "T", "Point_Hot"]	[]
13	BFBadsT.HXIn.F	76659.00...	kmol/hr		[]
14	BFBadsT.Nx	641.1413...			[]
15	BFBadsT_Q	175.4748...	GJ/hr	Heat integration ["Block BFBadsT", "Blk_Var", "Q", "Point_Hot"]	[]
16	BFBadsT_Tse	83.99672...	C	Heat integration ["Block BFBadsT", "Blk_Var", "T", "Point_Hot"]	[]
17	BFBrgnB.HXIn.F	389.2383...	kmol/hr		[]
18	BFBrgnB.Nx	7853.981...			[]
19	BFBrgnB_Q	-226.028...	GJ/hr	Heat integration ["Block BFBrgnB", "Blk_Var", "Q", "Point_Cold"]	[]
20	BFBrgnB_Tse	153.0096...	C	Heat integration ["Block BFBrgnB", "Blk_Var", "T", "Point_Cold"]	[]
21	BFBrgnT.HXIn.F	1000.664...	kmol/hr		[]
22	BFBrgnT.Nx	7853.981...			[]
23	BFBrgnT_Q	-581.077...	GJ/hr	Heat integration ["Block BFBrgnT", "Blk_Var", "Q", "Point_Cold"]	[]
24	BFBrgnT_Tse	138.6188...	C	Heat integration ["Block BFBrgnT", "Blk_Var", "T", "Point_Cold"]	[]
25	CMPads.GasOut.P	1.335419...	bar		[]
26	CMPads.J	1704.404...	kW		[]

Figure 130: Heat Integration Tags for BFB Output Variables (1)

Output Variables					
	Name	Value	Unit	Description	Tags
25	CMPads.GasOut.P	1.335419...	bar		[]
26	CMPads.J	1704.404...	kW		[]
27	CMPrgn.GasOut.P	1.229279...	bar		[]
28	CMPrgn.J	159.8628...	kW		[]
29	CW_SHXlean.HXIn.F	18208.97...	kmol/hr		[]
30	CW_SHXlean_Q	-266.153...	GJ/hr	Heat integration ["Block CW_SHXlean", "Blk_Var", "Q", "heater"]	[]
31	CW_SHXlean_SolidIn_T	90.14054...	C	Heat integration ["Block CW_SHXlean", "Port_Material_In", "T", "heater"]	[]
32	CW_SHXlean_SolidOut_T	65.0	C	Heat integration ["Block CW_SHXlean", "Port_Material_Out", "T", "heater"]	[]
33	Electricity.Consumption....	27.96400...	MW	Total consum...	[]
34	GasOut.F	5466.065...	kmol/hr		[]
35	GasOut.T	85.21273...	C		[]
36	GHXcmp.a_exch	1401.453...	m ²		[]
37	GHXcmp.GasIn.P	1.01325	bar		[]
38	GHXcmp.HXIn.F	40273.74...	kmol/hr		[]
39	GHXcmp_GasIn_T	141.3998...	C	Heat integration ["Block GHXcmp", "Port_Material_In", "T", "heater"]	[]
40	GHXcmp_GasOut_T	50.0	C	Heat integration ["Block GHXcmp", "Port_Material_Out", "T", "heater"]	[]
41	GHXcmp_Q	-588.665...	GJ/hr	Heat integration ["Block GHXcmp", "Blk_Var", "Q", "heater"]	[]
42	GHXfg.a_exch	5749.664...	m ²		[]
43	GHXfg.GasIn.P	1.01325	bar		[]

Figure 131: Heat Integration Tags for BFB Output Variables (2)

Output Variables					
	Name	Value	Unit	Description	Tags
41	GHXcmp_Q	-588.665...	GJ/hr	Heat integration	["Block GHXcmp", "Blk_Var", "Q", "heater"]
42	GHXfg.a_exch	5749.664...	m2		[]
43	GHXfg.GasIn.P	1.01325	bar		[]
44	GHXfg.HXin.F	28737.64...	kmol/hr		[]
45	GHXfg_GasIn_T	54.0	C	Heat integration	["Block GHXfg", "Port_Material_In", "T", "heater"]
46	GHXfg_GasOut_T	40.0	C	Heat integration	["Block GHXfg", "Port_Material_Out", "T", "heater"]
47	GHXfg_Q	-420.046...	GJ/hr	Heat integration	["Block GHXfg", "Blk_Var", "Q", "heater"]
48	LR_SHXlean_Q	-724.502...	GJ/hr	Heat integration	["Block LR_SHXlean", "Blk_Var", "Q", "heater"]
49	LR_SHXlean_SolidIn_T	158.4834...	C	Heat integration	["Block LR_SHXlean", "Port_Material_In", "T", "heater"]
50	LR_SHXlean_SolidOut_T	90.14054...	C	Heat integration	["Block LR_SHXlean", "Port_Material_Out", "T", "heater"]
51	LR_SHXrich_Q	724.5029...	GJ/hr	Heat integration	["Block LR_SHXrich", "Blk_Var", "Q", "heater"]
52	LR_SHXrich_SolidIn_T	80.14054...	C	Heat integration	["Block LR_SHXrich", "Port_Material_In", "T", "heater"]
53	LR_SHXrich_SolidOut_T	145.7366...	C	Heat integration	["Block LR_SHXrich", "Port_Material_Out", "T", "heater"]
54	MP_Steam.Consumption...	807.1050...	GJ/hr	Total consum...	[]
55	MP_Steam.Injection.Cap	10500.0	kmol/hr	Total consum...	[]
56	removalCO2	0.664852...			[]
57	removalH2O	0.288917...			[]
58	rhos	442.0	kg/m3		[]
59	status	0.0		Simulation Sta...	[]

Figure 132: Heat Integration Tags for BFB Output Variables (3)

Output Variables					
	Name	Value	Unit	Description	Tags
1	Absorber_Q	-1.4330...	GJ/hr	Heat integration	["Block Absorber", "Blk_Var", "Q", "Point_Hot"]
2	Absorber_T	33.4982...	C	Heat integration	["Block Absorber", "Blk_Var", "T", "Point_Hot"]
3	Comp_after_1_In_T	86.1882...	C	Heat integration	["Block Comp_after_1", "Port_Material_In", "T", "heater"]
4	Comp_after_1_Out_T	86.1882...	C	Heat integration	["Block Comp_after_1", "Port_Material_Out", "T", "heater"]
5	Comp_after_1_Q	0.0	GJ/hr	Heat integration	["Block Comp_after_1", "Blk_Var", "Q", "heater"]
6	Comp_after_2_In_T	131.092...	C	Heat integration	["Block Comp_after_2", "Port_Material_In", "T", "heater"]
7	Comp_after_2_Out_T	40.0	C	Heat integration	["Block Comp_after_2", "Port_Material_Out", "T", "heater"]
8	Comp_after_2_Q	-99.195...	GJ/hr	Heat integration	["Block Comp_after_2", "Blk_Var", "Q", "heater"]
9	Comp_before_1_In_T	115.783...	C	Heat integration	["Block Comp_before_1", "Port_Material_In", "T", "heater"]
10	Comp_before_1_Out_T	40.0	C	Heat integration	["Block Comp_before_1", "Port_Material_Out", "T", "heater"]
11	Comp_before_1_Q	-75.931...	GJ/hr	Heat integration	["Block Comp_before_1", "Blk_Var", "Q", "heater"]
12	Comp_before_2_In_T	115.955...	C	Heat integration	["Block Comp_before_2", "Port_Material_In", "T", "heater"]
13	Comp_before_2_Out_T	40.0	C	Heat integration	["Block Comp_before_2", "Port_Material_Out", "T", "heater"]
14	Comp_before_2_Q	-38.026...	GJ/hr	Heat integration	["Block Comp_before_2", "Blk_Var", "Q", "heater"]
15	Comp_before_3_In_T	102.193...	C	Heat integration	["Block Comp_before_3", "Port_Material_In", "T", "heater"]
16	Comp_before_3_Out_T	40.0	C	Heat integration	["Block Comp_before_3", "Port_Material_Out", "T", "heater"]
17	Comp_before_3_Q	-28.460...	GJ/hr	Heat integration	["Block Comp_before_3", "Blk_Var", "Q", "heater"]
18	Comp_before_4_In_T	97.9320...	C	Heat integration	["Block Comp_before_4", "Port_Material_In", "T", "heater"]
19	Comp_before_4_Out_T	40.0	C	Heat integration	["Block Comp_before_4", "Port_Material_Out", "T", "heater"]

Figure 133: Heat Integration Tags for Compressor Output Variables (1)

Output Variables					
	Name	Value	Unit	Description	Tags
20	Comp_before_4_Q	-26.565...	GJ/hr	Heat integration	["Block Comp_before_4", "Blk_Var", "Q", "heater"]
21	Comp_before_5_In_T	87.0014...	C	Heat integration	["Block Comp_before_5", "Port_Material_In", "T", "heater"]
22	Comp_before_5_Out_T	40.0	C	Heat integration	["Block Comp_before_5", "Port_Material_Out", "T", "heater"]
23	Comp_before_5_Q	-23.507...	GJ/hr	Heat integration	["Block Comp_before_5", "Blk_Var", "Q", "heater"]
24	Comp_before_6_In_T	84.7798...	C	Heat integration	["Block Comp_before_6", "Port_Material_In", "T", "heater"]
25	Comp_before_6_Out_T	40.0	C	Heat integration	["Block Comp_before_6", "Port_Material_Out", "T", "heater"]
26	Comp_before_6_Q	-28.398...	GJ/hr	Heat integration	["Block Comp_before_6", "Blk_Var", "Q", "heater"]
27	Comp_CO2_F	9625.26...	kmol/hr	Output stream	[]
28	Comp_CO2_P	152.787...	bar	Output stream	[]
29	Comp_CO2_T	40.0	C	Output stream	[]
30	Comp_CO2_z_CO2	0.99984...	kmol/...	Output stream	[]
31	Comp_CO2_z_H2O	0.00015...	kmol/...	Output stream	[]
32	Comp_CO2_z_TEGL	2.12657...	kmol/...	Output stream	[]
33	Electricity.Consumption	45.8944...	MW	Total consumpti...	[]
34	Gas_Cooler_2_In_T	121.111...	C	Heat integration	["Block Gas_Cooler_2", "Port_Material_In", "T", "heater"]
35	Gas_Cooler_2_Out_T	39.9999...	C	Heat integration	["Block Gas_Cooler_2", "Port_Material_Out", "T", "heater"]
36	Gas_Cooler_2_Q	-1.0598...	GJ/hr	Heat integration	["Block Gas_Cooler_2", "Blk_Var", "Q", "heater"]
37	HeatEx_1_Cold_In_T	33.4982...	C	Heat integration	["Block HeatEx_1", "Port_Material_In", "T", "HX_Cold"]
38	HeatEx_1_Cold_Out_T	90.3482...	C	Heat integration	["Block HeatEx_1", "Port_Material_Out", "T", "HX_Cold"]

Figure 134: Heat Integration Tags for Compressor Output Variables (2)

Output Variables					
	Name	Value	Unit	Description	Tags
33	Electricity.Consumption	45.8944...	MW	Total consumpti...	[]
34	Gas_Cooler_2_In_T	121.111...	C	Heat integration	["Block Gas_Cooler_2", "Port_Material_In", "T", "heater"]
35	Gas_Cooler_2_Out_T	39.9999...	C	Heat integration	["Block Gas_Cooler_2", "Port_Material_Out", "T", "heater"]
36	Gas_Cooler_2_Q	-1.0598...	GJ/hr	Heat integration	["Block Gas_Cooler_2", "Blk_Var", "Q", "heater"]
37	HeatEx_1_Cold_In_T	33.4982...	C	Heat integration	["Block HeatEx_1", "Port_Material_In", "T", "HX_Cold"]
38	HeatEx_1_Cold_Out_T	90.3482...	C	Heat integration	["Block HeatEx_1", "Port_Material_Out", "T", "HX_Cold"]
39	HeatEx_1_Hot_In_T	121.111...	C	Heat integration	["Block HeatEx_1", "Port_Material_In", "T", "HX_Hot"]
40	HeatEx_1_Hot_Out_T	45.2304...	C	Heat integration	["Block HeatEx_1", "Port_Material_Out", "T", "HX_Hot"]
41	HeatEx_1_Q	14.2141...	GJ/hr	Heat integration	["Block HeatEx_1", "Blk_Var", "Q", "HX_Hot", "HX_Cold"]
42	Mix_MU_In_T	45.2304...	C	Heat integration	["Block Mix_MU", "Port_Material_In", "T", "heater"]
43	Mix_MU_Out_T	37.7777...	C	Heat integration	["Block Mix_MU", "Port_Material_Out", "T", "heater"]
44	Mix_MU_Q	-1.4574...	GJ/hr	Heat integration	["Block Mix_MU", "Blk_Var", "Q", "HX_Hot", "HX_Cold"]
45	MP_Steam.Consumption	6.70749...	GJ/hr	Total consumpti...	[]
46	Regen_Q	6.70749...	GJ/hr	Heat integration	["Block Regen", "Blk_Var", "Q", "Point_Cold"]
47	Regen_T	121.111...	C	Heat integration	["Block Regen", "Blk_Var", "T", "Point_Cold"]
48	status	0.0	Simulation Statu...		[]
49	Trim_Cooler_1_In_T	38.2300...	C	Heat integration	["Block Trim_Cooler_1", "Port_Material_In", "T", "heater"]
50	Trim_Cooler_1_Out_T	39.9999...	C	Heat integration	["Block Trim_Cooler_1", "Port_Material_Out", "T", "heater"]
51	Trim_Cooler_1_Q	0.34551...	GJ/hr	Heat integration	["Block Trim_Cooler_1", "Blk_Var", "Q", "heater"]

Figure 135: Heat Integration Tags for Compressor Output Variables (3)

Run Simulation

Click ▶ on the left side of the panel. Run a single simulation.

Heat Integration Inputs

After the simulation for BFB and Compressor models are complete, a GAMS input file is generated for heat integration. This file is: \gams\GamsInput.inc. The user can verify the correctness of the heat integration inputs in this file. If there is an error, the user can navigate back to the Node Editor to correct the tags.

Simulation Results

The user views simulation results via the Node Editor. In this example, the most important results can be found in the **Output Variables** section in the **Heat Integration Node** and **Steam Cycle Node**. The heat integration results (Figure 136) include consumptions of steam and cooling water, the amount of heat recovered in the steam cycle, the heat exchanger area, the utility cost, the capital cost for the heat exchanger network, and the total annualized cost. The steam cycle calculation results (Figure 137) include net power output and net efficiency with CCS and heat integration, as well as their changes compared to the base case.

Output Variables					
	Name	Value	Unit	Description	Tags
1	Capital.Cost	175.985393479742	\$MM	Approximated capital cost for heat exchanger network	[]
2	Cooling_Water.Consumption	1449.2153	GJ/hr	Cooling water (20 C) consumption (Cost: \$0.21/GJ)	[]
3	FH.Heat.Addition	[138.4294, 177.709, 182.5095, 50.101, 0.0]	GJ/hr	Heat addition to feed water heaters	[]
4	Heat.Exchanger.Area	762315.9147	m^2	Heat exchanger area	[]
5	IP_Steam.Consumption	0.0	GJ/hr	Intermediate-pressure steam (230 C) consumption (Cost: \$8.04/GJ)	[]
6	LP_Steam.Consumption	729.1261	GJ/hr	Low-pressure steam (164 C) consumption (Cost: \$6.25/GJ)	[]
7	Total.Cost	98.08807348585074	\$MM/yr	Approximated total annualized cost for heat exchanger network	[]
8	Utility.Cost	38.89098720000005	\$MM/yr	Utility cost	[]

Figure 136: Heat Integration Results (Heat Integration Node)

Output Variables					
	Name	Value	Unit	Description	Tags
1	Delta.Efficiency.CCS	-11.574469000481773	%	Change of net efficiency due to CCS	[]
2	Delta.Efficiency.HI	1.4103439417636894	%	Change of net efficiency due to heat integration	[]
3	Delta.Power.CCS	-178.96396635956518	MW	Change of net power output due to CCS	[]
4	Delta.Power.HI	21.806680353	MW	Change of net power output due to heat integration	[]
5	Net.Efficiency.CCS	31.895874941281917	%	Net efficiency with CCS	[]
6	Net.Power.CCS	493.17271399343485	MW	Net power output with CCS	[]

Figure 137: Steam Cycle Calculation Results (Steam Cycle Node)

8. SURROGATE MODELS

Surrogate Models Overview

Large-scale computational models are crucial tools to analyze complex systems. When coupled with uncertainty quantification and optimization methods, the resulting computational expense becomes intractable. In order to face the computational burden, surface approximation methods, black box models, or surrogate models are commonly used. FOCUS provides a selection of surrogate modeling tools all using a similar work-flow. This section provides an overview of the surrogate modeling features and capabilities. The details of each tool are provided in the tutorial sections.

The following surrogate modeling tools are currently available:

- ACOSSO – Adaptive COmponent Selection and Shrinkage Operator is a regularization method for simultaneous model fitting and variable selection based in nonparametric regression methods. ACCOSSO is suitable for approximating models with many inputs and no sharp changes.
- ALAMO – Automated Learning of Algebraic Models for Optimization generates algebraic models from data sets. These surrogate models are ideal for equation oriented optimization problems (which are easily differentiable), such as super structure optimization.
- BSS-ANOVA – Bayesian Smoothing Spline Analysis of Variance is a method similar to ACOSSO.
- iREVEAL – Surrogate models for CFD simulations using Kriging or Neural Networks. It contains special features specifically designed for working with CFDs.

Data Selection

The **Data** tab allows the selection of training data to be used to generate a surrogate model (Figure 138). If the session is associated with a flowsheet data (results from a single flowsheet run, optimization runs, or UQ samples), then the flowsheet data is available to be the training data and the table will be populated accordingly.

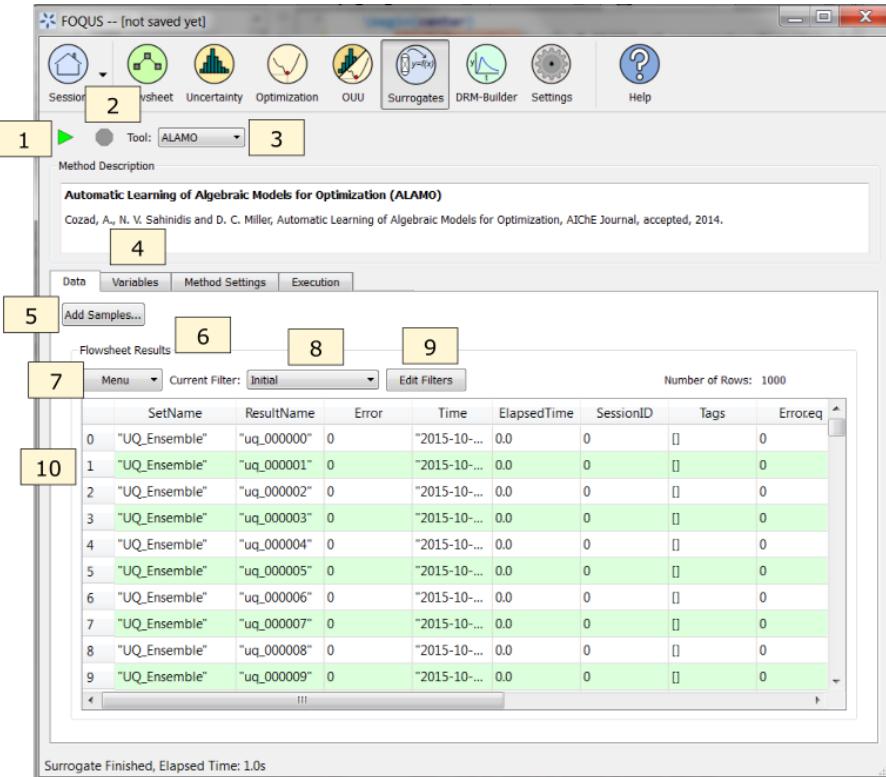


Figure 138: Surrogate Data Form

1. **Run** the surrogate modeling method.
2. **Stop** the surrogate modeling method.
3. **Surrogate modeling tool** enables the user to select the desired surrogate modeling tool from the **Tool** drop-down list.
4. **Description** of the selected surrogate method.
5. **Add Samples** enables the user to generate new training data using a model specified in the flowsheet or an emulator (i.e., a basic response surface provided as part of the UQ module).
6. **Flowsheet Results** are summarized below.
7. The data table has a **Menu** drop-down list that contains display, import/export, and edit commands.
8. Select a data filter from the **Current Filter** drop-down for current data display.
9. Add or edit new data filters from **Edit Filters**. This dialog is shown in Figure 33.
10. The **Display** table displays the results of flowsheet evaluations stored in the FOCUS session file. The columns are:
 - **SetName** is a name assigned to samples. This is typically equivalent to one UQ sample run or one optimization run.
 - **ResultName** is a string representing a result name.

- **Error** is the simulation result status; 0 indicates success, other numbers represent an error. A column for each node displays the error status of each node.
- **Time** displays the time when the result was stored.
- **Elapsed Time** describes how long a result took to calculate.
- **Tags** enables a list of string labels to be applied to results. This could be used to mark results to be used for a particular purpose such as model validation.
- The remaining columns display the input and output variables.

Filters can be used to select data. See Section 2.10 for more information on creating filters to the results. The “All” and “None” filters are available by default. These can be used, for example, to assign all the data as a training set, or to split the data into a separate training set and a test set.

Variables

The **Variables** section is illustrated in Figure 139. This section allows selection of input and output variables used in a surrogate model. Some surrogate methods such as ALAMO may generate and run additional samples while building surrogates. The **Min/Max** columns provide bounds on the variables. Selecting the checkbox next to the variable **Name** indicates that it should be included in the surrogate generation. Failure to select a checkbox for any variables will result in error during surrogate generation.

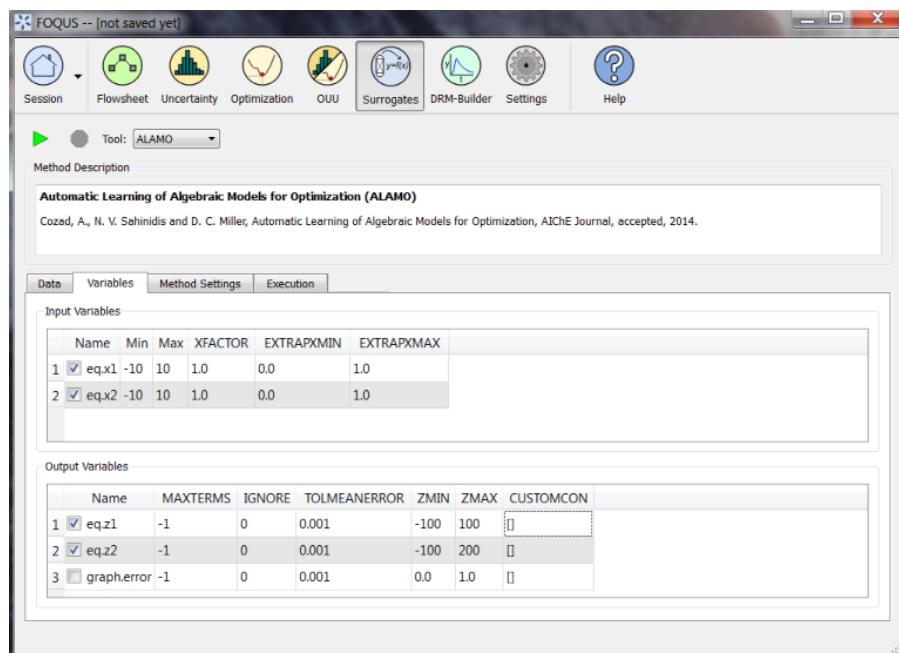


Figure 139: Surrogate Variable Selection

Method Settings

The **Method Settings** table is illustrated in Figure 140. The settings available in this table depend on the surrogate tool. A description of each setting is provided in the third column of the table.

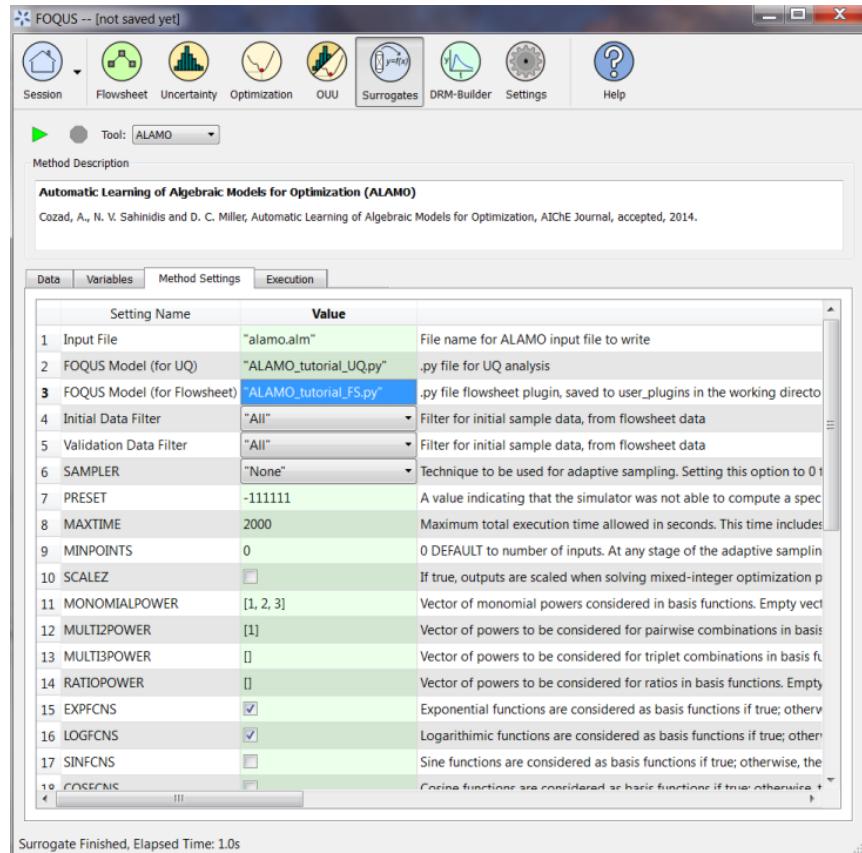


Figure 140: Surrogate Settings

Execution

Clicking **Run** starts the surrogate model building process. The execution monitor displays after **Run** is clicked (see Figure 141). The execution monitor displays the status of the surrogate build. The messages displayed depends on the surrogate tool.

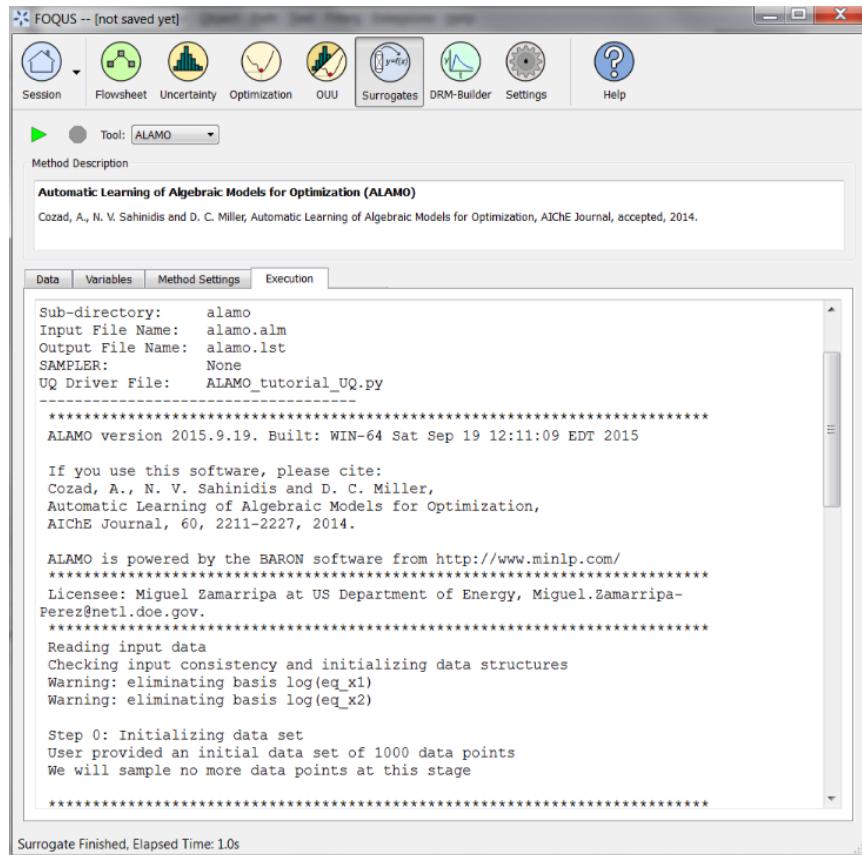


Figure 141: Surrogate Status Monitor

After a successful execution and model building, the results are displayed. Note that in this case, the surrogate modeling tool ends with an error, the errors are displayed in this window. After surrogate generation completes, one or two Python files will be generated depending on the tool. Each tool generates a file that encodes the surrogate model as a general Python script that can be used to evaluate output values for UQ analyses within the UQ module. The other file, if available, is a FOCUS flowsheet plugin model that allows the surrogate to be run in a FOCUS flowsheet. The next version of FOCUS will generate a FOCUS flowsheet plugin model (i.e., the second file) for all surrogate tools.

Tutorial – ALAMO

This tutorial focuses on the use of the ALAMO tool for building algebraic surrogate models. ALAMO builds simplified algebraic models, which are particularly well suited for rigorous equation oriented optimization. To keep the execution of this tutorial fast, a toy problem is used. In this case study the flowsheet calculations and sample generation are done within FOCUS, alternatively, the user can provide a simulation model such as: Excel, Aspen plus, Aspen custom modeler, etc.

Note: Before starting this tutorial the ALAMO product must be downloaded from the products page on the CCSI website. The path for the ALAMO executable file must be set in FOCUS

settings (see Section 2.3).

Flowsheet Setup

1. Open FOCUS.
2. Name the session “Surrogate_Tutorial_1” (Figure 142).

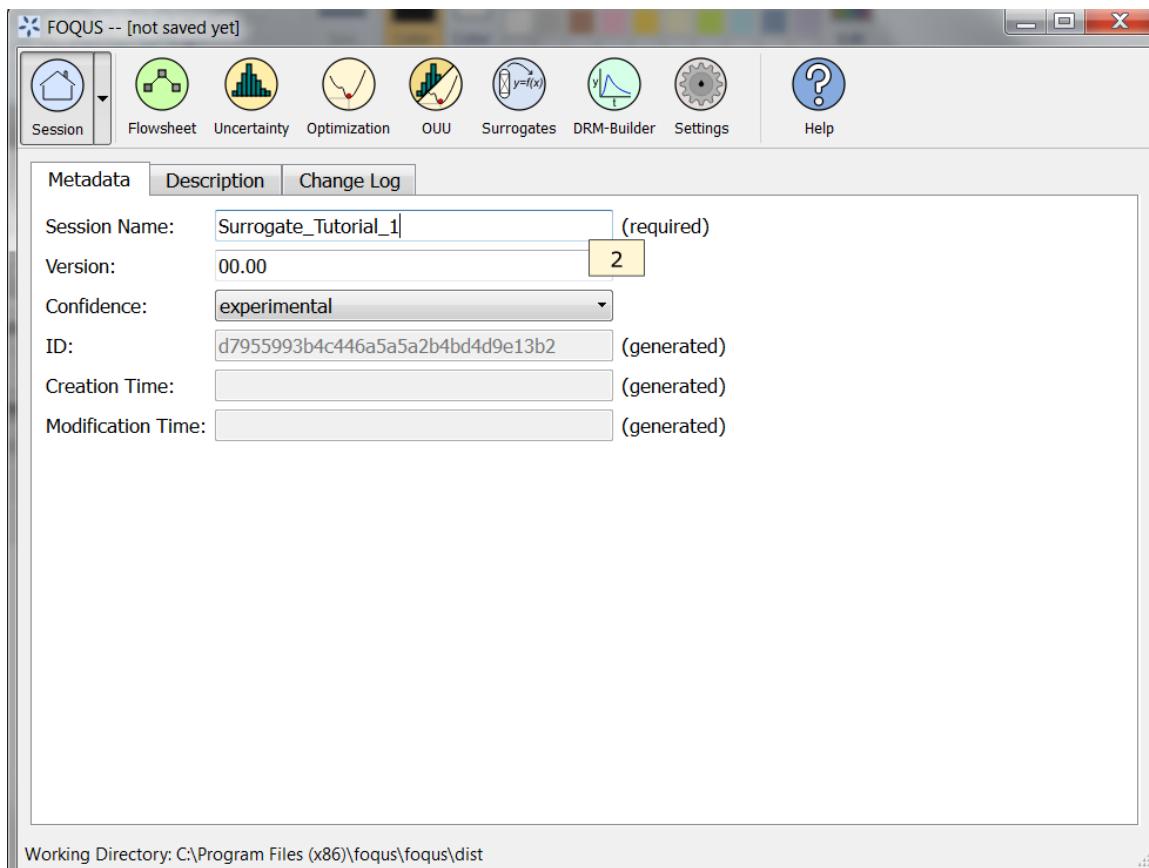


Figure 142: Session Set Up

3. Navigate to the Flowsheet Editor (Figure 143).
4. Add a Flowsheet Node named “eq.”
5. Display the Node Editor by clicking the **Node Editor** toggle button.

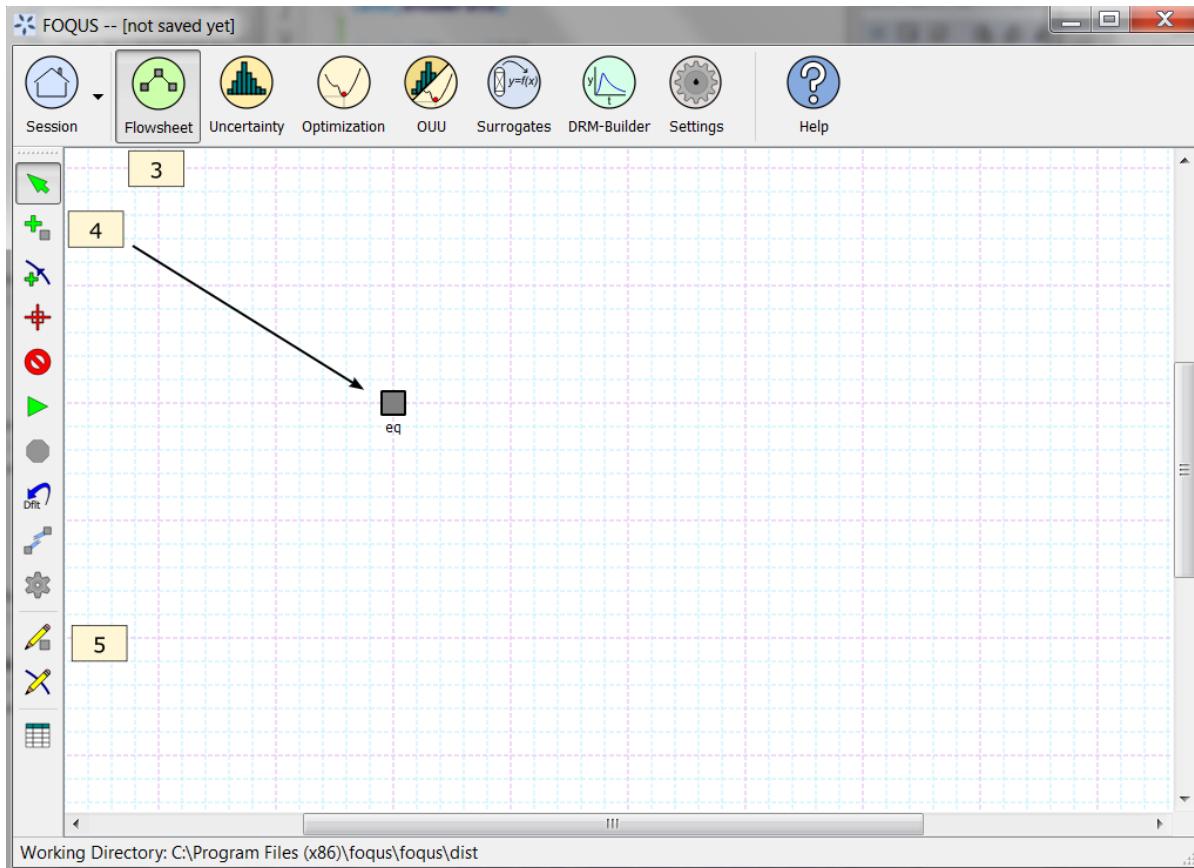
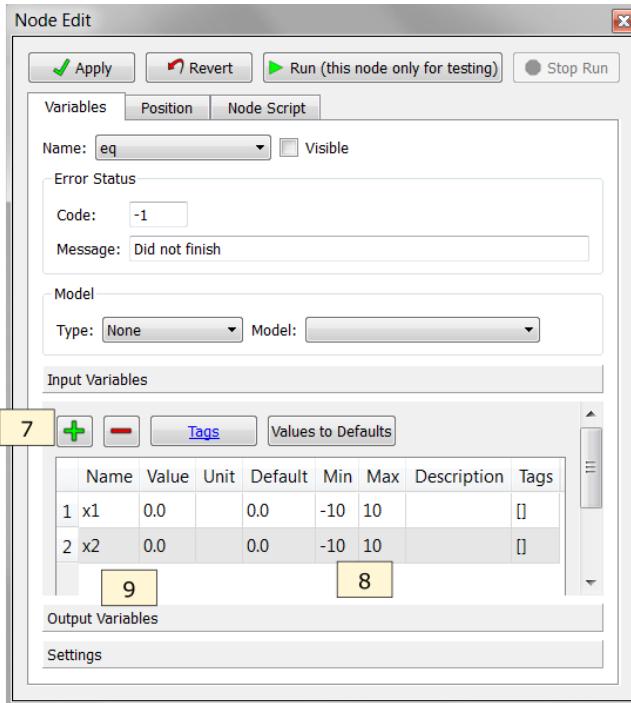


Figure 143: Flowsheet Setup

The **Node Editor** displays (Figure 144). The first step to setting up the node for this problem is to add input and output variables to the node.

6. If the input variables table is not displayed as shown in Figure 144, click the **Variables** tab and then click the **Input Variables** toolbox section.
7. Add the variables “x1” and “x2” by clicking the **Add** icon (+) above the input table.
8. Edit the **Min/Max** value for both variables to be “-10.0” and “10.0.”
9. Add two output variables “z1” and “z2.”

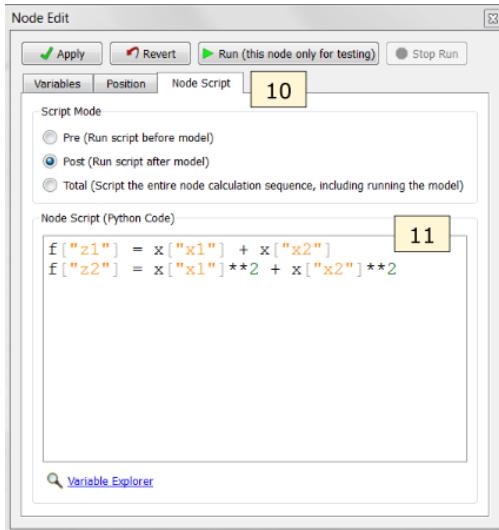
**Figure 144: Node Variables**

To keep the execution time short, the node will not be assigned to a simulation model and calculations are performed directly in FOCUS.

10. Click on the **Node Script** tab in the Node Editor to enter the test equation (this step replaces the use of a simulator).
11. Enter the following equations (Figure 145):

```
f["z1"] = x["x1"] + x["x2"]
f["z2"] = x["x1"]**2 + x["x2"]**2
```

The node script calculations are written in Python. The dictionary “f” stores output values while the dictionary “x” stores input values.

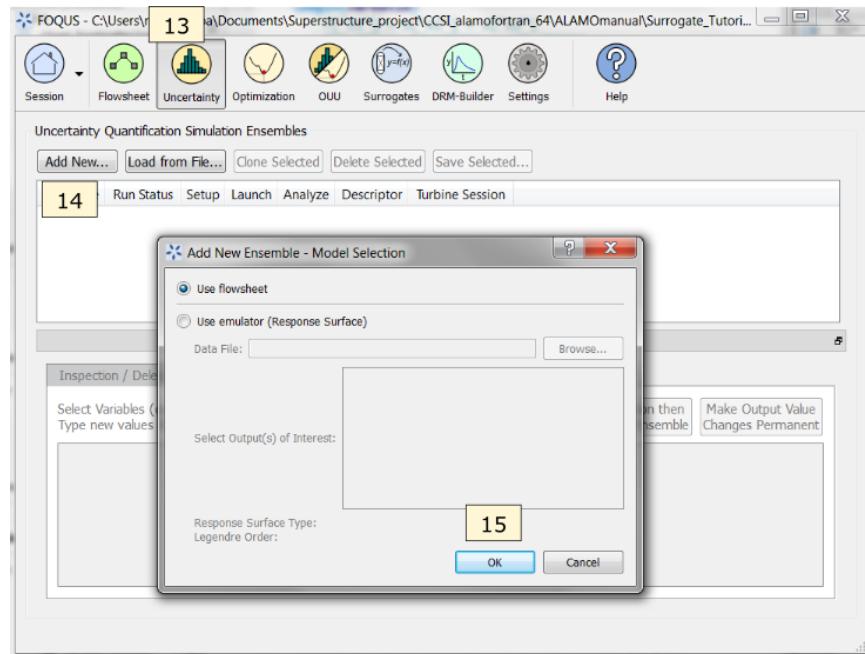
**Figure 145: Node Script**

12. Test the model by running the flowsheet with the value “2” for “x1” and “x2.” After running, the output variables should have the values “4.0” for “z1” and “8.0” for “z2.”

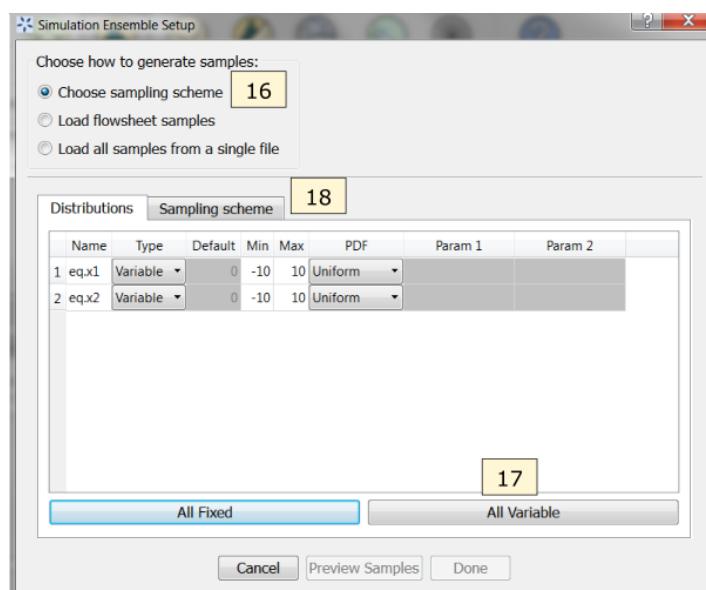
Creating Initial Samples

There are two ways to start an ALAMO run: (1) generate a set of initial data, (2) use ALAMO’s adaptive sampling with no initial data and let ALAMO generates its own samples. Adaptive sampling can be used with initial data to generate more points if needed. In this case, initial data is provided and adaptive sampling is used.

13. Select the UQ tool by clicking on the **Uncertainty** button on the Home window (Figure 146).
14. Click the **Add New** button.
15. The **Add New Ensemble - Model Selection** dialog will appear. Click **OK** to set up the sampling scheme.

**Figure 146: Add a New Sample Ensemble**

16. The sample ensemble setup dialog displays (Figure 147). Select [Choose sampling scheme](#).
17. Click the [All Variable](#) button.
18. Select the [Sampling scheme](#) tab.

**Figure 147: Sample Distributions**

19. The **Sampling scheme** dialog should display (Figure 148). Select “Latin Hypercube” from the list.
20. Set the **# of samples** to “10.”
21. Click **Generate Samples**.
22. Click **Done**.

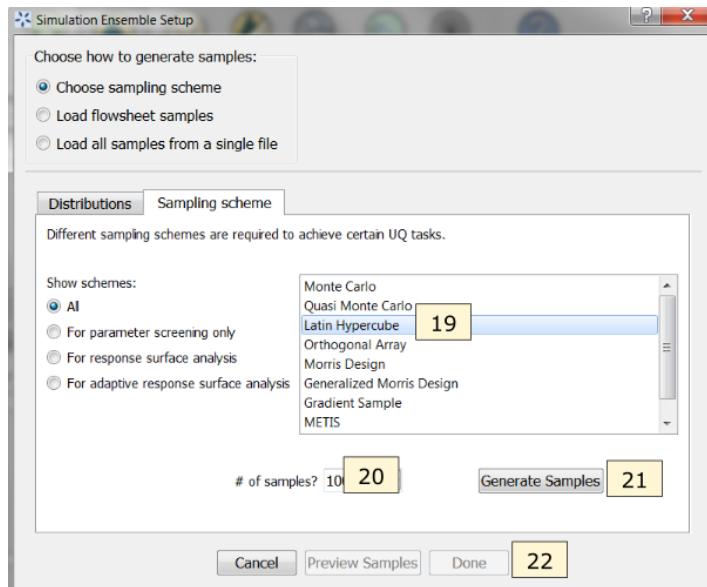


Figure 148: Sample Methods

23. Once the samples have been generated a new sample ensemble displays in the UQ tool window (Figure 149). Click **Launch** to run and generate the samples.

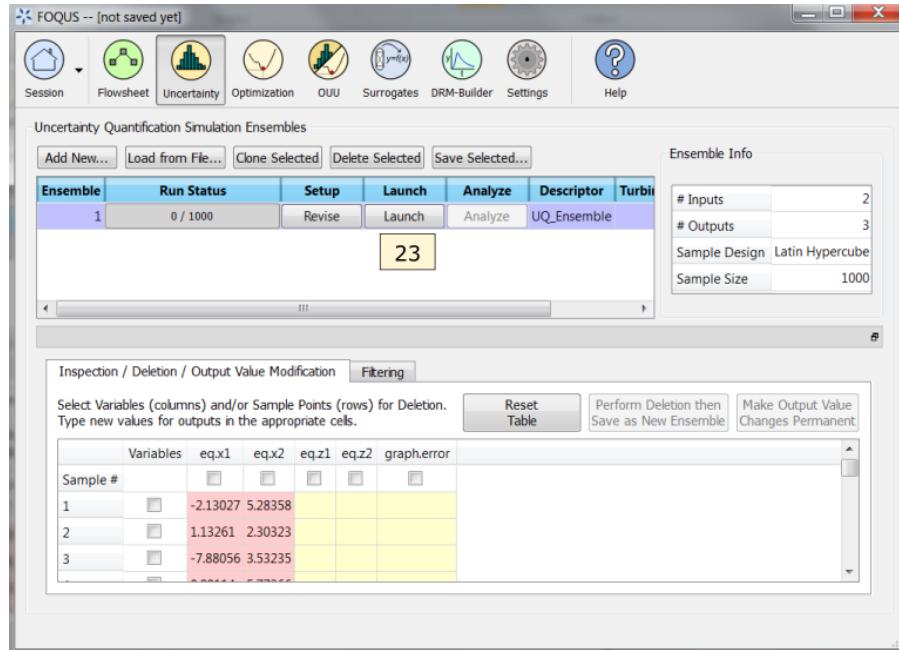
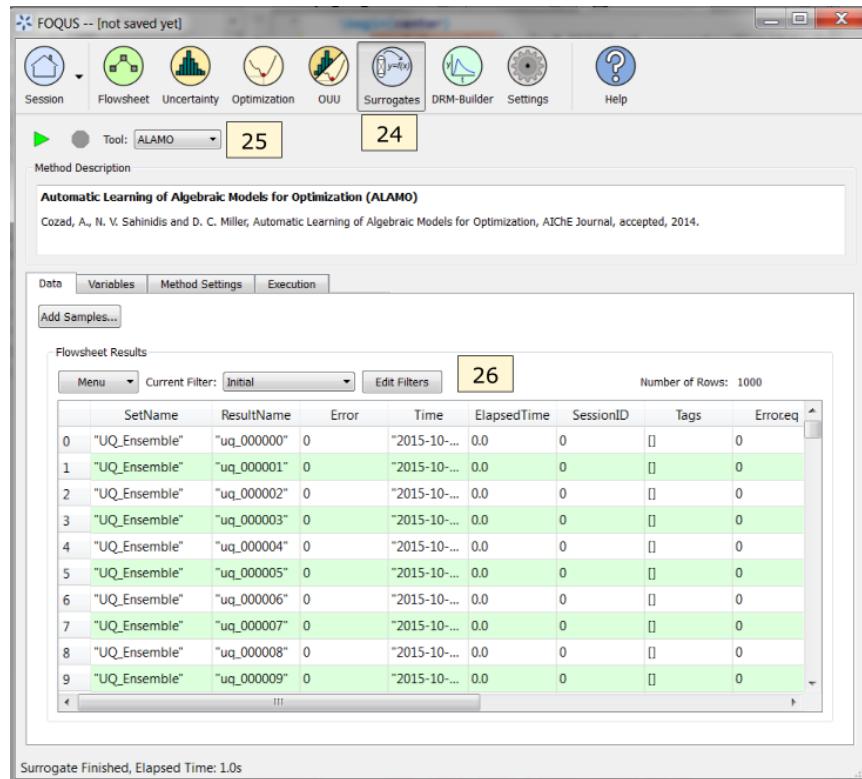


Figure 149: Run Samples

Data Selection

Initial and validation data can be specified by creating filters that specify subsets of flowsheet data. In this tutorial only initial data will be used. A filter must be created to separate the results of the single test run from the UQ samples.

24. Click on the **Surrogates** button from the Home window. The surrogate tool displays 150.
25. Select “ALAMO” from the **Tool** drop-down list.
26. Click **Edit Filters** in the **Flowsheet Results** section to create a filter.

**Figure 150: Surrogate Data**

27. Figure 151 displays the Data Filter Editor.
28. Add the filter for initial data.
 - (a) Click **New Filter**, and enter “Initial” as the filter name.
 - (b) Click **Add Rule**.
 - (c) In the “Term 1” column enter: set (no quotes).
 - (d) In the “Term 2” column enter: “UQ_Ensemble” (with quotes).
 - (e) In the “Operator” column select “=.”
29. Click **Done**.

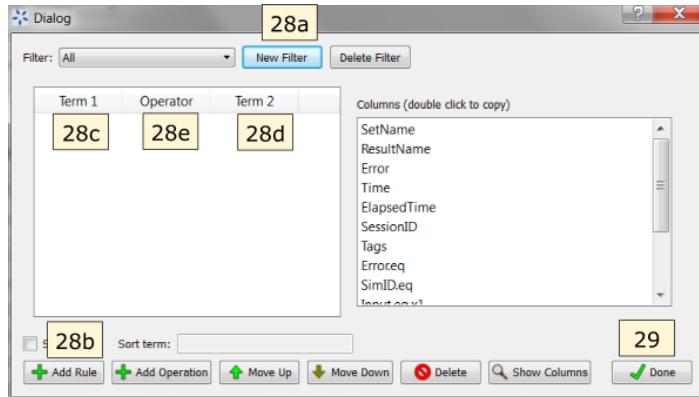


Figure 151: Data Filter Dialog

Variable Selection

In this section, input and output variables need to be selected. Generally, any input variables that vary in the data set should be selected. However, in some cases, variables may be found to have no, or very little, effect on the outputs. Only the output variables of interest need to be selected. Note: Each output is independent from each other and for the model building, selecting one output is the same as selecting more.

30. Select the Variables tab (Figure 152).
31. Select the checkbox for both input variables.
32. Select the checkbox for both output variables.

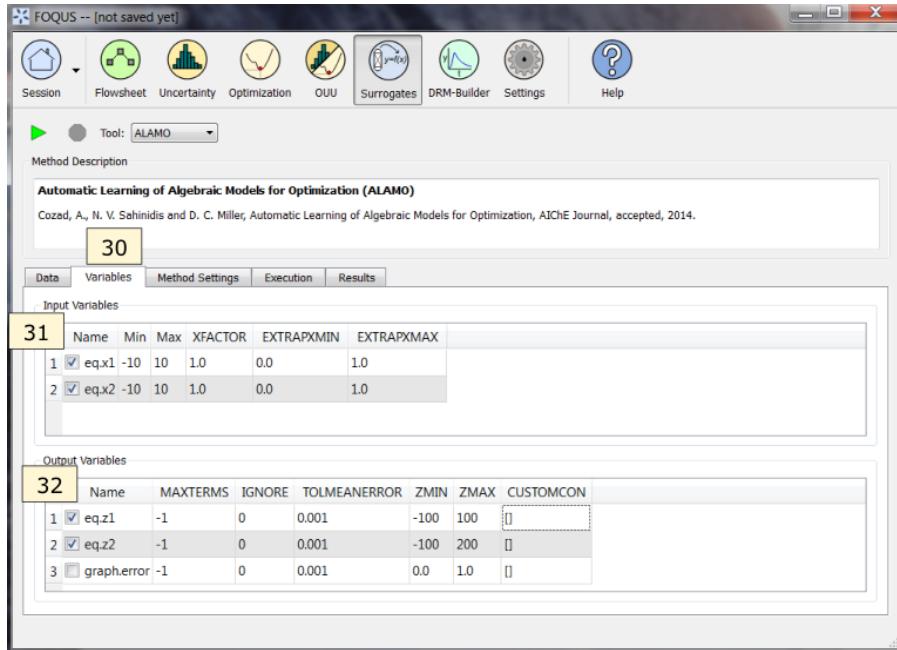


Figure 152: Variable Selection

Method Settings

The most important feature to generate "good" algebraic models is to configure the settings accordingly to the problem to be solved. Each setting has a good description in FOCUS. The JSON parser is used to read method settings values. Strings must be contained in quotes. Lists have the following format: [element 1, element 2].

33. Click on the **Method Settings** tab (see Figure 153).
34. Set the **FOCUS Model (for UQ)** to "ALAMO_tutorial_UQ.py."
35. Set the **FOCUS Model (for Flowsheet)** to "ALAMO_tutorial_FS.py"
36. Set **Initial Data Filter** to "Initial."
37. Set **SAMPLER** to select the adaptive sampling method: "None" "Random" or "SNOBFIT." Use "None" in this tutorial.
38. Set **MONOMIALPOWER** to select the single variable term powers to [1,2,3].
39. Set **MULTI2POWER** to select the two variable term powers to [1].
40. Select functions to be considered as basis functions (**EXPFCNS**, **LOGFCNS**, **SINFCNS**, **COSFCNS**).
41. Leave the rest of settings as default (see Table 8).
42. Save this FOCUS session for use in the ACOSSO and BSS-ANOVA tutorials.

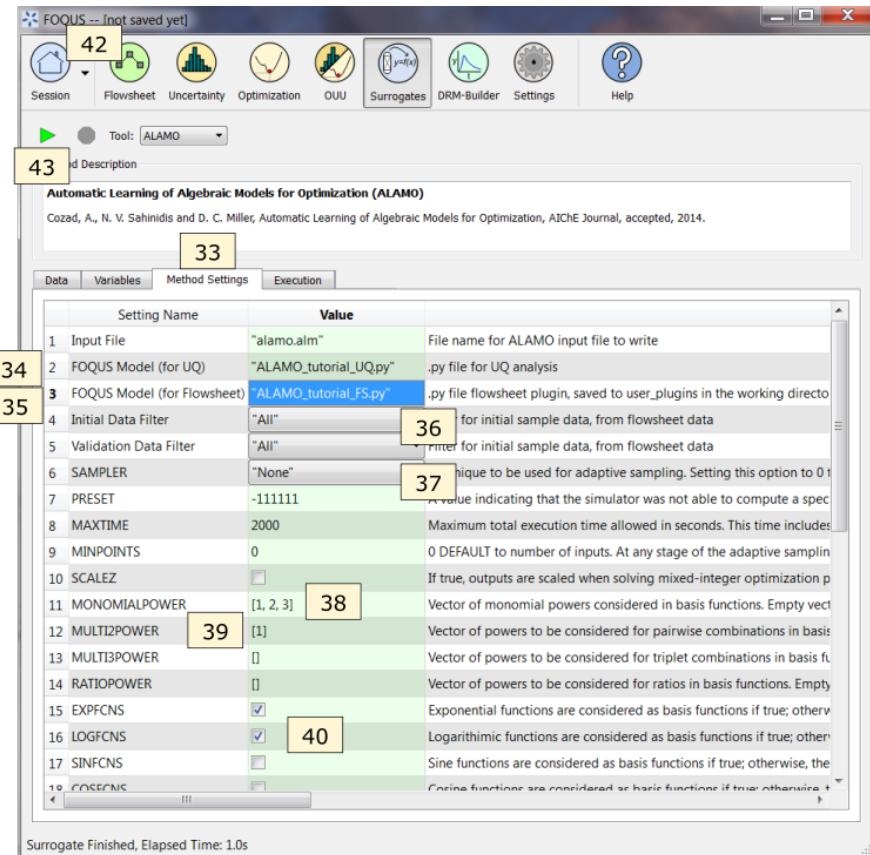


Figure 153: ALAMO Method Settings

Execution

43. Click the Run icon at the top of the window.
44. The ALAMO Execution tab starts displaying execution file path, sub-directories, input files, and output files.
 - (a) ALAMO version.
 - (b) License Information.
 - (c) Step 0 displays the data set to be used by ALAMO.
 - (d) Step 1 displays the modeler used by ALAMO to generate the algebraic model.
 - (e) Once the surrogate model has finished, the equations are displayed in the execution window. It may be necessary to scroll up a little. The result is shown in Figure 154.
 - (f) Finally, the statistics display the quality metrics of the models generated.

The screenshot shows the ALAMO software interface during execution. It displays the following information:

- Model building for variable eq_z1**
- $eq_z1 = 0.98 * eq_x1$
- $eq_z1 = 1.0 * eq_x1 + 1.0 * eq_x2$
- Model building for variable eq_z2**
- $eq_z2 = 1.5 * eq_x1^2 * 2.0$
- $eq_z2 = 1.0 * eq_x1^2 * 2.0 + 1.0 * eq_x2^2 * 2.0$
- Calculating quality metrics on observed and va**
- Quality metrics for output eq_z1**
- RMSE: 0.150E-10
- RMSE for validation dataset: 0.150E-10
- R2: 1.00
- R2 for validation dataset: 1.00
- Model size: 2
- BIC: 13.8
- CP: -996.
- AICc: -0.4988E+05
- HQC: -0.4988E+05
- MSE: 0.224E-21
- Convex penalty: 20.0
- Quality metrics for output eq_z2**
- RMSE: 0.147E-09
- RMSE for validation dataset: 0.147E-09
- R2: 1.00
- R2 for validation dataset: 1.00
- Model size: 1
- BIC: 13.8
- CP: -996.
- AICc: -0.453E+05
- HQC: -0.453E+05
- MSE: 0.216E-19
- Convex penalty: 20.0
- Total execution time: 0.94E-01 s
- Times breakdown

 - OLR time: 0.16E-01 s in 71 ordinary linear regressions
 - MIP time: 0.0 s in 0 quadratic integer programming problems
 - Simulation time: 0.0 s to simulate 0 points
 - All other time: 0.78E-01 s in 1 iteration

- Normal termination

Figure 154: ALAMO Execution

Results

The results are exported as a PSUADE driver file that can be used to perform UQ analysis of the models, and a FOQUUS Python plugin model that allows it to be used in a FOQUUS flowsheet. The equations can also be viewed in the results section.

See tutorial Section 8.6 and 8.7 for information about analyzing the model with the UQ tools or running the model on the flowsheet.

As mentioned in section 8.2.5 the method settings are very important. A brief description and hints are included in Table 8.

Table 8: ALAMO Method Settings

Method Settings	Description
Initial Data Filter	Filter to be applied to the initial data set. Data filters help the user to generate models based on specific data for each variable.
Validation Data filter	Data set used to compute model errors at the validation phase. The number of data points in a preexisting validation data set can be specified by the user.
SAMPLER	Adaptive sampling method to be used. Options: "None", "Random" and "SNOBFIT". Adaptive sampling method to be used by ALAMO when more sampling points are needed by the model. If Random is used a simulator must be provided by the user. If SNOBFIT is used a simulator must be provided by the user and MATLAB must be installed.

MAXTIME	Maximum execution time in seconds. This time includes all the steps on the algorithm, if simulations are needed they run in this time.
MINPOINTS	Convergence is assessed only if the simulator is able to compute the output variables for at least MINPOINTS of the data set. A reduced number of MINPOINTS may reduce the computational time to get a model, but also reduces the accuracy of the model. MINPOINTS must be a positive integer.
PRESET	Value to be used if the simulator fails. This value must be carefully chosen to be an otherwise not realizable value for the output variables.
MONOMIALPOWERS	Vector of monomial powers to be considered as basis functions, use empty vector for none []. Exponential terms allowed in the algebraic model. i.e., if selecting [1,2] the model considers x_1 and x_1^{**2} as basis functions.
MULTI2POWER	Vector of pairwise combination of powers to be considered as basis functions. Pairwise combination of powers allowed in the algebraic model. i.e., [1,2] allows terms like x_1*x_2 in the algebraic model.
MULTI3POWER	Vector of three variables combinations of powers to be considered as basis functions.
EXPFCNS, LOGFCNS, SINFCNS, COSFCNS functions	Use or not of exp, log, sin, and cos functions as basis functions in the model.
RATIOPOWER	Vector of ratio combinations of powers to be considered in the basis functions. Ratio combinations of powers are [empty as default].
Radial Basis Functions	Radial basis functions centered around the data set provided by the user. These functions are Gaussian and are deactivated if their textual representation requires more than 128 characters (in the case of too many input variables and/or datapoints).
RBF parameter	Constant penalty used in the Gaussian radial basis functions.
Modeler	Fitness metric to be used for model building. Options: BIC (Bayesian Information Criterion), Mallow's Cp, AICc (Corrected Akaike's Information Criterion), HQC (Hannan-Quinn Information Criterion), MSE (Mean Square Error), and Convex Penalty.
ConvPen	Convex penalty term. Used if Convex Penalty is selected.
Regularizer	Regularization method is used to reduce the number of potential basis functions before the optimization.
Tolrelmetric	Convergence tolerance for the chosen fitness metric is needed to terminate the algorithm.

ScaleZ	If used, the variables are scaled prior to the optimization problem is solved. The problem is solved using a mathematical programming solver. Usually, scaling the variables may help the optimization procedure.
GAMS	GAMS is the software used to solve the optimization problems. The executable path is expected or the user must declare GAMS.exe in the environment path.
GAMS Solver	Solver to be used by GAMS to solve the optimization problems. Mixed integer quadratic programming solver is expected like BARON (other solvers can be used).
MIPOPTCR	Relative convergence tolerance for the optimization problems solved in GAMS. The optimization problem is solved when the optcr is reached. 5 to 1 % is expected (0.005 to 0.001).
MIPOPTCA	Absolute convergence tolerance for mixed-integer optimization problems. This must be a nonnegative scalar.
Linear error	If true, a linear objective function is used when solving the mixed integer optimization problems; otherwise, a quadratic objective function is used.
Constraint Regression (CONREG)	Specify whether constraint regression is used or not, if true bounds on output variables are enforced.
CRNCUSTOM	If true, Custom constraints are entered in the Variable tab.
CRNINITIAL	Number of random bounding points at which constraints are sampled initially (must be a nonnegative integer).
CRNMAXITER	Maximum allowed constrained regressions iterations. Constraints are enforced on additional points during each iteration (must be positive integer).
CRNVIOL	Number of bounding points added per round per bound in each iteration (must be positive integer).
CRNTRIALS	Number of random trial bounding points per round of constrained regression (must be a positive integer).
CUSTOMBAS	A list of user-supplied custom basis functions can be provided by the user. The parser is not case sensitive and allows for any Fortran functional expression in terms of the XLABELS (symbol ^ may be used to denote power).

Tutorial – ACOSSO

This tutorial covers the ACOSSO surrogate modeling method. The Adaptive COnponent Selection and Shrinkage Operator (ACOSSO) surface approximation was developed under the Smoothing Spline Analysis of Variance (SS-ANOVA) modeling framework [3]. As it is a smoothing type method, ACOSSO works best when the underlying function is somewhat smooth. For functions which are known to have sharp changes or peaks, etc., other methods may be more appropriate. Since it implicitly performs variable selection, ACOSSO can also work well when there are a large number of input variables. The ACOSSO procedure also allows for categorical inputs [4].

This tutorial uses the same flowsheet and sample setup as the ALAMO tutorial in Section 8.2. The statistics software “R” is also required to use ACOSSO and BSS-ANOVA. Before starting this tutorial, you will need to install R version 3.1 or later (see <https://cran.r-project.org/>).

Once R is installed, you will need to install the “quadprog” package. ACOSSO requires this package for solving quadratic programming problems. You will only need to perform this step once.

1. Start R. In Windows, this must be done with administrative privileges. Either run this from an administrator account, or right-click “R x64 3.1.2” and click “Run with administrator” and type in administrator credentials.
2. Inside the R console, type:
 - `install.packages('quadprog')`
 - `library(quadprog)`
 - `q()`

The first line installs the package. If prompted for a CRAN mirror, select the one closest to you geographically. The second line loads the package. The last line quits R. If prompted to save workspace image, choose ‘y’.

Once you have done these steps, ACOSSO is ready to be invoked inside FOQUS.

1. Set the path to the RScript executable.
 - (a) Click the **Settings** button in the Home window.
 - (b) Change the RScript path if necessary. The **Browse** button opens a file browser that can be used to set the path.
2. Complete the ALAMO tutorial in Section 8.2 through Step 32, or load the FOQUS session saved after completing the ALAMO tutorial.
3. Click the **Surrogates** button in the Home window (Figure 155).
4. Select “ACOSSO” in the **Tool** drop-down list.
5. Select the **Method Settings** tab.

6. Set “Data Filter” to “Initial.”
7. Set “Use Flowsheet Data” to “Yes.”
8. Set “FOQUS Model (for UQ)” to “ACOSSO_Tutorial_UQ.py.”
9. Set “FOQUS Model (for Flowsheet)” to “ACOSSO_Tutorial_FS.py.”
10. Click the **Run** icon (Figure 155).

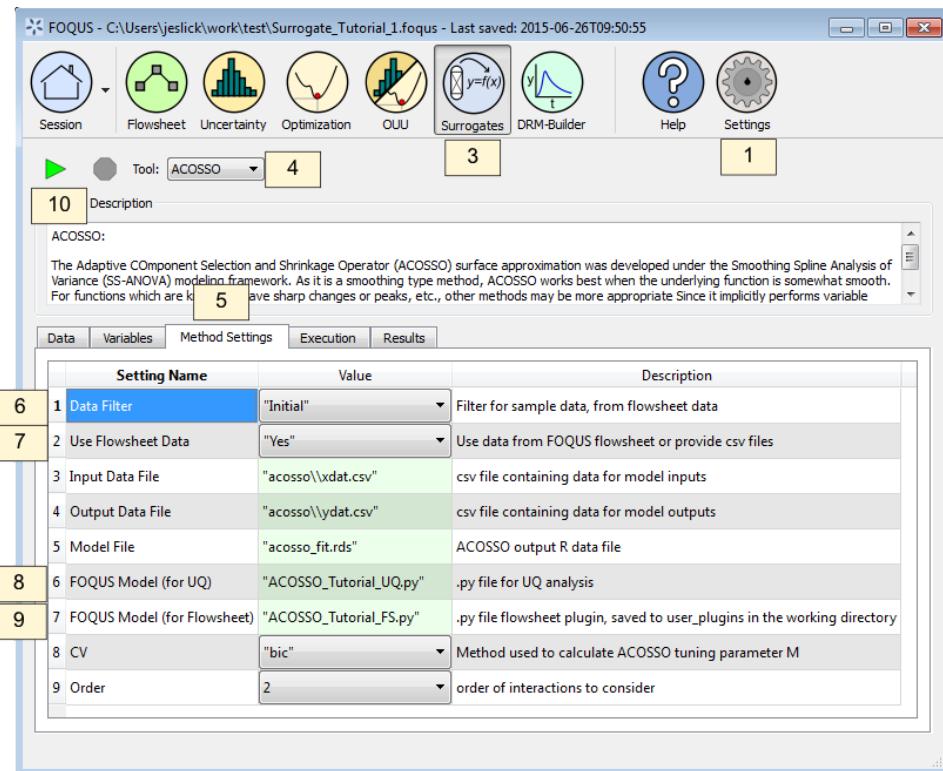


Figure 155: ACOSSO Session Set Up

11. The execution window will automatically display. While ACOSSO is running, the execution window may show warnings, but this is normal.
12. When the run completes, a UQ driver file is created, allowing the ACOSSO surrogate to be used as a user-defined response surface in UQ analyses. (See Section 8.6.)
13. ACOSSO also produces a flowsheet plugin; however.

Tutorial – BSS-ANOVA

This tutorial covers the BSS-ANOVA surrogate modeling method. The Bayesian Smoothing Spline ANOVA (BSS-ANOVA) is essentially a Bayesian version of ACOSSO [5]. It is Gaussian Process (GP) model with a non-conventional covariance function that borrows its form from SS-ANOVA.

It tackles the high dimensionality (of inputs) on two fronts: (1) variable selection to eliminate uninformative variables from the model and (2) restricting the level of interactions involved among the variables in the model. This is done through a fully Bayesian approach which can also allow for categorical input variables with relative ease. Since it is closely related to ACOSSO, it generally works well in similar settings as ACOSSO. The BSS-ANOVA procedure also allows for categorical inputs [4]. In this current implementation, BSS-ANOVA is more computationally intensive than ACOSSO, so ACOSSO is preferred for faster surrogate generation.

This tutorial uses the same flowsheet and sample setup as the ALAMO tutorial in Section 8.2. The statistics software “R” is also required to use ACOSSO and BSS-ANOVA. Before starting this tutorial, you will need to install R version 3.1 or later (see <http://cran.r-project.org/>).

1. Set the path to the RScript executable.
 - (a) Click the **Settings** button from the Home window.
 - (b) Change the RScript path if necessary. The **Browse** button opens a file browser that can be used to set the path.
2. Complete the ALAMO tutorial in Section 8.2 through Step 32, or load the FOCUS session saved after completing the ALAMO tutorial.
3. Click the **Surrogates** button from the Home window (Figure 156).
4. Select “BSS-ANOVA” in the **Tool** drop-down list.
5. Select the **Method Settings** tab.
6. Set “Data Filter” to “Initial.”
7. Set “Use Flowsheet Data” to “Yes.”
8. Set “FOCUS Model (for UQ)” to “bssanova_tutorial_uq.py.”
9. Set “FOCUS Model (for Flowsheet)” to “bssanova_tutorial_fs.py.”
10. Click the **Run** icon (Figure 156).

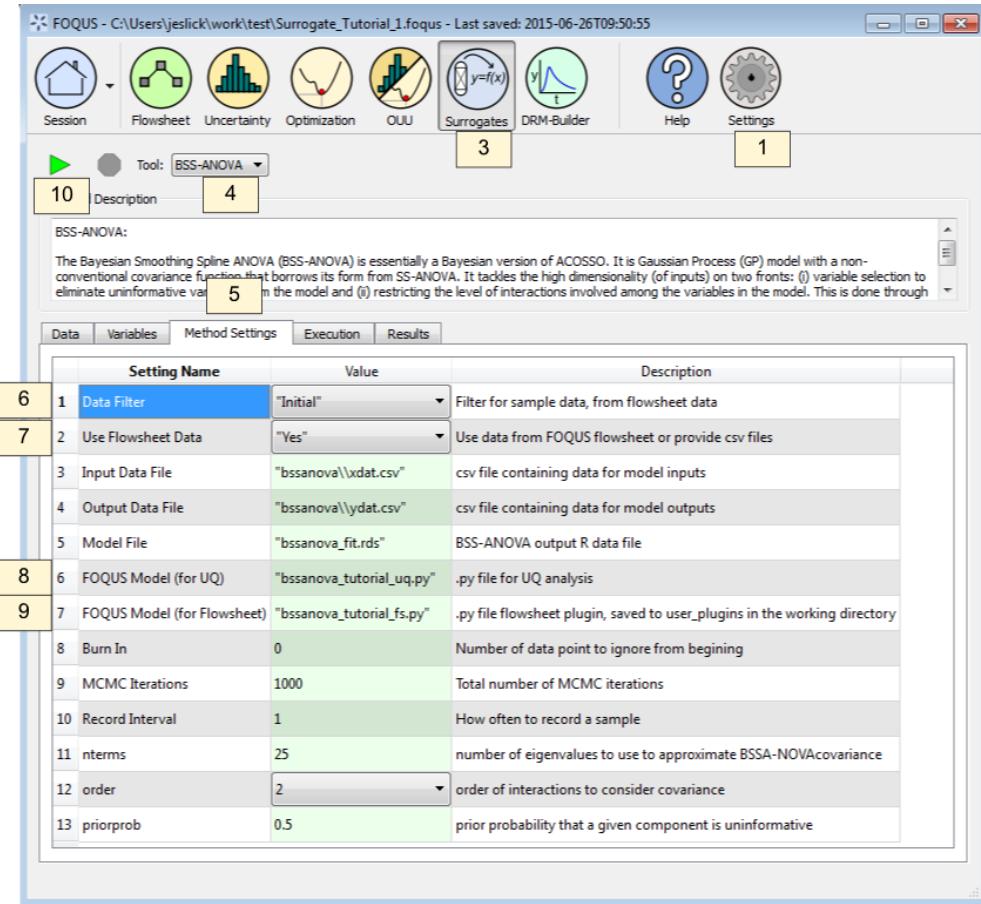


Figure 156: BSS-ANOVA Session Set Up

11. The execution window will automatically display. While BSS-ANOVA is running, the execution window may show warnings, but this is normal.
12. When the run completes, a UQ driver file is created, allowing the BSS-ANOVA surrogate to be used as a user-defined response surface in UQ analyses. (See Section 8.6.)
13. BSS-ANOVA also produces a flowsheet plugin.

Tutorial – iREVEAL

The iREVEAL documentation is currently separate, but will be merged in the next release. The iREVEAL user's manual is included in the FOCUS installation. There is a link to the manual in the Windows start menu under CCSI Tools\FOCUS.

Tutorial – Surrogates with UQ Tools

For the purpose of this tutorial, we will use ACOSSO to demonstrate the use of a surrogate within the UQ module. The steps are the same regardless of the surrogate tool chosen.

To perform the UQ analysis, Python is required for use the “User Regression” response surface that will be used. Before starting this tutorial, you will need to install Python 2.7.x (not Python 3). (See <https://www.python.org/downloads/>). In addition, if *.py files have been re-associated with other executables (e.g. editors), please change the association back to python.exe.

1. Load a fresh session by clicking the **Session** button from the Home window. Select **Open Session** and then navigate to the “examples/UQ” directory. Select “Rosenbrock_no_vectors.foqus.” This will load a session with a simple flowsheet containing a single node.
2. Click **Settings** and ensure that (1) FOQUS Flowsheet Run Method is set to “Local”, and that (2) proper paths are set for PSUADE and RScript.
3. Train an ACOSSO surrogate of this node by clicking the **Surrogates** button from the Home window.
 - (a) Click **Add Samples** and select “Use Flowsheet”. This will display the **Simulation Ensemble Setup** dialog.
 - (b) Within this dialog, ensure all variables are set to “Variable” type in the **Distributions** tab. In the **Sampling scheme** tab, select “Monte Carlo” as your sampling scheme, set the number of samples to 100, and then click **Generate Samples** to generate the set of input values. Click **Done** to return to the Surrogates screen.
 - (c) Once sample generation completes, click the **Uncertainty** button from the Home window.
 - (d) Click the **Launch** button to generate the samples.
 - (e) Click the **Surrogates** button from the Home window. The **Data** tab of the Surrogates screen should now displays a **Flowsheet Results** table that is populated with the values of the new input samples.
 - (f) From the **Variables** tab, select all of the checkboxes. (There should be six checkboxes for input variables and one checkbox for output variable.) Here, you are defining the inputs and outputs for your surrogate function.
 - (g) From the **Method Settings** tab, note the name of the file next to “FOQUS Model (for UQ)”. This will be the name of the UQ driver file that contains the Python code that implements the surrogate function.
 - (h) On top of this screen, select “ACOSSO” as your surrogate tool from the **Tool** drop-down list and then click on the green arrow to start training the surrogate.
 - (i) Once complete, a popup window will display, reminding you of the location of the drive file. Note the location as you will need this information later inside the UQ module.
4. Perform a response-surface-based uncertainty analysis by clicking the **Uncertainty** button from the Home window.
 - (a) In the **Uncertainty Quantification Simulation Ensembles** table. A row corresponding to the ensemble that was just generated for surrogate training should be displayed. This same ensemble can be used or a new one can be created to be used as the test data set for analysis. In the row corresponding to the ensemble to be analyzed, click the **Analyze** button to proceed. This action will bring up an analysis dialog.

- (b) Within this analysis dialog, navigate to “Analysis” section. For Step 1, select “Response Surface”. For Step 3, select “User Regression” in the first drop-down list. Lastly, for “User Regression File”, browse to the same location as the UQ driver file that was generated within the Surrogates module. (This is the same location that was previously noted from the popup message.) At this point, your surrogate function is now set up as a user-defined response surface and all response-surface-based UQ analyses are accessible.
- (c) Click **Validate** (Step 4) to perform response surface validation. Once complete, a figure with cross-validation results will be displayed: a histogram of errors to the left and a plot of predicted values versus actual values to the right. For more information, refer to the UQ Tutorial in Section 4.2.4.
- (d) Once a “Response Surface” has been validated, other UQ analysis options are available. Choose “Uncertainty Analysis” in Step 5 and click **Analyze** to perform uncertainty analysis using your ACOSO surrogates.

During validation, if the error, “RSAnalyzer: RSTest.hs.m does not exist.” displays, this is likely caused by incompatibility with the surrogate and the test data. An example scenario might be your test data has six inputs, but your surrogate assumes five inputs. This is easily fixed by returning to the Surrogates screen, clicking on the **Variables** tab, and making sure the appropriate selections are made (i.e., check off six inputs instead of just five).

Tutorial – Surrogates with the Flowsheet

This section provides a brief tutorial for using the flowsheet plugin models generated by surrogate modeling methods. In the next FOQUS release all surrogate modeling methods will produce a model that can be run in a FOQUS flowsheet. **Currently iREVEAL does not produce a flowsheet model.**

Before doing this tutorial complete the ALAMO tutorial in Section 8.2.

1. Open FOQUS. If FOQUS has not been closed since completing the ALAMO tutorial, close it and reopen it. There is a known issue where existing flowsheet model plugins may not update until FOQUS is restarted.
2. Enter “FS_Plugin_Tutorial” as the Session Name.
3. Click the **Flowsheet** button from the Home window.
4. Click the **Add Node** icon in the left toolbar (see Figure 157).
5. Click a location for the node in the Flowsheet area.
6. Enter “model” for the node name (without quotes).
7. Click the **Node Editor** icon in the left toolbar (see Figure 157).
8. In the **Node Editor**, select “Plugin” from the Model **Type** drop-down list.
9. Select “ALAMO_Tutorial_FS” from the **Model** drop-down list.

10. Set the Value of the Input Variables “eq.x1” to 2.
11. Set the Value of the Input Variables “eq.x2” to 3.
12. Click the Run icon in the left toolbar (see Figure 157).
13. Wait for the Flowsheet evaluation to complete. It should finish successfully.
14. Check the value of the Output Variables; the approximate values should be $z_1 = 5$ and $z_2 = 13$.

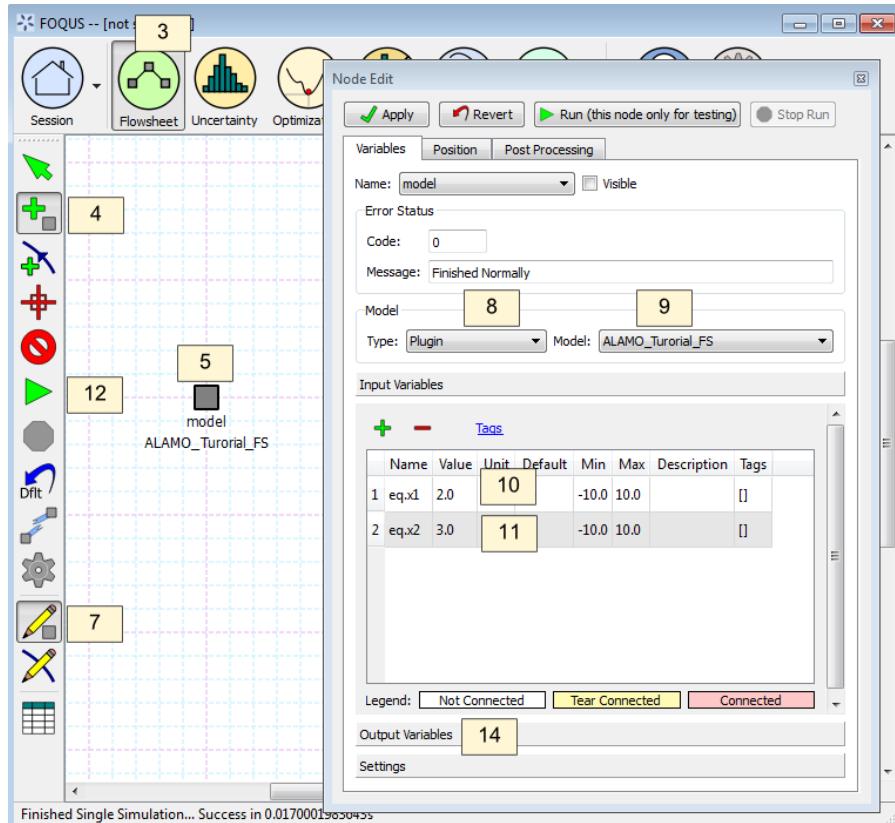


Figure 157: Plugin Flowsheet

9. SIMSINTER

SimSinter Configuration

SimSinter is the standard interface library that FOCUS and Turbine use to drive the target simulation software. SimSinter currently supports AspenPlus, Aspen Custom Modeler (ACM), gPROMS, and Microsoft Excel. SimSinter is used to: (1) open the simulator, (2) initialize the simulation, (3) set variables in the simulation, (4) run the simulation, and (5) get resulting output variables from the simulation.

To drive a particular simulation, SimSinter must be told which input variables to set and which output variables to read when the simulation is finished (there are generally far too many variables in a simulation to set and read them all). Each simulation must have a “Sinter Config File” which records this information. FOCUS keeps the simulation file and the “Sinter Config File” together and sends them to the Turbine gateway when a simulation run is requested.

The configuration is simplified by a GUI included with the SimSinter distribution called, “SinterConfigGUI.” FOCUS can launch the SinterConfigGUI on simulations that have not been configured. To run the “SinterConfigGUI” the user must have:

1. SimSinter distribution installed. SimSinter is installed by the FOCUS bundle installer.
2. The simulation file the user wants to configure. For example, if the user has an Aspen Custom Modeler simulation called BFB.acmf, that file must be on the user’s computer, and the user should know its location.
3. The application used to execute the simulation file. For example, if the user wants to configure an Aspen Custom Modeler simulation called BFB.acmf, Aspen Custom Modeler must be installed on the user’s machine.

The rest of this section details two step-by-step tutorials on configuring a simulation with “SinterConfigGUI.” The first simulation is an Aspen Custom Modeler simulation and the second, Aspen Plus. Please also see the D-RM Builder tutorials for configuring dynamic ACM models. For more details on SimSinter or a tutorial on how to configure a Microsoft Excel file, please see the “SimSinter Technical Manual,” which is included in the FOCUS distribution. The default location is at C:\Program Files (x86)\foqus\foqus\doc. It is also available on the CCSI website.

Aspen Custom Modeler Configuration

1. The “SinterConfigGUI” can be launched from FOCUS, via the Create/Edit button found in File→Add/Update Model to Turbine or “SinterConfigGUI” may be run on its own by selecting SimSinter → SinterConfigGUI from the Start menu.
2. The splash window displays, as shown in Figure 158. The user may click the splash screen to proceed, or wait ten seconds for it to close automatically.



Figure 158: SinterConfigGUI Splash Screen

3. The SinterConfigGUI Open Simulation window displays (Figure 159). If “SinterConfigGUI” was opened from FOCUS, the filename text box already contains the correct file. To proceed immediately click **Open File and Configure Variables** or click **Browse** to search for the file. For this tutorial, the ACM model for bubbling fluidized bed adsorber installed in the FOCUS examples\OUU\BFB_Cap folder is selected (BFB_OUU_COE.acmf). Once the file is selected, click **Open File and Configure Variables**. The user can open a fresh ACM simulation (.acmf file) or an existing SimSinter configuration file. For this example, open a fresh simulation.

Note: Opening the simulation may take a few minutes depending on how quickly Aspen Custom Modeler can be opened.

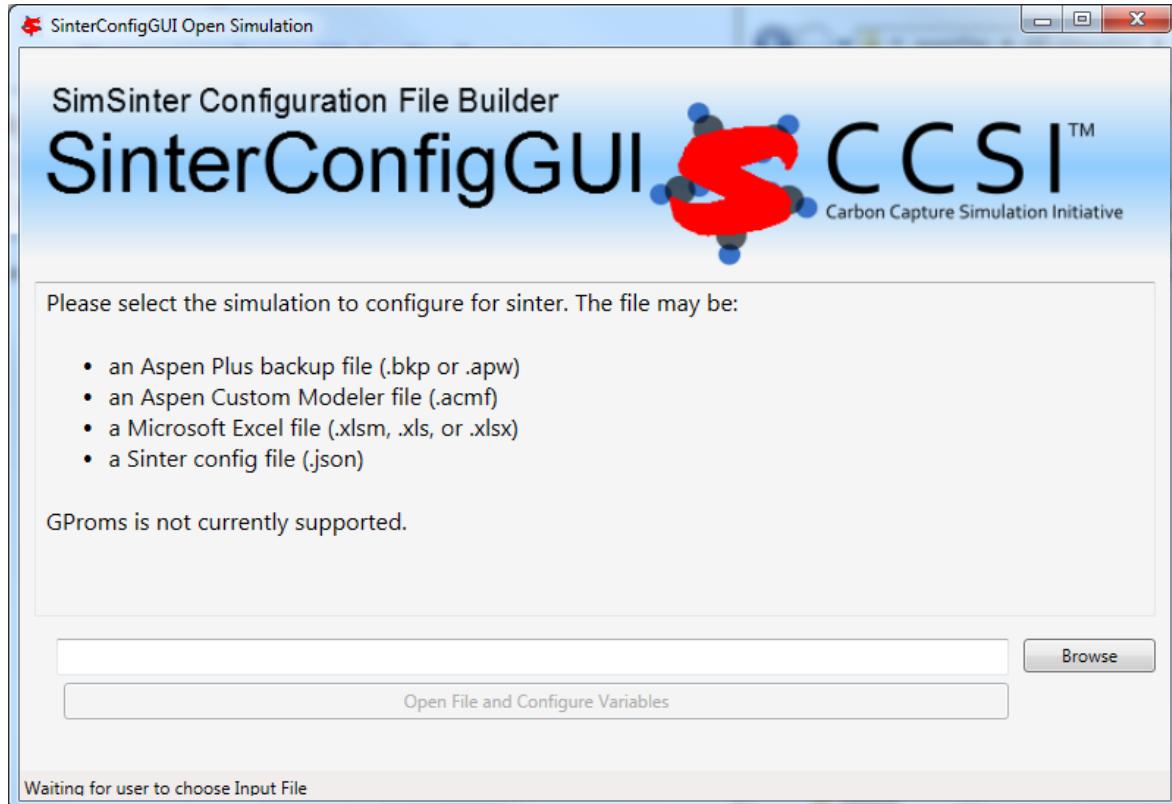


Figure 159: SinterConfigGUI Open Simulation Window

4. Aspen Custom Modeler starts in the background. This is so the user can observe things about the simulation while working on the configuration file.
5. The SinterConfigGUI Simulation Meta-Data window displays. (Figure 160). The first and most important piece of metadata is **SimSinter Save Location** at the top of the window. This is where the sinter configuration file is saved. The system attempts to locate a reasonable file location and file name; however, the user must confirm the correct file location, since it automatically overwrites whatever file name currently exists.

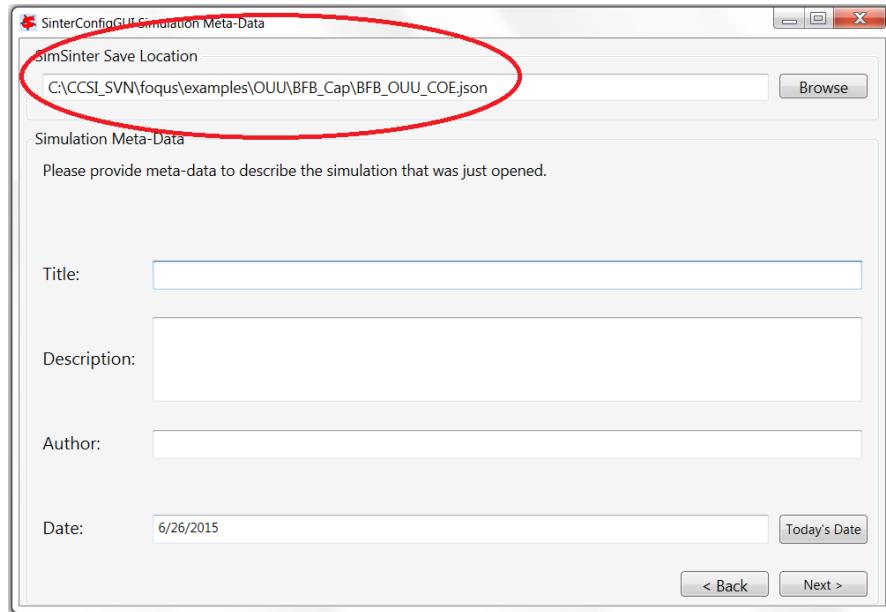


Figure 160: SinterConfigGUI Simulation Meta-Data Page Save Name Text Box

6. Continue to complete the remaining fields and then click Next (Figure 161).

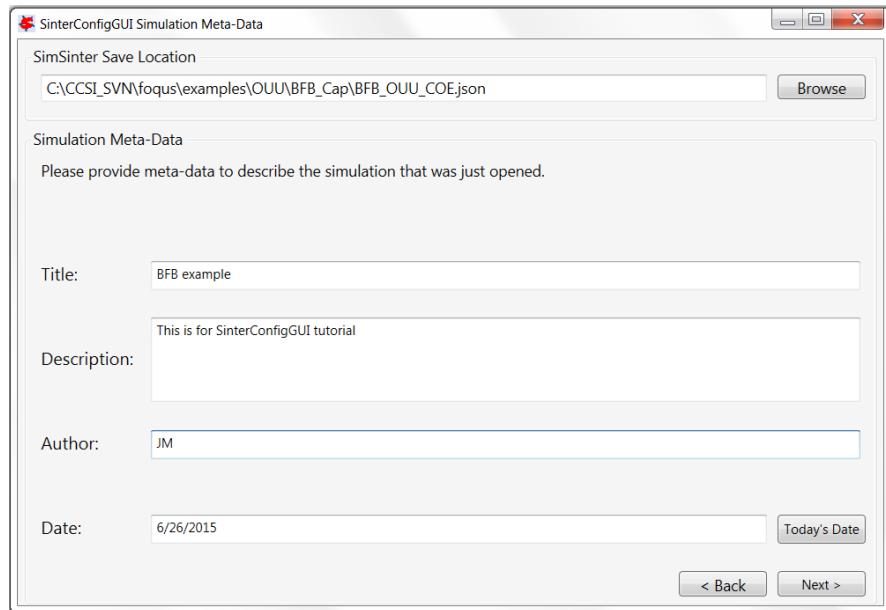


Figure 161: SinterConfigGUI Simulation Meta-Data Page with Data Completed

7. In the **SinterConfigGUI Variable Configuration Page**, (Figure 162) notice that the ACM **Selected Input Variables**: TimeSeries, Snapshot, RunMode, printlevel and homotopy are already included in the input variables. TimeSeries and Snapshot are for dynamic simulations. RunMode can be either “Steady State” or “Dynamic”. The Dynamic mode requires

a dynamic ACM model. For this simulation, the RunMode is Steady State. The **homotopy** variable can be set to “1” so that homotopy is on by default. Notice that the **Dynamic** column (the first column) in each row contains a checkbox, enabling the user to select if the input variable in the row is a dynamic variable. Also notice that a **Variable Type** search box is on the left. This search is exactly the same as Variable Find on the Tools menu in Aspen Custom Modeler. Please refer to the ACM documentation for details on search patterns.

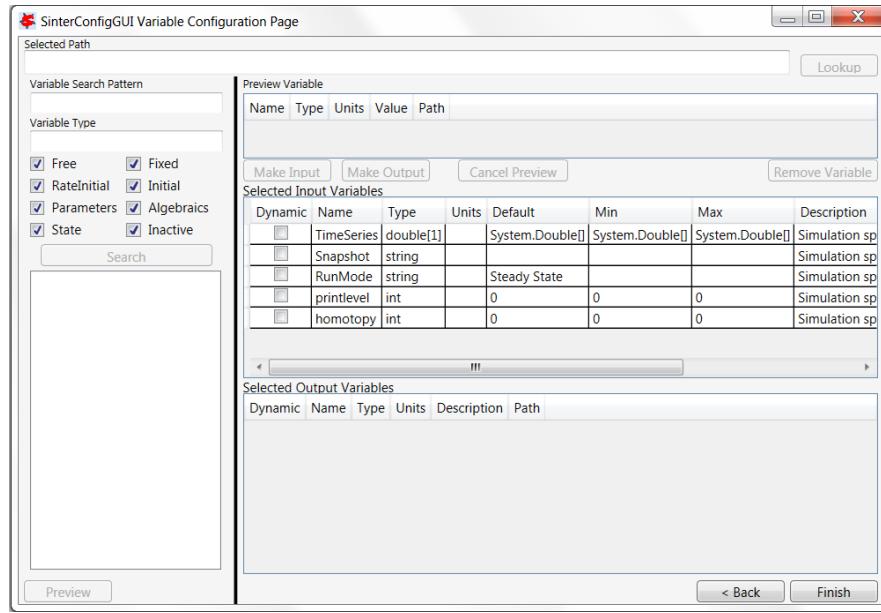


Figure 162: SinterConfigGUI Variable Configuration Page before Input

8. A search for everything in the “BFBAdsT” block has been selected. The following **Search in Progress** dialog is displayed (Figure 163). Sometimes large searches take a while.

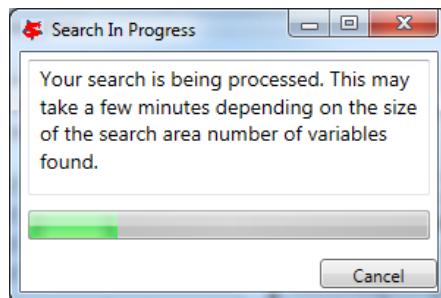


Figure 163: Search in Progress Bar Page

9. First, select the “BFBadsT.A1” scalar variable in the **Selected Path** field (Figure 164).

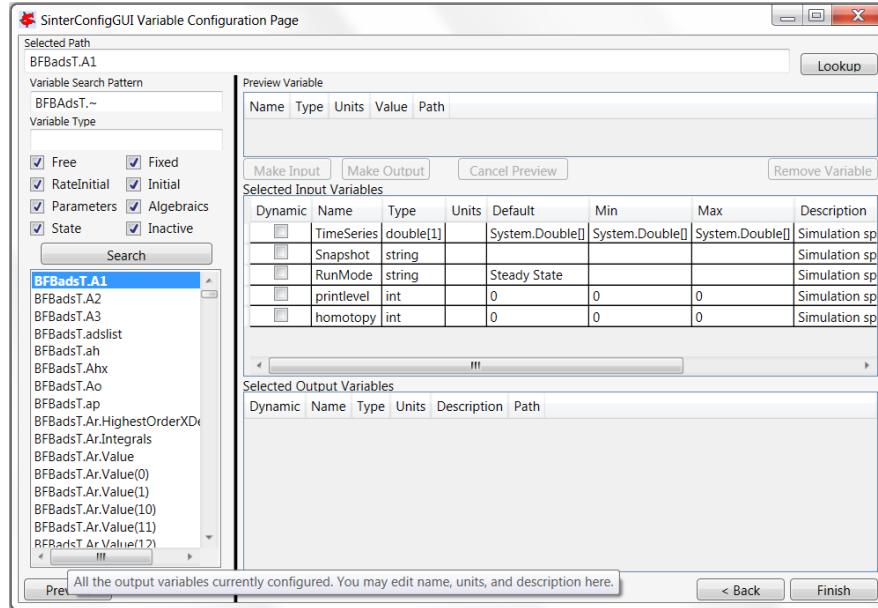


Figure 164: SinterConfigGUI Variable Configuration Page BFBadsT.A1 Selected

10. If the user double-clicks, presses Enter, or clicks **Preview** or **Lookup**, information displays in the **Preview Variable** section (Figure 165). Here, the user can verify the variable choices.

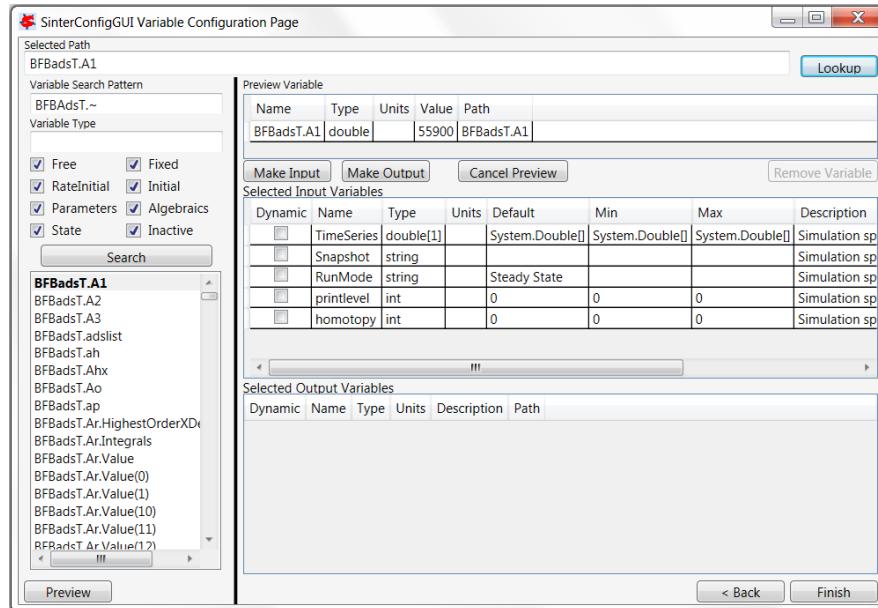


Figure 165: SinterConfigGUI Variable Configuration Page BFBadsT.A1 Preview

11. “BFBadsT.A1” is the correct variable; therefore, click **Make Input**. Information displays in the **Selected Input Variables** section (Figure 166).

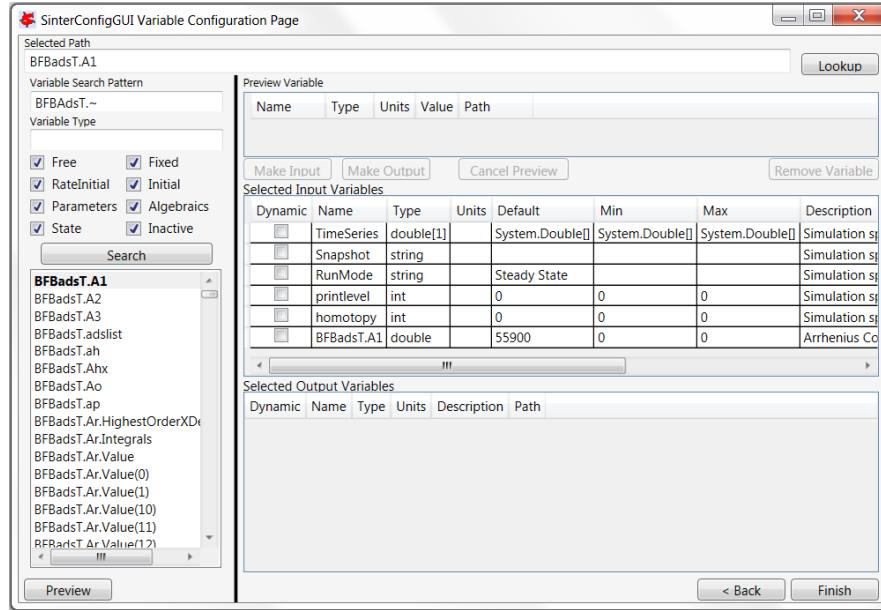


Figure 166: SinterConfigGUI Variable Configuration Page BFBadsT.A1 Made Input

12. Change the variable name from “BFBadsT.A1” to something more descriptive (e.g., “WaterA”). Set **Name**, **Description** and **Min/Max** as shown in Figure 167.

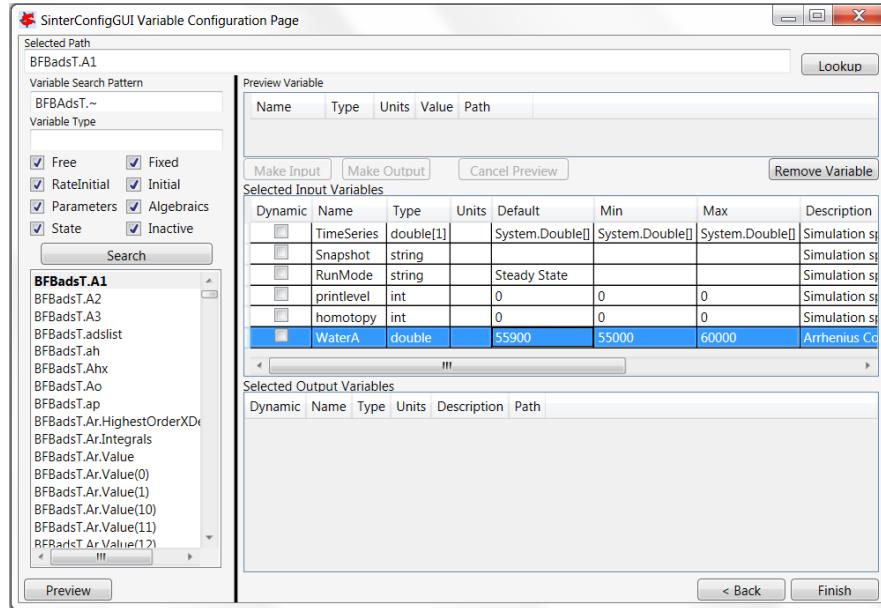


Figure 167: SinterConfigGUI Variable Configuration Page BFBadsT.A1 Change Name

13. One input variable is now displayed (Figure 168). At least one output variable is required. In this example, the vector of calculated bubble sizes is wanted. Scroll down under **Search** and select “BFBadsT.db.Value,” “BFBadsT.db.Value(0),” “BFBadsT.db.Value(1),” etc. If a

name with a number in parenthesis at the end is selected, it is a specific entry in the vector. If a basic name is selected (“BFBadsT.db.Value”), the entire vector is displayed. Select the whole vector and click **Preview**.

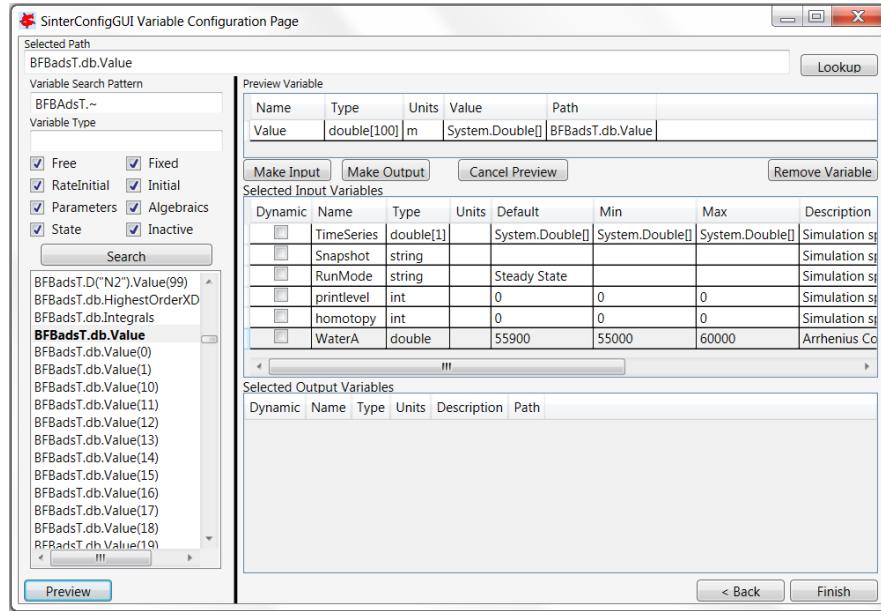


Figure 168: SinterConfigGUI Variable Configuration Page Vector Preview

- Click **Make Output** if the variable the user wants is selected. Notice that this variable has a unit “m” (Figure 169).

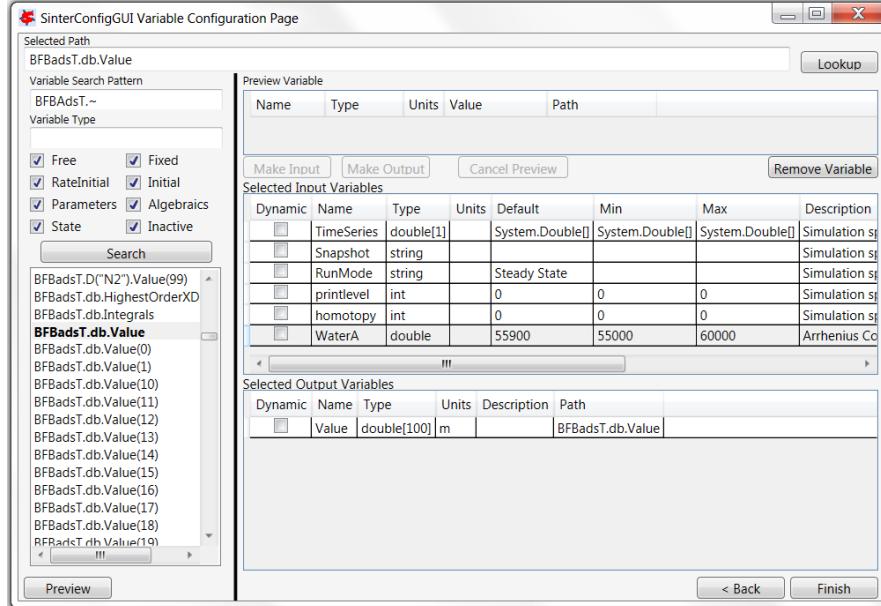


Figure 169: SinterConfigGUI Variable Configuration Page Vector As Output

15. Change the Name of the variable to “Diameter.” Bubble size is measured in meters; however, meters should be converted to millimeters (mm). Now, the output from the simulation should present bubble diameter in mm (Figure 170). Internal to the simulation, the unit remains “m.”

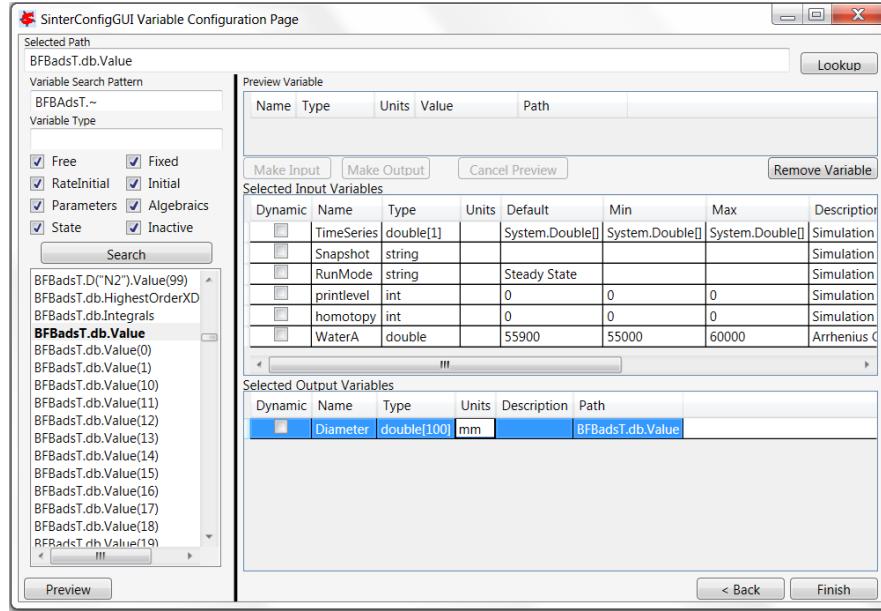


Figure 170: SinterConfigGUI Variable Configuration Page Output Change Units

16. To add a single item in a vector, select “BFBadsT.Ar.Value(1)” and click Make Input (See Figure 171). To remove item that was just added, select it and click Remove Variable.

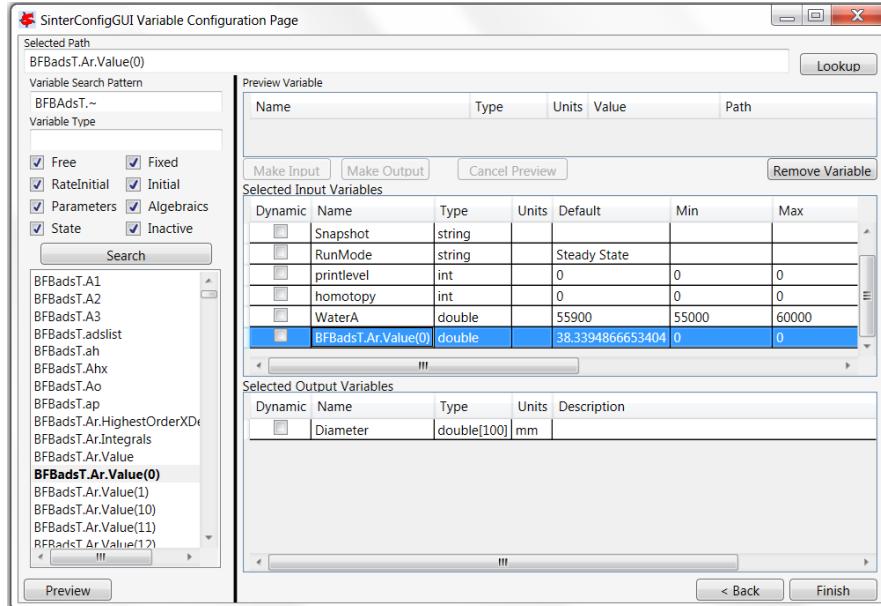


Figure 171: SinterConfigGUI Variable Configuration Page Removal Demo

17. Select the correct variable vector “BFBadsT.Ar.Value” and make it an input (Figure 172). Notice that a **Default** or **Min/Max** cannot be set in the GUI for a vector. The correct defaults (from the simulation) are set automatically. To change the **Min/Max** values, the user must edit the JSON file in a text editor.

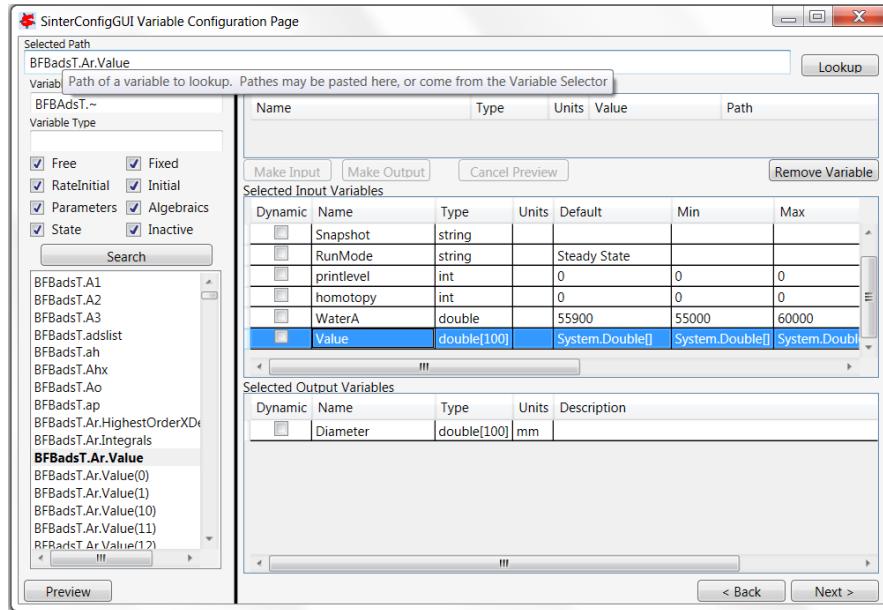


Figure 172: SinterConfigGUI Variable Configuration Page Read Input

18. Click **Next** to display the SinterConfigGUI Vector Default Initialization window as shown in Figure 173. Since the input variable “Value” is a vector, its default values can be modified in the window. In this case there is no need to change the values.

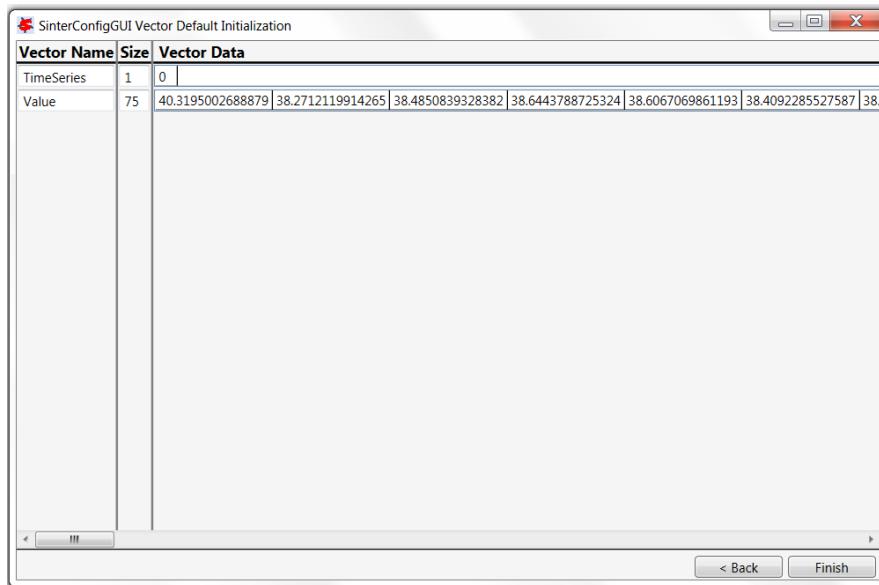


Figure 173: SinterConfigGUI Vector Default Initialization Input Page

19. The simulation is now setup. Save the configuration file by clicking **Finish**. The file is saved to the location specified on the SinterConfigGUI Simulation Meta-Data page. Clicking **Finish** will close the SinterConfigGUI, but NOT Aspen Custom Modeler. The user must close ACM manually.
 20. If “SinterConfigGUI” was launched from FOCUS, the path to the configuration file is automatically passed to FOCUS. The next step in FOCUS is to click **OK** in the Add/Update Turbine Model window. FOCUS may then be used to upload it to the Turbine gateway. If “SinterConfigGUI” was not launched from FOCUS (e.g., it was launched from the Start menu), the configuration file name must be entered in FOCUS manually.

Aspen Plus Configuration

1. The initial steps for opening a simulation and entering metadata for an Aspen Plus simulation are similar to ACM. Refer to the SimSinter ACM tutorial 9.2. In this tutorial, a flash model “Flash_Example.bkp” installed in the “C:\SimSinterFiles\Aspen_Plus_Install_Test” is used as an example. Open the Aspen Plus file and enter the metadata as shown in Figure 174.

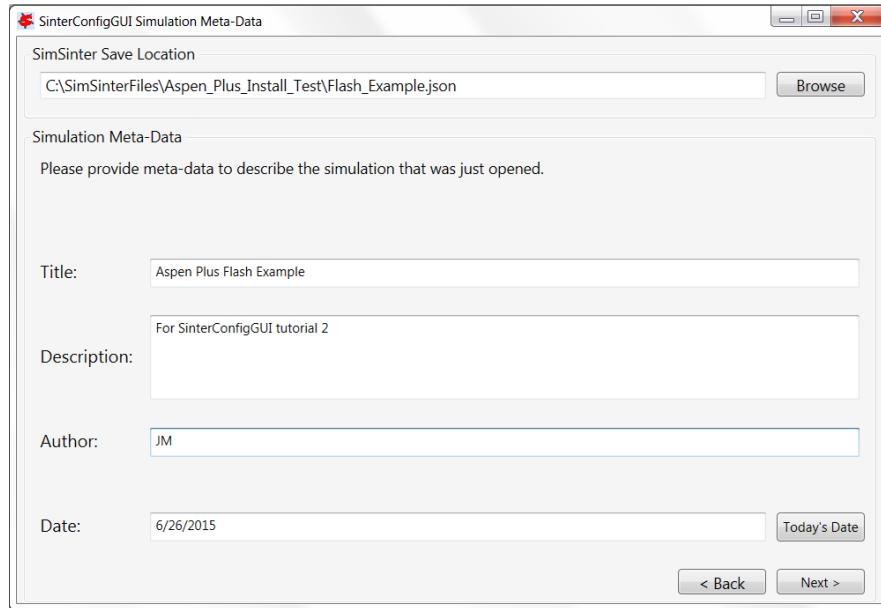


Figure 174: SinterConfigGUI Simulation Meta-Data with Data Completed

2. The **SinterConfigGUI Variable Configuration Page** displays as illustrated in Figure 175. Aspen Plus has no settings, so there are no setting variables in the input section. Unlike ACM, AspenPlus displays the **Variable Tree** on the left side, so the user can explore the tree as is done in Aspen Plus Tools → Variable Explorer. Unfortunately, searching is not possible.

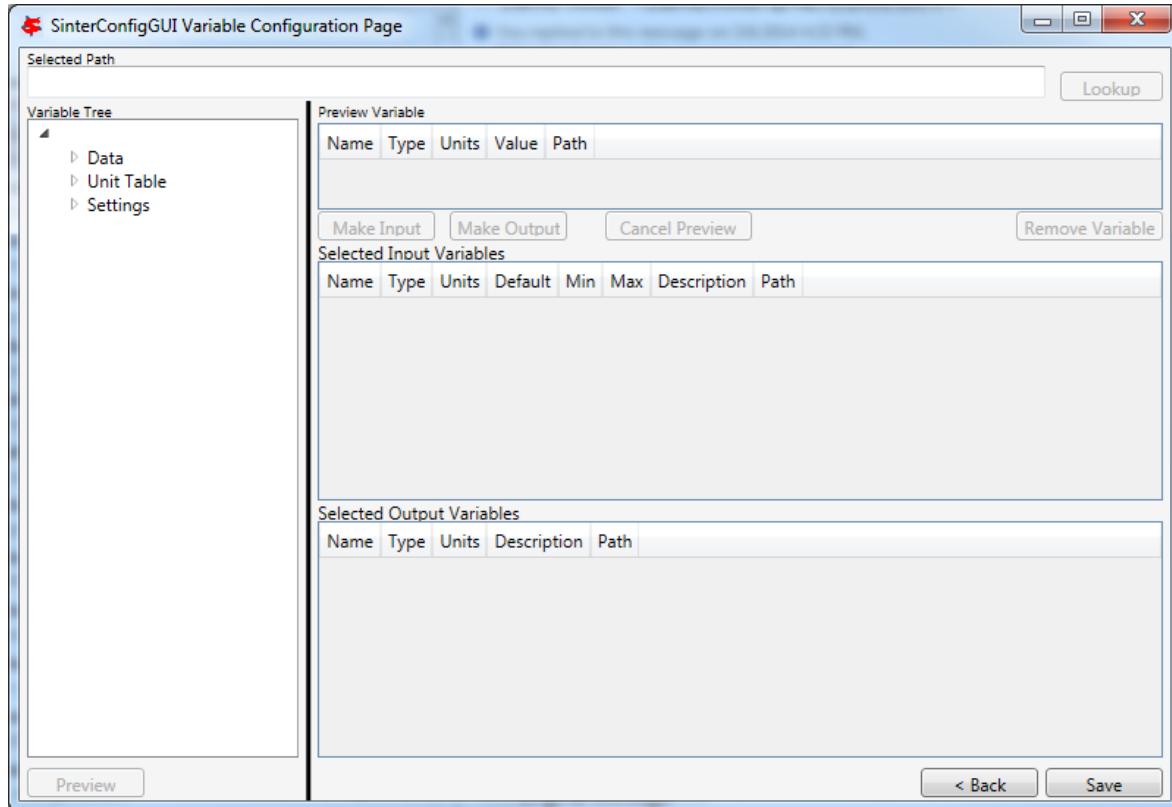


Figure 175: SinterConfigGUI Variable Configuration Page Empty Variables

3. **Variable Tree** nodes can be expanded for searching (Figure 176).

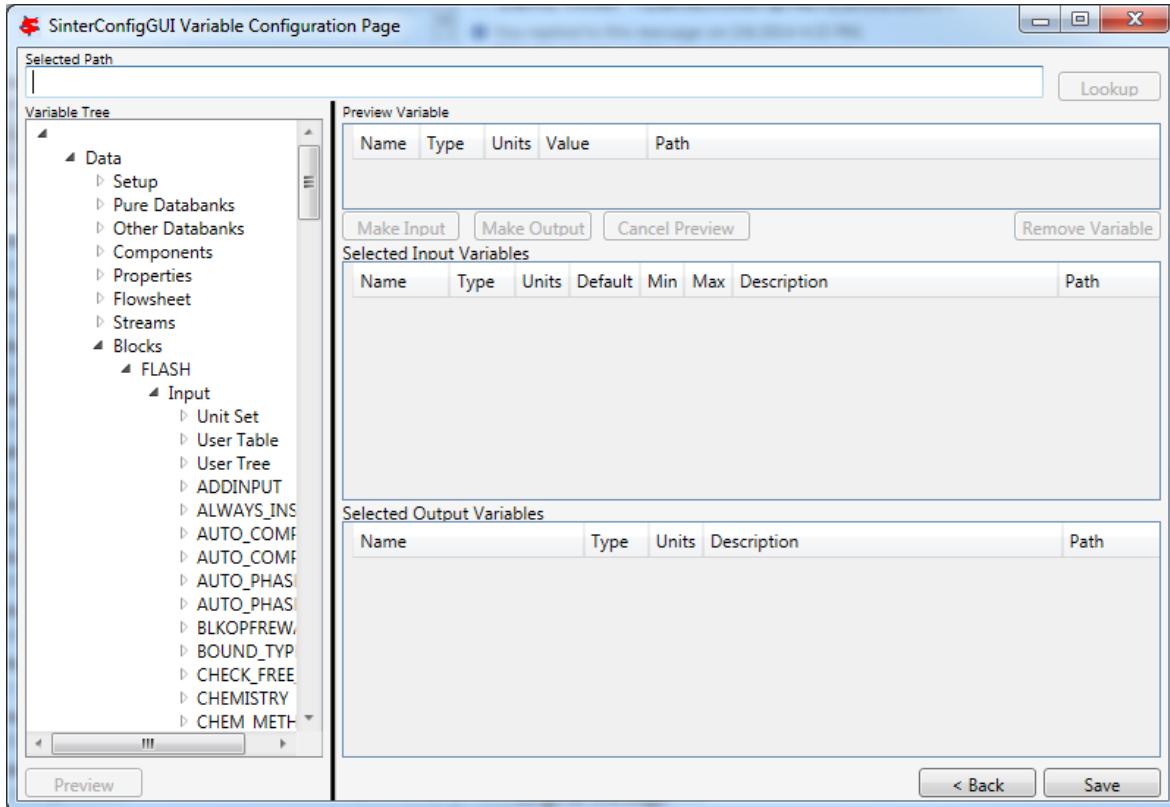


Figure 176: SinterConfigGUI Variable Configuration Page Expanded Aspen Plus Variable Tree

4. The user can type the node address directly into the **Selected Path** field (this is useful for copy/paste from Aspen Plus' Variable Explorer) (Figure 177). Click **Lookup** or **Preview** (which automatically causes the tree to expand and selects selected variables in the **Variable Tree**).

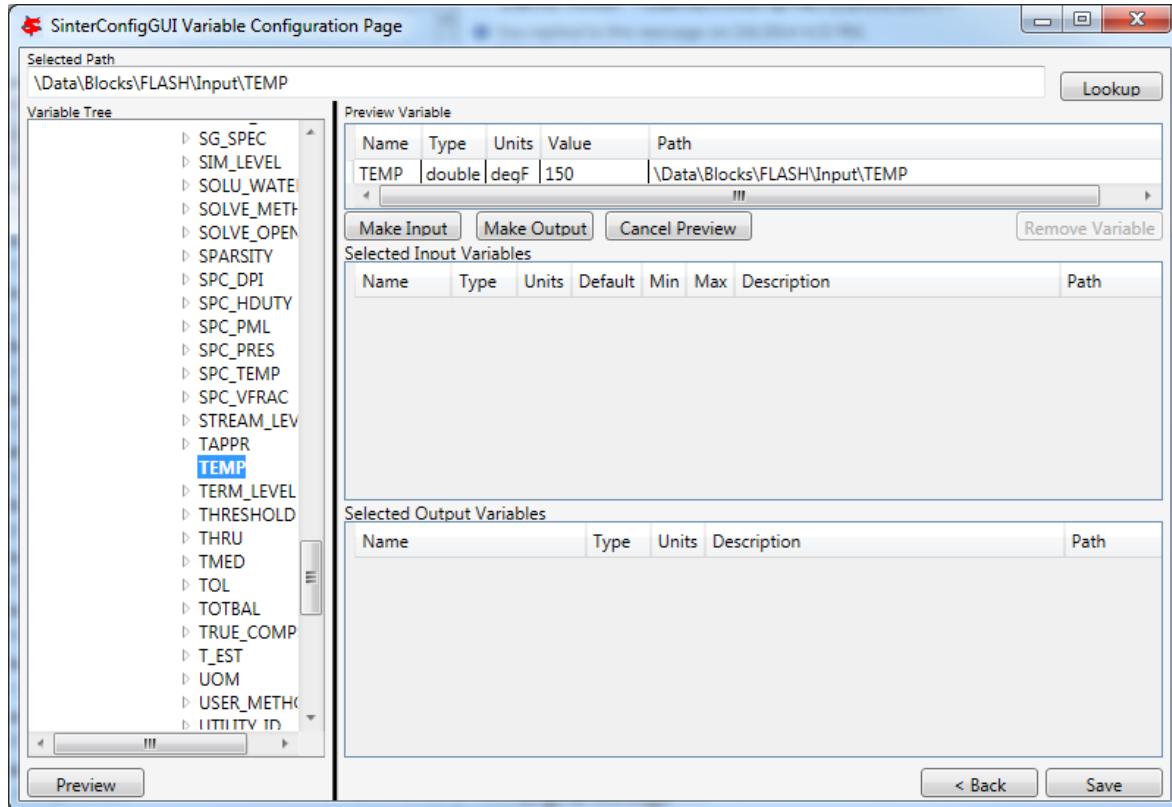


Figure 177: SinterConfigGUI Variable Configuration Page Aspen Plus Variable Selected

5. To make the temperature of the Flash chamber an **Input Variable**, click **Make Input**. Additionally, the user can **Name** the variable, fix the **Description**, and enter the **Min/Max** fields by clicking on the appropriate text and entering it.

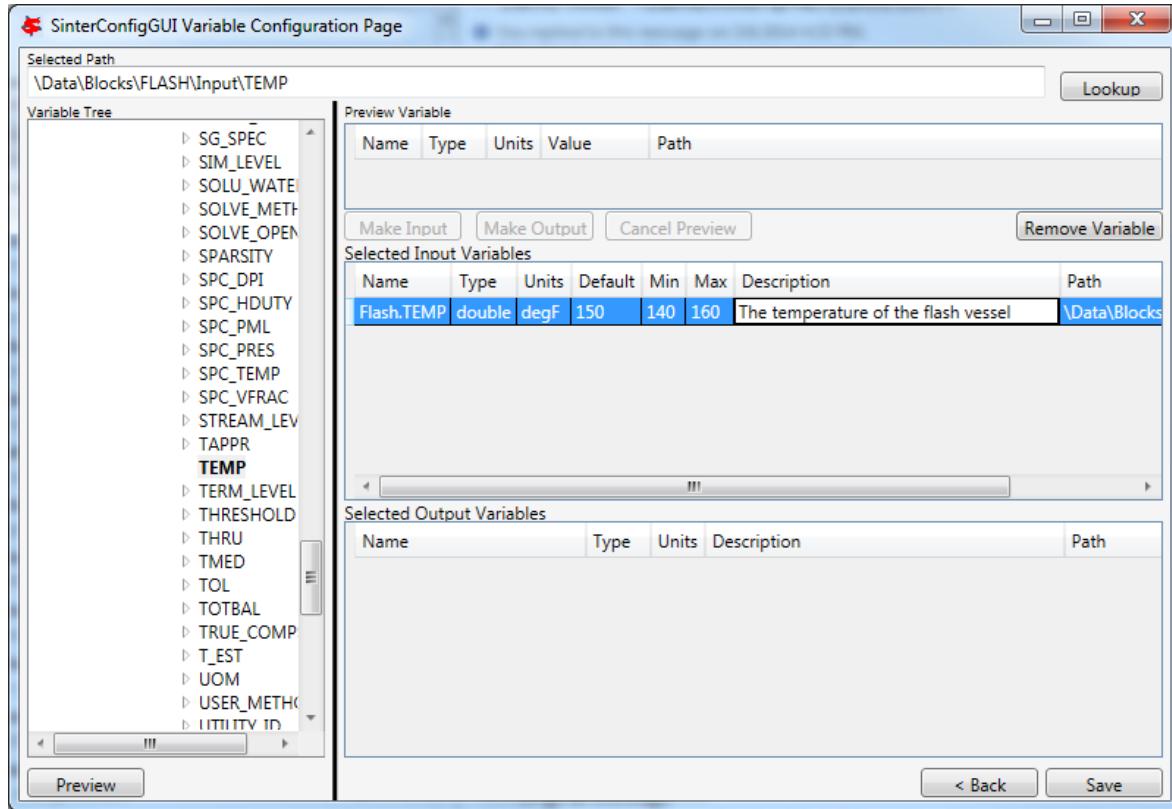


Figure 178: SinterConfigGUI Variable Configuration Page Input Variable

6. Select an **Output Variable**, **Preview** it, and click **Make Output**. Next, update the fields as with the **Input Variable** to give a better **Name** and **Description**.

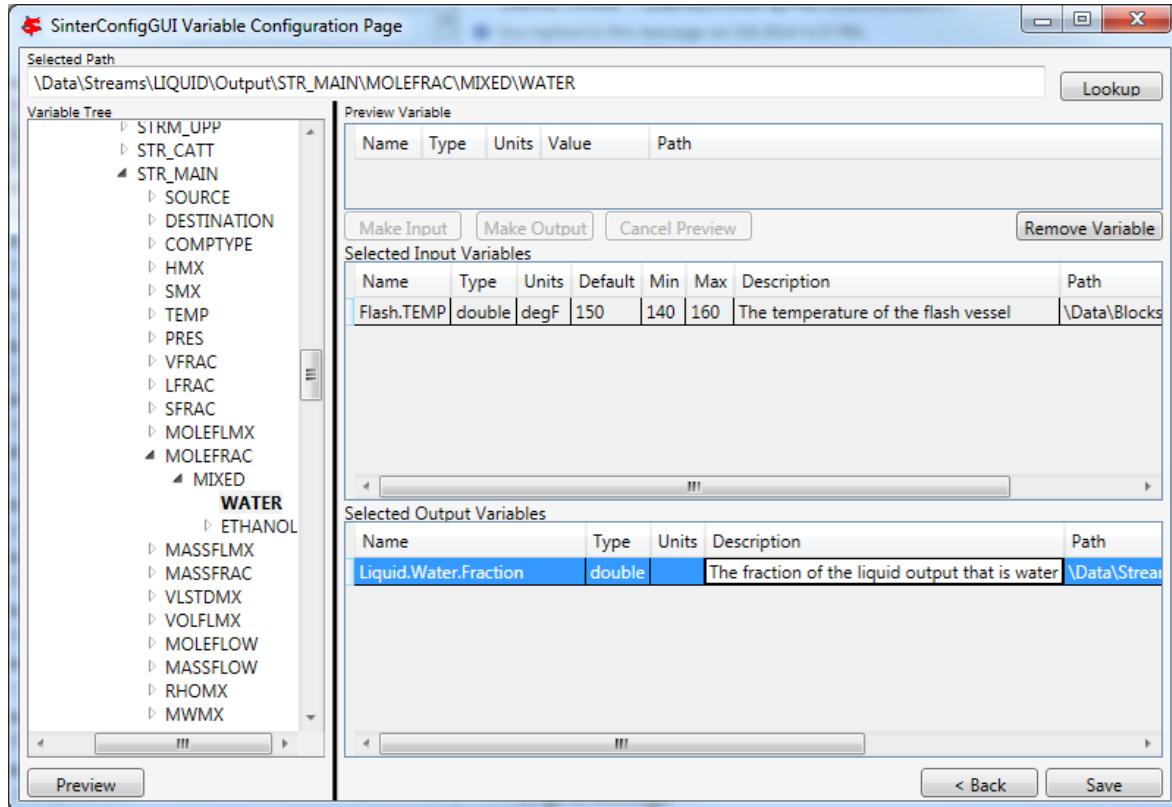


Figure 179: SinterConfigGUI Variable Configuration Page Output Variable

7. The task is complete. Save it by clicking **Save** or CTRL+S. The file is saved to the location specified in the SinterConfigGUI Simulation Meta-Data page. If the user wishes to save a copy under a different name, navigate back to the SinterConfigGUI Simulation Meta-Data page and change the name.

Microsoft Excel Configuration

1. The “SinterConfigGUI” can be launched from FOCUS, via the **Create/Edit** button found in **File→Add/Update Model to Turbine** or “SinterConfigGUI” may be run on its own by selecting **CCSI Tools → FOCUS → SinterConfigGUI** from the Start menu.
2. The splash window displays, as shown in Figure 158. The user may click the splash screen to proceed, or wait 10 seconds for it to close automatically.



Figure 180: SinterConfigGUI Splash Screen

3. The SinterConfigGUI Open Simulation window displays (Figure 159). If “SinterConfigGUI” was opened from FOCUS, the filename text box already contains the correct file. To proceed immediately click **Open File and Configure Variables** or click **Browse** to search for the file. For this tutorial, a fresh copy of the BMI test is opened. It can be found at: C:\SimSinterFiles\Excel_Install_Test\exceltest.xlsx.

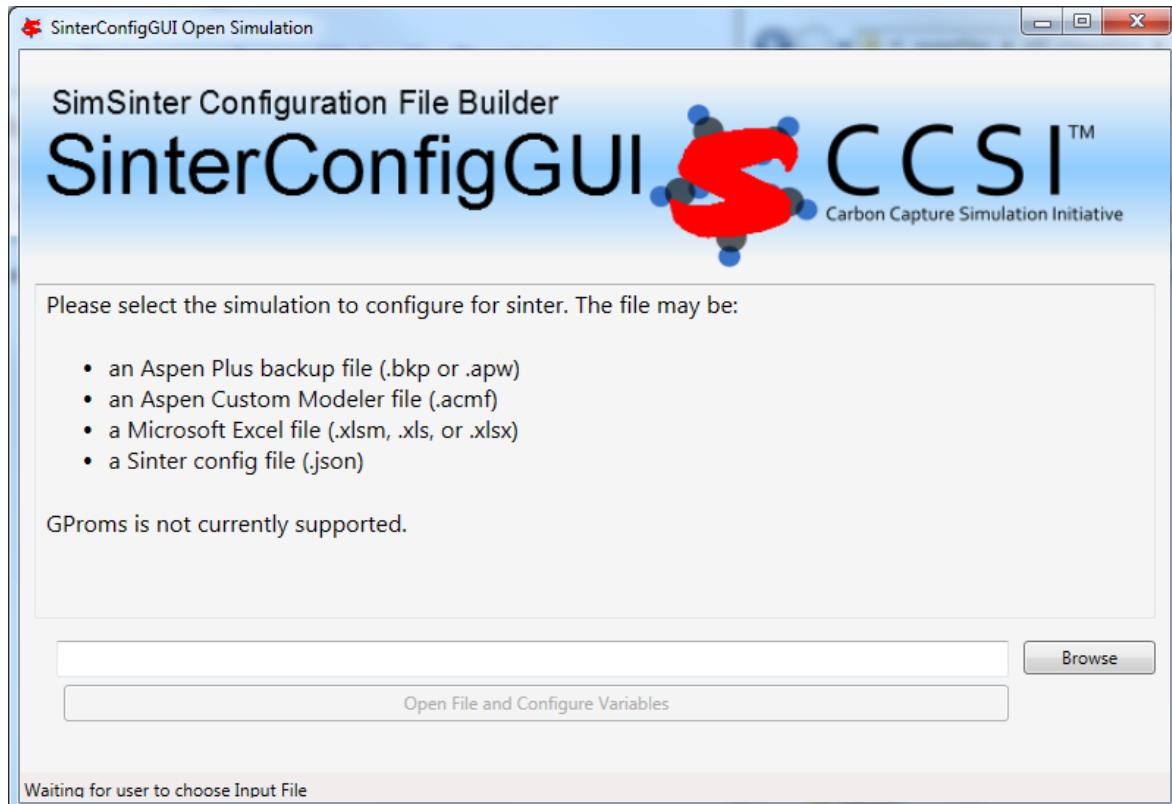


Figure 181: SinterConfigGUI Open Simulation Screen

4. Microsoft Excel starts in the background. This is so the user can observe things about the worksheet while working on the configuration file.
5. In the “SinterConfigGUI” the SinterConfigGUI Simulation Meta-Data page is now displayed (Figure 182). The first and most important piece of metadata is **Save Location** at the top of the window. This is where the sinter configuration file is saved. The system attempts to locate a reasonable file location and file name; however, the user must confirm the correct file location, since it automatically overwrites whatever filename currently exists.

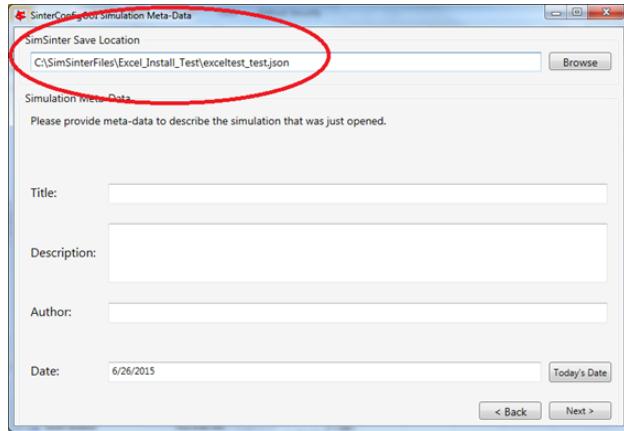


Figure 182: SinterConfigGUI Simulation Meta-Data Save Text Box

6. Continue to complete in the remaining fields and click **Next**.
7. In the SinterConfigGUI Variable Configuration Page, (Figure 183) notice that the Excel setting variable **macro** is already included in the **Selected Input Variables**. If the Excel spreadsheet has a macro that should be run after SimSinter sets the inputs, but before SimSinter gets the outputs, enter the macros name in the **Name** text box. If the default is left blank, no macro is run (unless a name is supplied in the input variables when running the simulation).

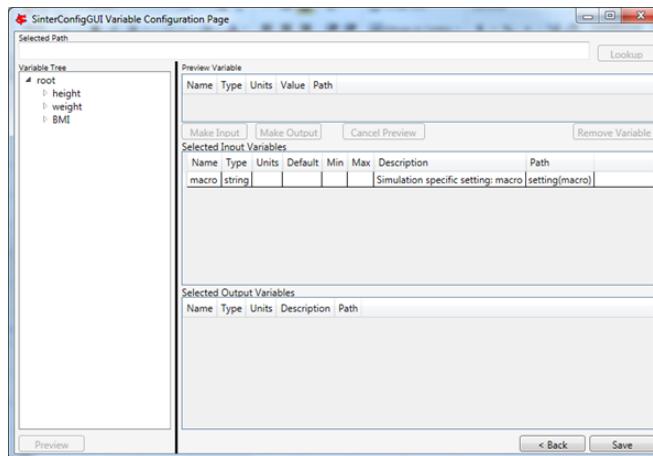


Figure 183: SinterConfigGUI Variable Configuration Page before Input

8. The Excel simulation has the same **Variable Tree** structure as Aspen Plus, as shown in (Figure 184). Only the variables in the active section of the Excel spreadsheet appear in the **Variable Tree**. If, for some reason, a cell does not appear the in tree, the user may manually enter the cell into the **Selected Path** text box. In this case, select the "height\$C\$4" variable. Note: Row is first in the **Variable Tree**, yet column is first in the **Path**.

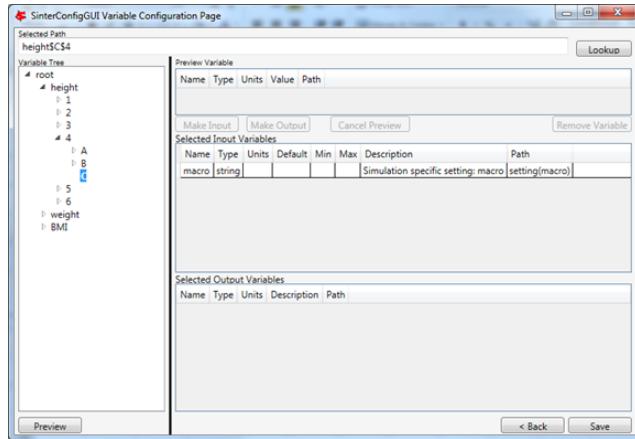


Figure 184: SinterConfigGUI Variable Configuration Page Selecting a Variable from the Excel Variable Tree

9. If the user double-clicks, presses enter, clicks Preview, or clicks Lookup, the variable will be displayed in the Preview Variable frame. Click the Make Input button to make the variable an input variable. Now the variable is in the Selected Input Variables section, and its meta-data may be edited (Figure 185).

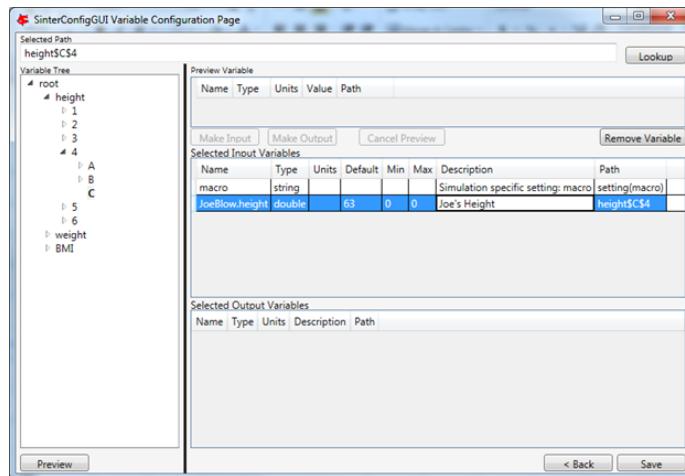


Figure 185: SinterConfigGUI Variable Configuration Page Description “Joe’s Height”

10. Enter an output variable (such as, “BMI\$C\$3”), by selecting the variables in the Variable Tree, clicking Preview, and then clicking Make Output (Figure 186).

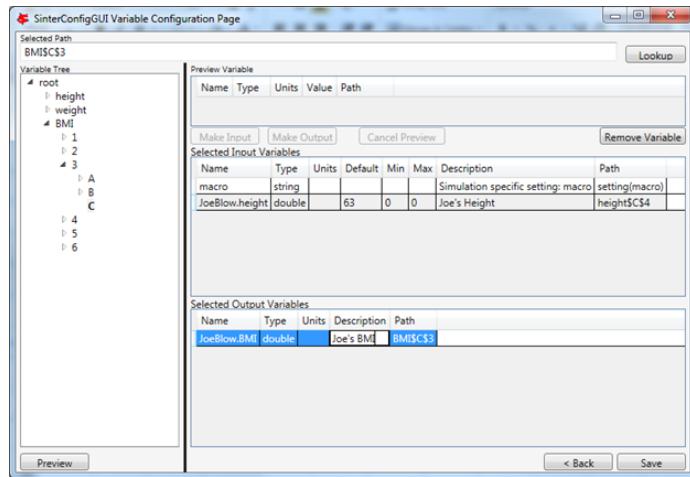


Figure 186: SinterConfigGUI Variable Configuration Page Selecting Excel Output Variables

11. The simulation is now set up. To save the configuration file, click **Finish** or press CTRL+S. The file is saved to the location that was set on the SinterConfigGUI Simulation Meta-Data window. A user can save a copy under a different name, by navigating back to the SinterConfigGUI Simulation Meta-Data window using **Back**, and then changing the name. This creates a second version of the file.

gPROMS Configuration

gPROMS is significantly different from the other simulators SimSinter supports, and the workflow is also significantly different. If you plan to use gPROMS simulations with FOCUS, the CCSI team strongly encourages you to read the “SimSinter gPROMS Technical Manual,” which is included in the FOCUS distribution. The default location is at C:\Program Files (x86)\foqus\foqus\doc. It is also available on the CCSI website.

Unlike Aspen, changes must be made to the gPROMS simulation process in order to work with SimSinter. Therefore, this section consists of a series of tutorials for every step of configuring gPROMS and SimSinter to work together. All the tutorials are required in order to have a gPROMS simulation be runnable with SimSinter. They are divided up to make later reference easier.

Configuring gPROMS to Work with SimSinter

Unlike Aspen, changes have to be made to the gPROMS simulation process in order to work with SimSinter. In fact, SimSinter does not define the inputs to the simulation, gPROMS does. On the other hand, gPROMS does not determine the outputs, SimSinter does. This odd and counter-intuitive situation is the result of how gPROMS gO:Run_XML is designed.

The modification to the gPROMS simulation must be done by a developer with an intimate understanding of the simulation, usually the simulation writer. In some cases additional variables may need to be added to handle an extra step between taking the input and inserting it into the variable where gPROMS will use the data.

1. Open the gPROMS simulation file (ends in .gPJ) in ModelBuilder 4.0 or newer. For this example, use the gPROMS install test file “BufferTank.FO.gPJ”, found in:
C:\SimSinterFiles\gPROMS_Test\BufferTank.FO.gPJ
Double-click on the .gPJ file to open ModelBuilder, as shown in Figure 187.

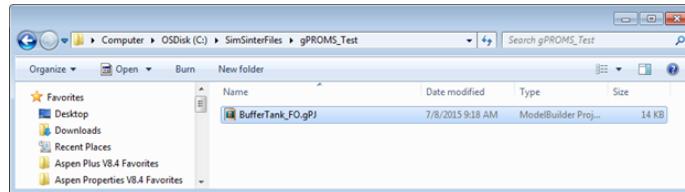


Figure 187: Opening BufferTank in gPROMS Model Builder

2. This simulation was originally a simple BufferTank simulation. However, it was modified into an example of all the different kinds of variables the user can pass into gPROMS via SimSinter. Therefore, it has a lot of extra variables that do not really do anything, with very generic names, like “SingleInt.” The simulation consists of a single model, “BufferTank”, that contains all the simulation logic, and most of the parameter and variable declarations. The SimSinter simulation will change some of these PARAMETERS and VARIABLES to change the output of the simulation.

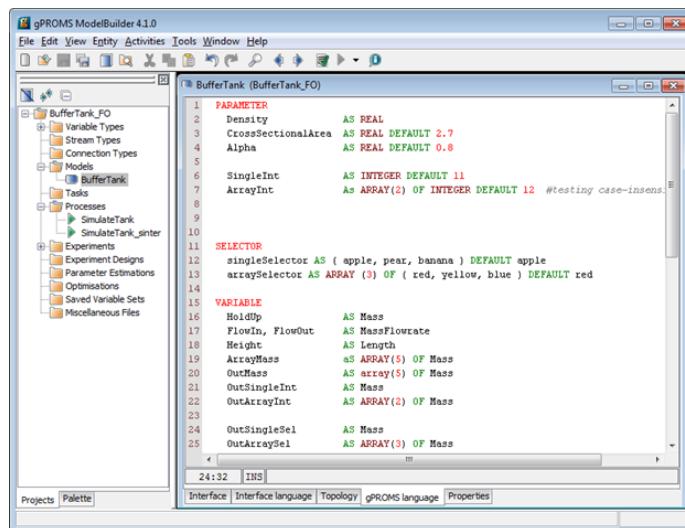
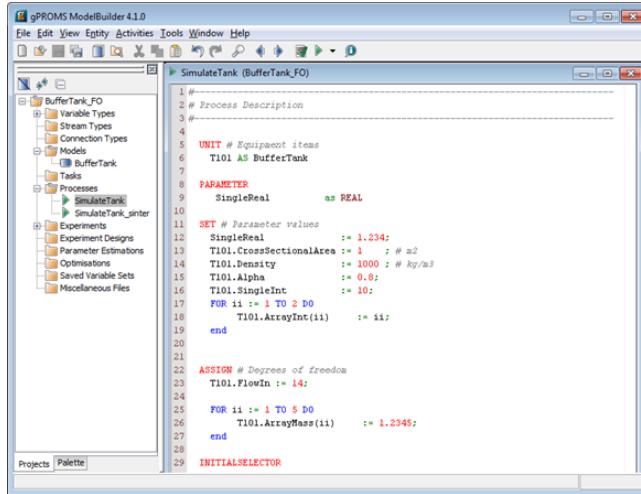


Figure 188: Viewing BufferTank in gPROMS Model Builder

3. The example file contains two Processes. SimSinter can only run gPROMS Processes, so any gPROMS simulation must be driven from a Process. “SimulateTank” is the original BufferTank example with hardcoded values, “SimulateTank_Sinter” contains the example of setting values with Sinter. The “SimulateTank_Sinter” example will be recreated in this tutorial.



```

gPROMS ModelBuilder 4.1.0
File Edit View Entity Activities Tools Window Help
Projects Palette
SimulateTank (BufferTank_FO)
1 #-- Process Description
2 #-- Equipment items
3
4
5 UNIT # Equipment items
6 T101 AS BufferTank
7
8 PARAMETER
9 SingleReal as REAL
10
11 SET # Parameter values
12 SingleReal := 1.234;
13 T101.CrossSectionalArea := 1; # m2
14 T101.Density := 1000; # kg/m3
15 T101.Alpha := 0.8;
16 T101.SingleInt := 10;
17 FOR ii := 1 TO 2 DO
18   T101.ArrayInt(ii) := ii;
19 end
20
21
22 ASSIGN # Degrees of freedom
23 T101.FlowIn := 14;
24
25 FOR ii := 1 TO 5 DO
26   T101.ArrayMass(ii) := 1.2345;
27 end
28
29 INITIALSELECTOR

```

Figure 189: Viewing SimulateTank in gPROMS Model Builder

4. First copy the existing hard-coded Process “SimulateTank”.

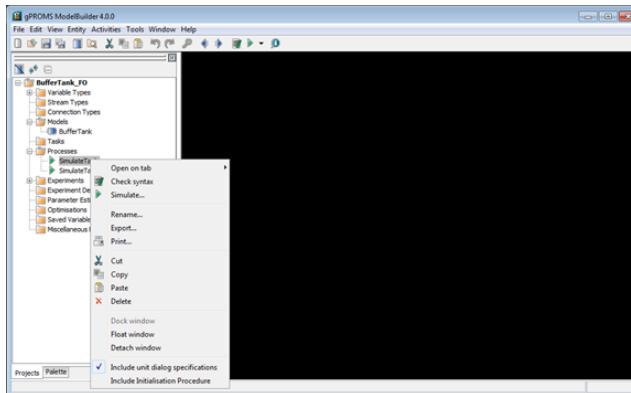


Figure 190: Copying SimulateTank

5. Right-click on Processes and select Paste to make a new process.

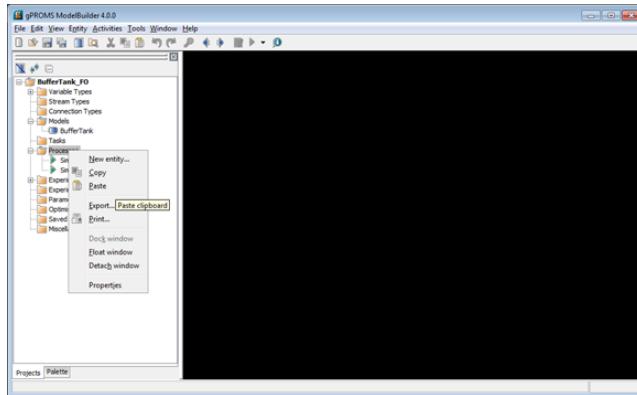


Figure 191: Paste SimulateTank

6. The new process will be named “SimulateTank_1”. Rename the process by right-clicking on it and selecting **Rename**.



Figure 192: Rename SimulateTank

7. Now open up the new “SimulateTank_tutorial” Process. It has the same hard-coded values as “SimulateTank”.

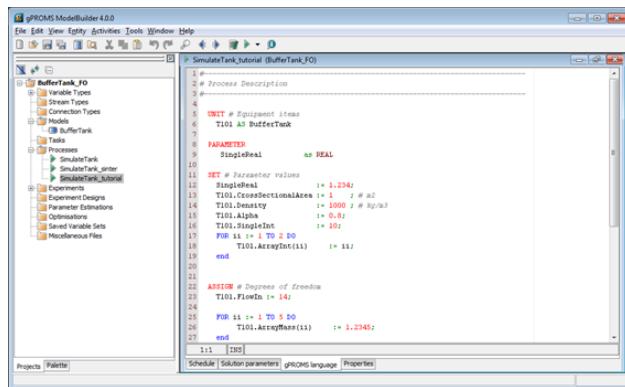


Figure 193: Opening SimulateTank_tutorial

8. First, the user needs to add a FOREIGN_OBJECT named “FO” in the PARAMETER section. Then the user needs to set that FOREIGN_OBJECT to “SimpleEventFOI::dummy” in the SET section. This FOREIGN_OBJECT is how inputs are received from SimSinter.

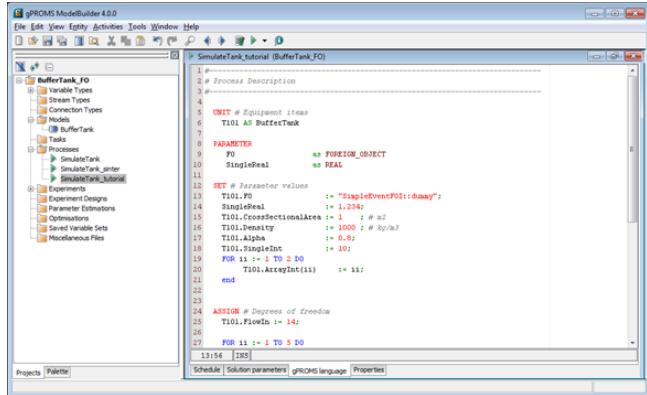


Figure 194: Adding the FOREIGN_OBJECT

9. This particular simulation has a large number of input variables that simply demonstrate how to set different types. These are named based on their type. Any variable named similarly to “SingleInt” or “ArraySelector” can be safely ignored for this tutorial. For a full list of the methods for setting different types see the later section specifically for covering that. Any variable in the simulation can be an input, whether it is defined in the Process or one of the models referenced by the process, or in a model referenced by a model, etc. All inputs take their values from the FOREIGN_OBJECT defined, followed by the type name, two underscores, the input variable name, an open parenthesis, an optional index variable (for arrays), and closed with a close parenthesis and a semicolon. For a scalar:

FO.<Type>_<InputName>();

SimSinter can only handle arrays inputted in FOR loops such as:

```
FOR ii := 1 TO <array size> DO
    <Array Name>(ii) := FO.<Type>1..<Input Name>(ii);
END
```

For this example the user only really needs to set “T101.Alpha” in PARAMETER, “T101.FlowIn” in ASSIGN, and “T101.Height” in INITIAL.

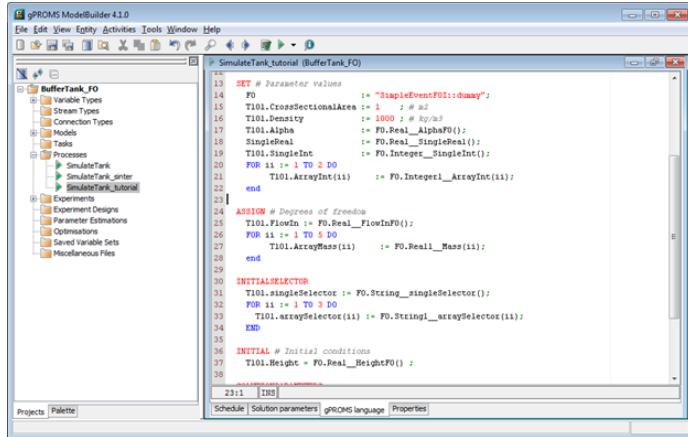


Figure 195: Setting up Input Variables

- Now test “SimulateTank_tutorial” by selecting it and clicking the green **Simulate** triangle. When the simulation runs it will ask for every input variable the user has defined. For the example variables that do not effect the simulation, such as “SingleInt”, any valid value will work. For the values that do effect the simulation, these values work:

REAL_AlphaFO = .08

REAL_FlowInFO = 14

REAL_HeightFO = 7.5

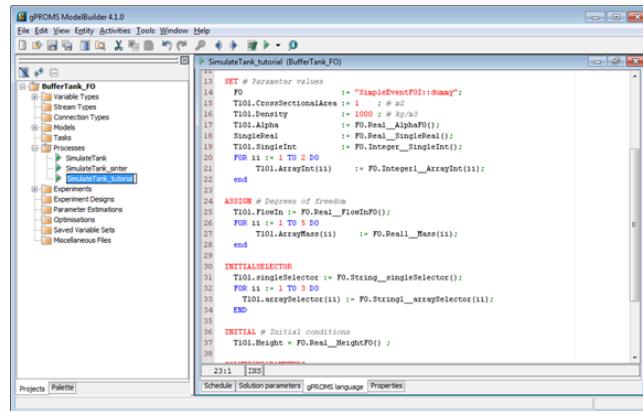


Figure 196: Testing SimulateTank_Tutorial

Exporting an Encrypted Simulation to Run with SimSinter

SimSinter can only run encrypted gPROMS simulations. These files have the .gENCRYPT extension. If the additions to the simulation for reading input variables ran correctly in the previous section, the user is ready to export that process for use by SimSinter.

- Right-click on the Process to export (“SimulateTank_tutorial”) and select **Export**.

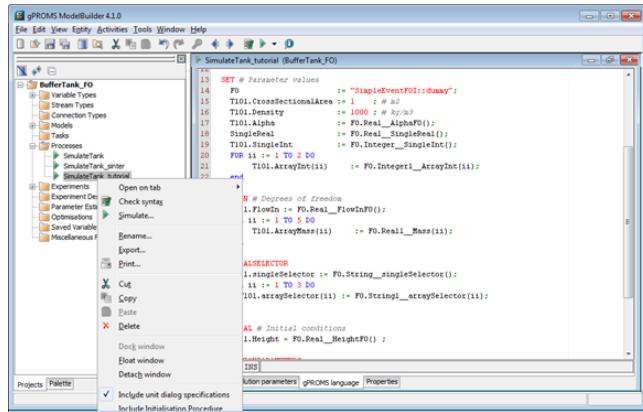


Figure 197: Select “Export”

2. In the resulting Export window, select **Encrypted input file for simulation by gO:RUN** and click **OK**.

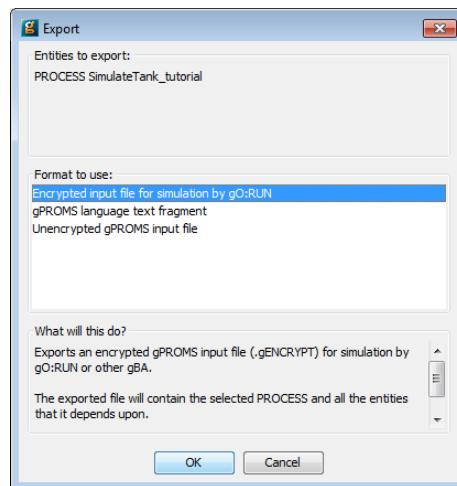


Figure 198: Select “Encrypted Input File”

3. On the second page, set the **Export directory** to a directory the user can find. Preferably one without any other files in it so the user will not be confused by the output. If the filename or the **Encryption password** are not changed, SimSinter will be able to guess the password. If either of those values are changed, the user will have to set the correct password in the SinterConfigGUI password setting. A Decryption password is probably unnecessary, as the user has the original file. SimSinter does not use it. When the user has finished setting up these fields, click **Export Entity**.

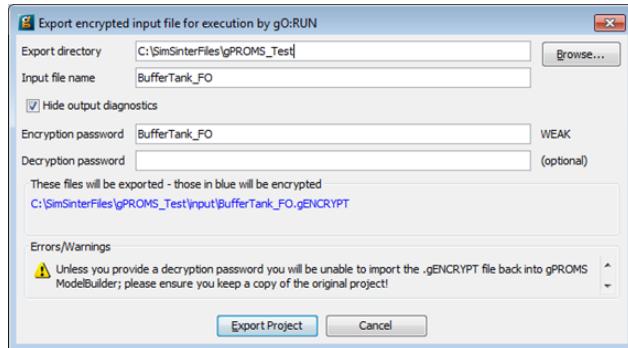


Figure 199: Export Entity Page

- The resulting .gENCRYPT file will be saved to a subdirectory named “Input” in the save directory specified in Step 3. The first part of the name will be identical to the .gPJ file-name. The user should not rename it because the SinterConfig file will guess this name, and currently changing it requires editing the SinterConfig file.

Configuring SimSinter to Work with gPROMS

Now that the gPROMS process has been prepared, the SimSinter configuration can be done.

- The “SinterConfigGUI” can be launched from FOCUS, via the Create/Edit button found in File→Add/Update Model to Turbine or “SinterConfigGUI” may be run on its own by selecting CCSI Tools → FOCUS → SinterConfigGUI from the Start menu.
- The splash window displays, as shown in Figure 200. The user may click the splash screen to proceed, or wait 10 seconds for it to close automatically.



Figure 200: SinterConfigGUI Splash Screen

3. The SinterConfigGUI Open Simulation window displays (Figure 201). If “SinterConfigGUI” was opened from FOCUS, the filename text box already contains the correct file. To proceed immediately click **Open File and Configure Variables** or click **Browse** to search for the file.

This tutorial will use the .gPJ file edited in Section 9.5.1. Remember that SinterConfigGUI cannot read the .gENCRYPT file that is actually run by SimSinter. Instead, the user must open the .gPJ file the ModelBuilder uses.

Once the file is selected, click **Open File and Configure Variables**.

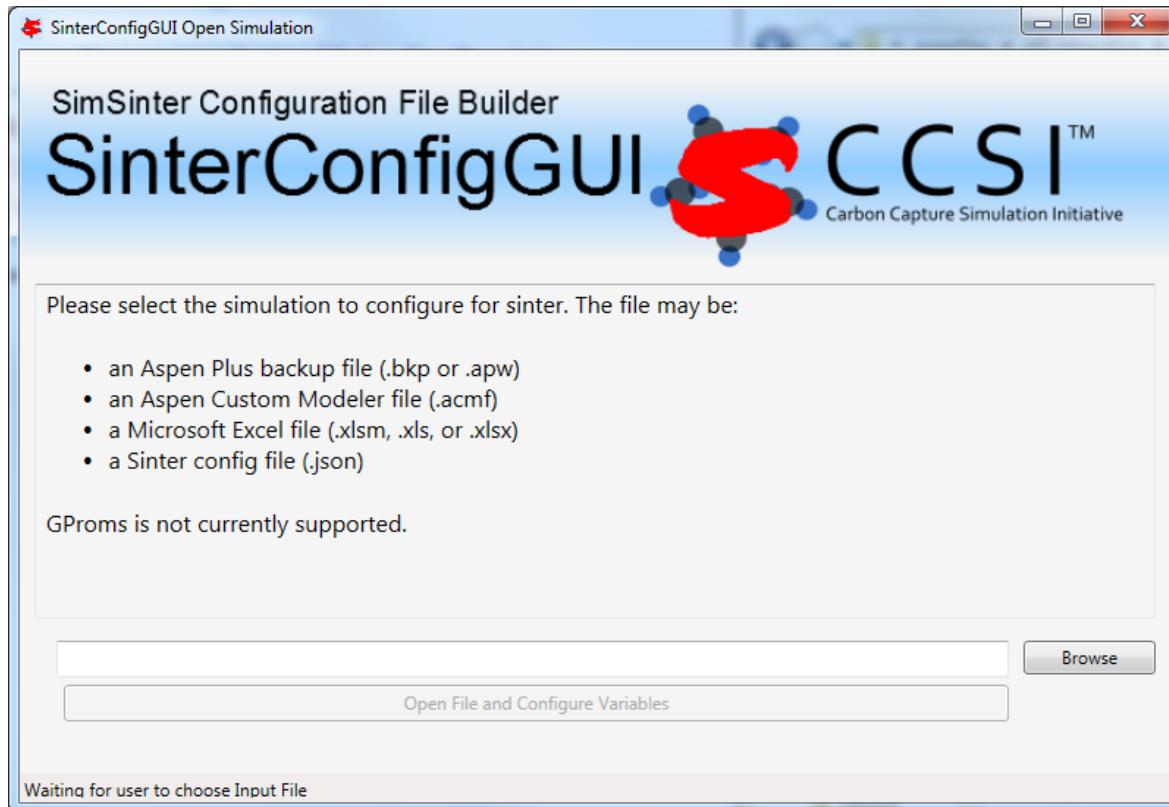


Figure 201: SinterConfigGUI Open Simulation Screen

4. The SinterConfigGUI Simulation Meta-Data window displays as shown in (Figure 202). Unlike the other simulations, gPROMS has not started up in any way. SinterConfigGUI does not get information from gPROMS directly, it must parse the .gPJ file instead.
5. The first and most important piece of meta-data is the **SimSinter Save Location** at the top of the window. This is where the Sinter configuration file is saved. The system suggests a file location and name. The user should confirm this is the intended location of the files to not accidentally overwrite other files. Enter the remaining fields to provide the meta-data to describe the simulation that was just opened and then click **Next**.

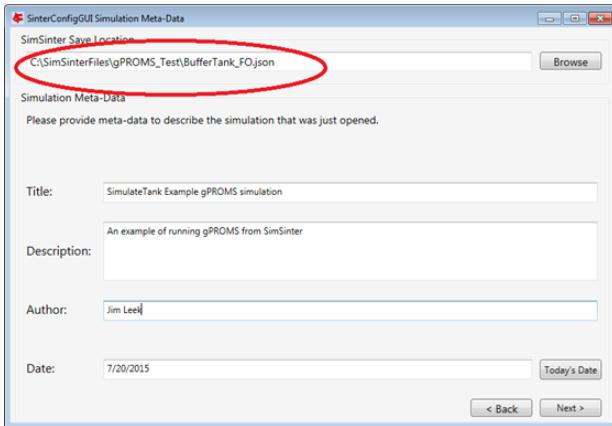


Figure 202: SinterConfigGUI Simulation Meta-Data Save Text Box

6. The SinterConfigGUI Variable Configuration Page window displays as shown in Figure 203. gPROMS has two settings, ProcessName and password. SimSinter has guessed at both the ProcessName and the password. For this example the password is correct, but the ProcessName is incorrect. SimulateTank is the process that isn't configured to work with SimSinter. On the left side we can see the Variable Tree. The root is connected to the three processes defined in this .gPJ file. First, change the ProcessName to “SimulateTank_tutorial”.

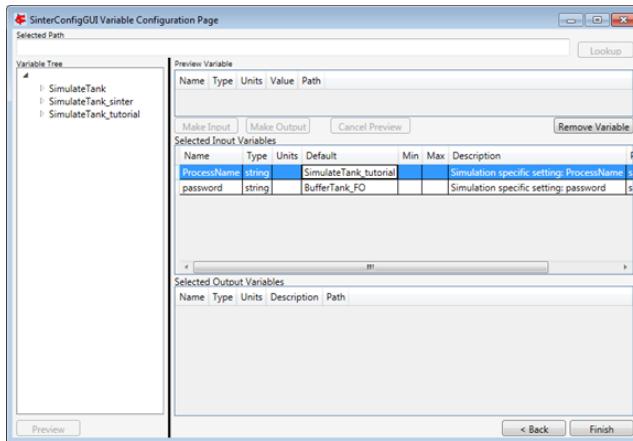


Figure 203: SinterConfigGUI gPROMS Settings Configuration

7. After changing the ProcessName, click Enter (or clicks away). The Selected Input Variables will automatically display all of the available input variables. This is because the input variables have been configured in gPROMS, and SimSinter has parsed them out of the .gPJ file, as long as you have the ProcessName set correctly. This also means that the user cannot add new input variables in SinterConfigGUI, only in gPROMS. SimSinter also does its best to identify the Default values, Min, and Max of the variables. The default can only be calculated from the file if it was defined purely in terms of actual numbers. SimSinter cannot evaluate other variables or functions. Therefore,

`DEFAULT 2 * 3.1415 * 12`

will work. However,

`DEFAULT 2 * PI * radius`

will not work, because SimSinter does not know the value of either PI or radius, and SimSinter will just set the default to 0.

Min and **Max** values are taken from the variable type, if there is one.

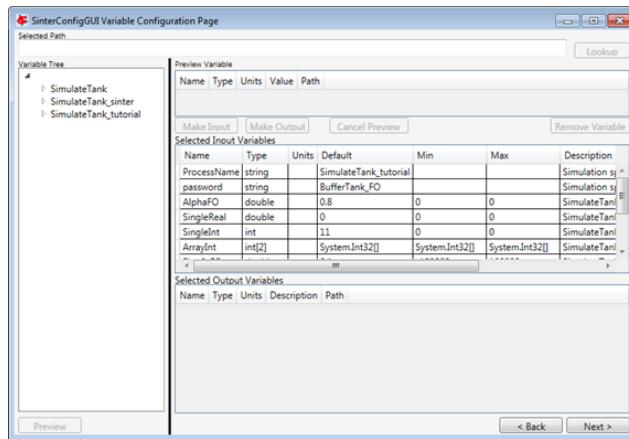


Figure 204: SinterConfigGUI Automatically Displays Input Variables

- Now the output values can be entered. Expand the “SimulateTank_tutorial” Process on the Variable Tree, expand the “T101” model, and then double-click on “FlowOut” to make it the Preview Variable. Notice that the **Make Input** button is disabled. As stated above, the user cannot make new Input Variables in SinterConfigGUI. Only **Make Output** is allowed.

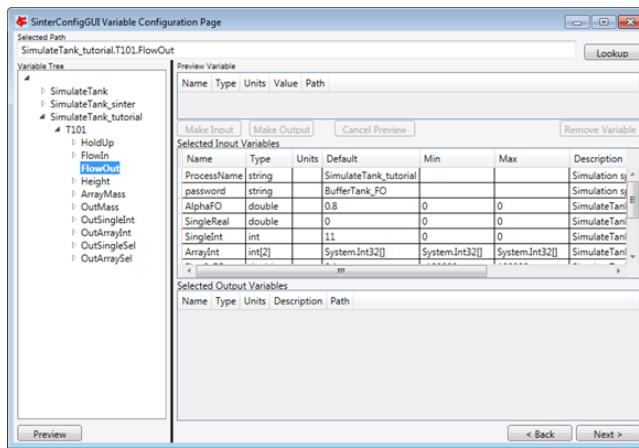


Figure 205: Preview of the FlowOut Variable

- If **Make Output** is clicked, “FlowOut” will be made an Output Variable as shown in Figure 206. The Description can be updated, but SimSinter made a good guess in this example; therefore, there is no need to change the description.

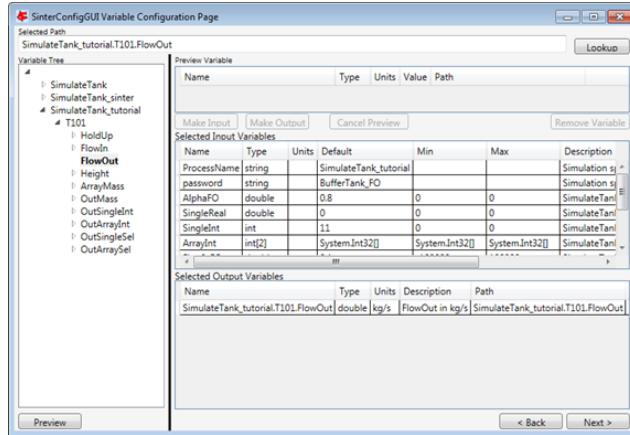


Figure 206: FlowOut as an Input Variable

10. By the same method, make Output Variables “HoldUp” and “Height.”

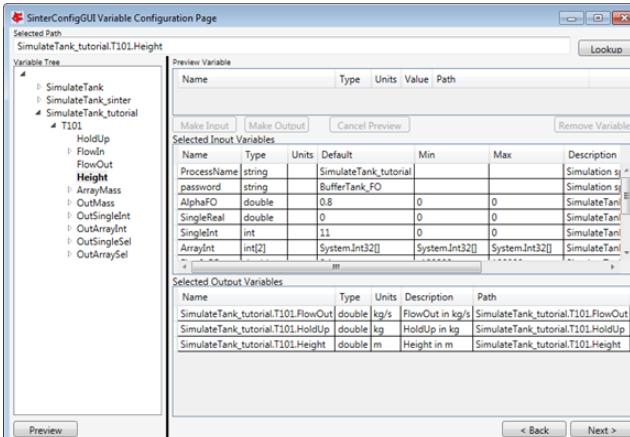
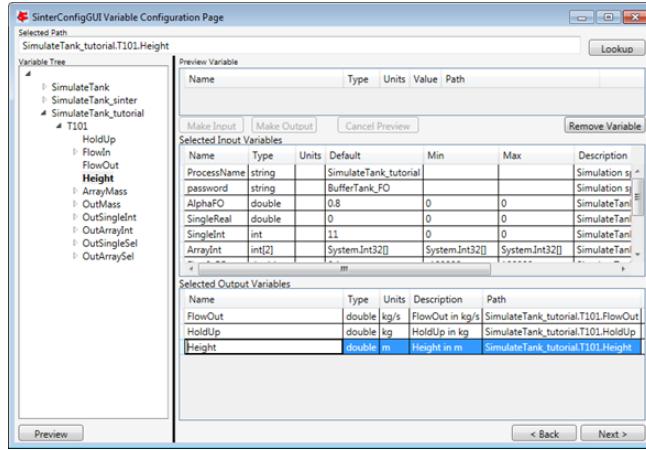


Figure 207: HoldUp and Height Output Variables

11. The variables names should be made shorter. Simply click on the Name column and change the name to your preferred name.

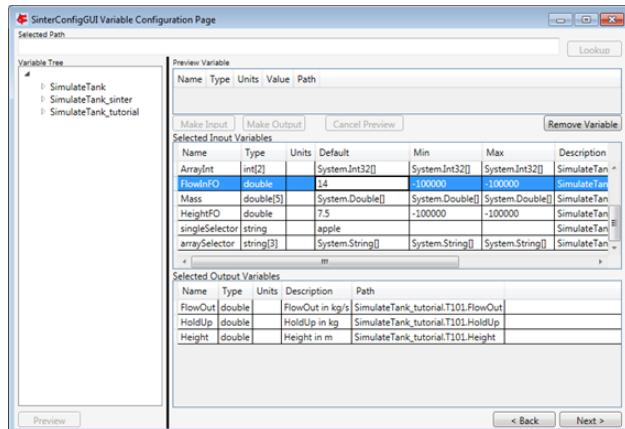
**Figure 208: Editing Variable Names**

12. For future testing, make sure the defaults are good values. The only three input variables that matter have the following defaults:

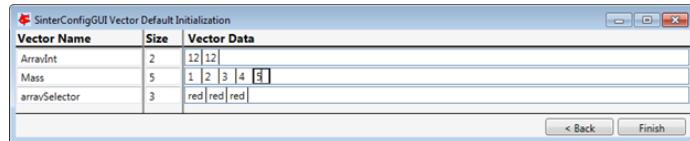
AlphaFO = 0.8

FlowInFO = 14

HeightFO = 7.5

**Figure 209: Editing Defaults**

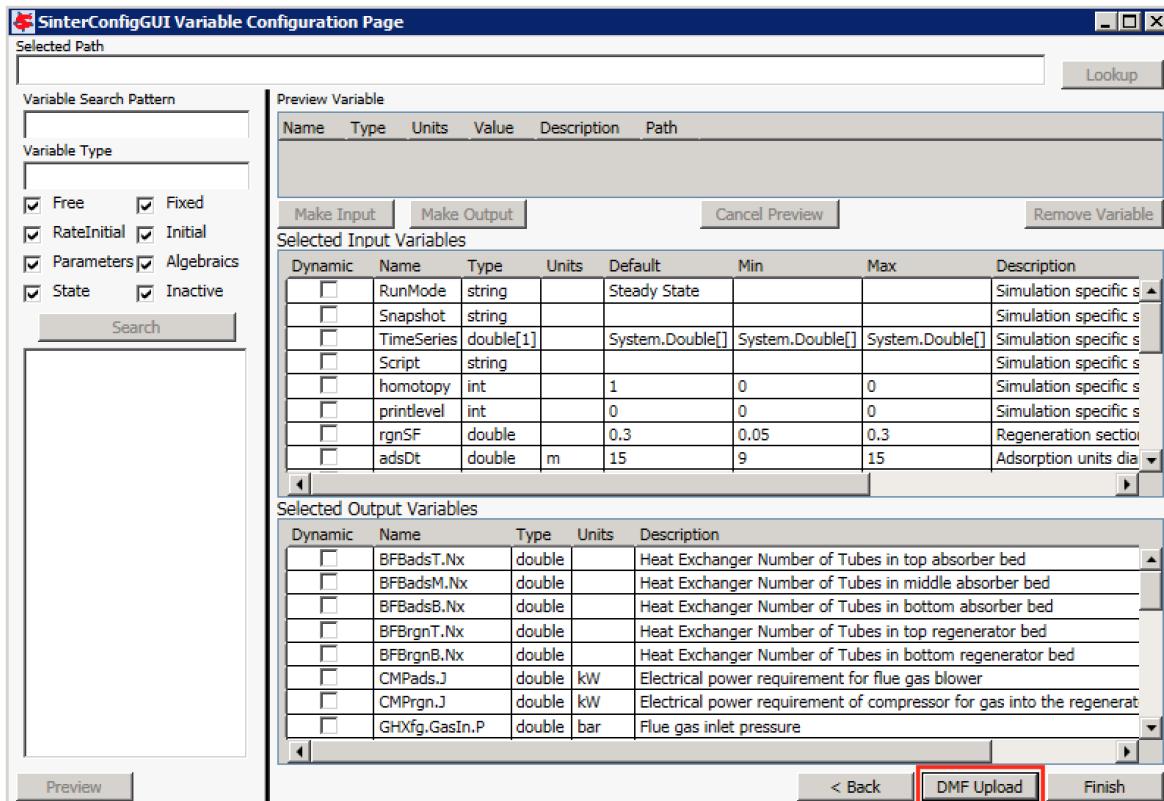
13. When finished making output variables, click **Next** at the bottom of the variables page. If there were any input vectors, the Vector Default Initialization page will display. Here the default values of the vectors may be edited.

**Figure 210: Editing Vectors**

14. Finally, click **Finish** and save your configuration file. Your gPROMS simulation should now be runnable from FOQUS.

DMF Integration

To provide a smoother workflow within the CCSI toolset, SimSinter is integrated with the Data Management Framework (DMF). When the DMF is setup, SimSinter provides the option to upload simulations directly to the DMF (see Figure 211). For more information on how to setup the DMF, please refer to section ??.

**Figure 211: DMF Upload button allows simulations to be uploaded directly to the Data Management Framework**

10. DEBUGGING

This chapter contains information that may be helpful in resolving a problem or filing a bug report.

How to Debug

Log files may contain very useful information when reporting problems. The log files are contained in the logs sub-directory of the FOCUS working directory. To change the log message levels in FOCUS go to the FOCUS Settings button from the Home window. From there various log settings can be changed. The debugging log level provides the highest level of information.

Almost any error that occurs in FOCUS should be logged. Occasionally, an error may occur that is difficult to find, or causes FOCUS to crash before logging it. In that case the “FOCUS Console” application can be used. All output from FOCUS, including messages that cannot be seen otherwise will be shown in a “cmd” window which will remain open even after FOCUS closes.

When running heat integration, the debugging information can be found in
\\gams\\HeatIntegration.lst. This file includes detailed results and errors returned by GAMS.

Most UQ routines interact with PSUADE via Python wrappers. When PSUADE is running, the stdout is written to psuadelog in the working directory. (At present, only some PSUADE commands write to this log; however, this will be standardized in the near future so that all PSUADE commands write to this log.) Other errors that are due to the Python wrappers or PySide GUI components are written to the logs subdirectory in the working directory.

Known Issues

The following are known unresolved issues:

- The FOCUS flowsheet can be edited while a flowsheet evaluation, optimization, or UQ is running. This should not be allowed, and may cause problems.
- With the windows installer, FOCUS may produce output to standard error, especially if it immediately fails to launch. Output is usually caught and redirected to the FOCUS log and displayed in dialog boxes within FOCUS, but rare instances may occur where error messages are not caught. Output to standard error is logged in the directory with foqus.exe in the file foqus.exe.log. The user does not typically have permission to write to the FOCUS install location, so an error message such as “Cannot write to foqus.exe.log” will be displayed. If this occurs there are two solutions (1) change the permissions for the FOCUS install directory or (2) run “FOCUS Console” application, which will direct FOCUS output to the “cmd” window.
- The win32com module generates Python code, which it needs to run. This code is generated in the FOCUS install location “\\dist\\win32com\\gen_py.” In some cases there may be a problem writing to that directory due to permission settings. This will prevent FOCUS from running simulations locally. If this error is encountered the solution is to make the “gen_py” directory user writable. So far, in testing, this error seems to occur in Windows 8 and 10, but

not 7.

- The user regression analysis features, iREVEAL and ALAMO, of the UQ tool requires a separate Python 2.7 installation. Furthermore, Python must be both in PATH variable and associated with .py files. Details on installing Python and fixing any issues encountered may be found in the iREVEAL Installation Guide and the iREVEAL User Manual, Known Issues section.
- FOQUUS has trouble getting files from Turbine and saving them to the DMF when dealing with files in Turbine involving directories.
- The default port for TurbineLite is 8080. If another program is already using port 8000, there will be an error in FOQUUS when connecting to TurbineLite. In the **Turbine** Tab of the Settings window, there is a tool to change the TurbineLite port. If the TurbineLite port is changed the configuration file that FOQUUS uses to connect to TurbineLite, must also be changed.

Reporting Issues

To report an issue, please send an email to:

ccsi-support@acceleratecarboncapture.org

Please include detailed steps on how to reproduce the error, including screenshots and log files.

11. References

- [1] C. Tong, “PSUADE User’s Manual, Version 1.2.0,” Tech. Rep. LLNL-SM-407882, Lawrence Livermore National Laboratory, Livermore, CA 94551-0808, May 2011.
- [2] A. Cozad, N. V. Sahinidis, and D. C. Miller, “Automatic Learning of Algebraic Models for Optimization,” *AIChe Journal*, vol. 60, pp. 2211–2227, 2014.
- [3] C. B. Storlie, H. D. Bondell, B. J. Reich, and H. H. Zhang, “Surface estimation, variable selection, and the nonparametric oracle property,” *Statistica Sinica*, vol. 21, no. 2, pp. 679–705, 2011.
- [4] C. B. Storlie, B. J. Reich, J. C. Helton, L. P. Swiler, and C. J. Sallaberry, “Analysis of computationally demanding models with continuous and categorical inputs,” *Reliability Engineering & System Safety*, vol. 113, pp. 30–41, 2013.
- [5] B. J. Reich, C. B. Storlie, and H. D. Bondell, “Variable selection in bayesian smoothing spline anova models: Application to deterministic computer codes,” *Technometrics*, vol. 51, no. 2, pp. 110–120, 2009.
- [6] J. H. Wegstein, “Accelerating Convergence of Iterative Processes,” *j-CACM*, vol. 1, no. 6, pp. 9–13, 1958.
- [7] N. Hansen, *Towards a New Evolutionary Computation. Advances in Estimation of Distribution Algorithms*, ch. The CMA Evolution Strategy: A Comparing Review, pp. 75–102. Springer, 2006.
- [8] S. G. Johnson, “The nlopt nonlinear-optimization package.” <http://ab-initio.mit.edu/nlopt>, May 2015.
- [9] E. Jones, T. Oliphant, P. Peterson, *et al.*, “Scipy: Open source scientific tools for python.” <http://www.scipy.org/>, May 2015.
- [10] K. Bhat, B. Sherman, K. Ajayi, B. Ng, J. Eslick, J. Ou, and J. Kress, “Solventfit: A calibration tool for solvent-based CO₂ capture models,” in *2015 CCSI Industry Advisory Board (IAB) Program Review Meeting*, (Reston, VA), September 2015.