Graphical User Interface for Simulink Integrated Performance Analysis Model

R. Caitlyn Durham

Marshall Space Flight Center

August 7, 2009

Reviewed by NASA Mentor
Don Krupp
EV82

_____
Signature Here

Graphical User Interface for Simulink Integrated Performance Analysis Model

R. Caitlyn Durham[1]
West Virginia University, Morgantown, WV, 26508

The J-2X Engine (built by Pratt & Whitney Rocketdyne,) in the Upper Stage of the Ares I Crew Launch Vehicle, will only start within a certain range of temperature and pressure for Liquid Hydrogen and Liquid Oxygen propellants. The purpose of the Simulink Integrated Performance Analysis Model is to verify that in all reasonable conditions the temperature and pressure of the propellants are within the required J-2X engine start boxes. In order to run the simulation, test variables must be entered at all reasonable values of parameters such as heat leak and mass flow rate. To make this testing process as efficient as possible in order to save the maximum amount of time and money, and to show that the J-2X engine will start when it is required to do so, a graphical user interface (GUI) was created to allow the input of values to be used as parameters in the Simulink Model, without opening or altering the contents of the model. The GUI must allow for test data to come from Microsoft Excel files, allow those values to be edited before testing, place those values into the Simulink Model, and get the output from the Simulink Model. The GUI was built using MATLAB, and will run the Simulink simulation when the "Simulate" option is activated. After running the simulation, the GUI will construct a new Microsoft Excel file, as well as a MATLAB matrix file, using the output values for each test of the simulation so that they may graphed and compared to other values.

I. Introduction

In order to return manned spacecraft to the moon, the National Aeronautics and Space Administration (NASA) has created the Constellation Program. The Constellation Program consists of two main launch vehicles, the Crew Launch Vehicle (CLV), Ares I, and the Cargo Launch Vehicle (CaLV), Ares V. Ares I has the primary purpose of taking the astronaut crew to orbit where it will rendezvous with the Altair Lunar Lander that was launched by Ares V. Then, the Orion Crew Exploration Vehicle (CEV) and the Altair Lunar Lander will travel to the moon.

The Ares I launch vehicle consists of a first stage, an upper stage, the Orion Capsule, and the Launch Abort System (LAS.) The Upper Stage has the liquid fueled J-2X engine, built by Pratt & Whitney Rocketdyne, as well as the fuel tanks, interstage and instrument unit.

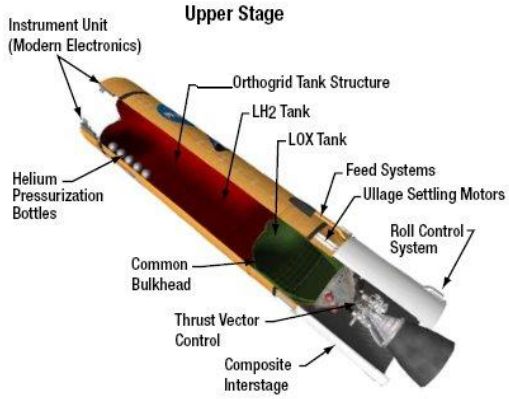


Figure 1. Ares I Upper Stage

In order for the J-2X engine to start properly, the temperature and pressure of the Liquid Hydrogen (LH2) and Liquid Oxygen (LOX) propellants must be within a certain "start box." The Simulink Integrated Performance Analysis Model simulates the conditions of the propellants using specific inputs. The graphical user interface (GUI) provides a way to test variable inputs to simulate different scenarios in an efficient and fairly simple way.

[1] Student Trainee, EV82 Stage Analysis, Marshall Space Flight Center, West Virginia University

II. GUI Version 1

The objectives of GUI Version 1 were that it would have a way to test parameters individually, get the initial (or default) values from a MATLAB (dot)mat file, and run the simulation using the values that are in the GUI's edit boxes.

A. Initial GUI Version 1

To create the GUI, a plan was written that listed all of the things that needed to be done in order to create a successful GUI program. Simple MATLAB mat-files and a Simulink model were created to test the GUI as it was built. Research was done on both Simulink and MATLAB GUIs to learn how to build these elements. Through a series of trial and error, a Simulink model of the quadratic equation, which was chosen because of its simplicity, was created. The GUI would then be used to input the values A, B, and C into the model.

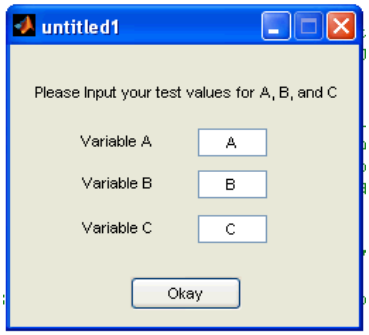


Figure 2. Initial GUI Version 1

MATLAB has built a program called Graphical User Interface Design Environment (GUIDE) that will assist in building GUIs. In GUIDE you can lay out the elements of the GUI such as text, buttons, and edit-text boxes. The initial GUI layout consisted of three edit-text boxes, their descriptions, a title, and an Okay button (see Figure 2, page 2.) While this dialog box worked well with the quadratic equation model, it was impractical when used with a model that had more than three variables and were not titled A, B, and C. The GUI needed to be able to read the names of the variables, instead of calling all variables A, B, and C, because A, B, and C may not be the names of the parameters in the user's model. It also needed to have more than three edit-boxes; the exact number of required edit boxes (for the number of parameters) was unknown.

B. Final GUI Version 1

In order to accommodate more parameters, variable names and their default values, and selectable Excel data and Simulink models, changes and additions were made to the initial GUI Version 1 (see Figure 3, page 2) which would become the final GUI Version 1(referred to as simply GUI Version 1.) It was assumed that GUI Version 1 (final) was the complete working GUI that had all of the required functions. Multiple trial runs of different pieces of code were done to determine how the GUI would act in certain situations.

In the end, the GUI was able to get a users Simulink model and Excel file using the MATLAB built in function "uigetfile." GUI Version 1 was able to take an Excel file, which contained names of parameters in the first column, and their values in the second column, and load its values and names into the GUI. It would then show the names of the variables and an edit box containing their value from the excel file. If the number of variables in the Excel file was less than 60 (the maximum possible number of parameters that the GUI would accept) it would hide all of the unused edit boxes. Once the user selects a model or enters a name, that model will open. If the user pushes the "Cancel" button, it will close the Simulink Model, and the GUI.

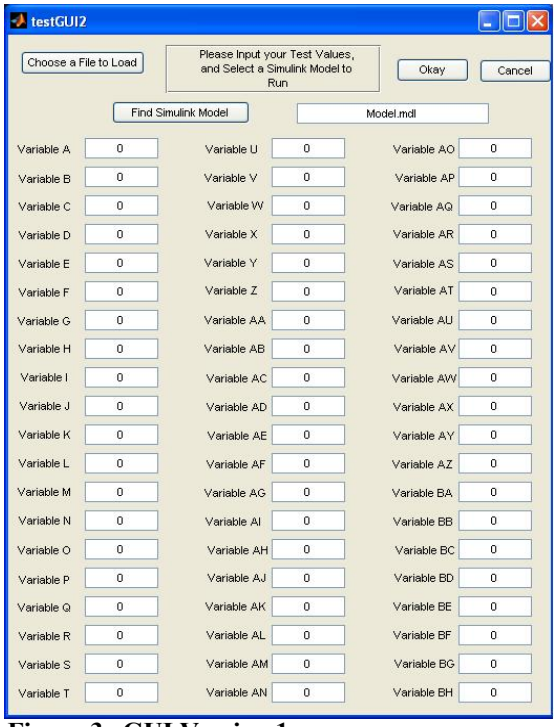


Figure 3. GUI Version 1

Before the user selects an Excel file or a Simulink model, GUI Version 1 shows 60 edit text boxes and their names. The names of the variables go from A to BH, and their initial value is zero. The names A – BH are just for the code, and do not relate to the actual data or to the model. The edit box beside the "Find Simulink Model" button lists the model that the user has selected. Because it is an edit text box, the user can enter the name of their model manually. Before the user selects a model to run, the GUI displays "Model.mdl" in the edit text box as a default and as an example on how to format the name of your model if the user decides to input it manually.

When the user pushes the "Okay" button, it will run the simulation using the values that are in the visible edit boxes. Once the simulation is complete, it will close the GUI dialog box but not the model. The output of the simulation will follow the model. If the output comes to the workspace as a variable, you can then type the name of the output variable in the command window to see its values.

While this GUI works, there was confusion concerning all of the required functions of the GUI. This GUI does not allow the user to have more than 60 variables, it does not load any tables, it does not do anything with the Simulink output, and it cannot test multiple values of one parameter at the same time.

III. GUI Version 2

In order to make the GUI requirements clear, many functions of the GUI were specifically stated in the Task Description (Appendix A, page 10.) The most important requirements were that the GUI must be able to load a minimum of 100 parameters that are required to run the simulation and a minimum of 20 tables, allow the user to edit values that it loaded from a Microsoft Excel spreadsheet, allow the user to test a certain parameter at multiple values, run the simulation, and save the outputs from the simulation in a new file.

Building the GUI started with a Dialog Box similar to the Select Model and Simulate dialog box in the current GUI (see section III.C, Select a Model and Simulate.) Initially the user would choose whether they wanted to edit values in the table or test multiple values of the parameter. Multiple individual functions (not built by MATLAB GUIDE) were written in the beginning stages of GUI Version 2 to do certain tasks that the GUI would need to perform so that the GUI code would be as simplified as possible. Examples of these functions include functions to determine the number of rows in the Excel file, to determine the number of tests that need to be done based on parameter minimum and maximum values, and to load the tables listed in the Excel file.

In GUI Version 2 dialog boxes the Okay, Cancel, and Help buttons all generally have the same purpose throughout every dialog box. By pushing the Okay button, the user is stating that they are finished with that dialog box and are ready to move on. The cancel button will, in most cases, move the user back one dialog box. The help button is meant to give the user assistance with using the GUI by opening the GUI tutorial in the MATLAB help browser.

Each dialog box inside of the main GUI has a specific purpose to run the GUI while giving the user as much control as possible. Each dialog box is its own entity, and each of those individual dialog boxes are connected by using one dialog box to run the next. While this GUI has a fairly high value of usability, the user will need to work from top to bottom on each GUI, and read most of the descriptions in order to ensure that they understand the GUI's requirements.

The GUI does not do everything for the user. The user must also provide the Excel file in the correct format, and prepare the model. The parameter values in the model must be manually set by the user to the variable name that is provided in the fourth column of the Excel file.

A. Welcome Dialog Box

The GUI will not output the correct values if the user does not use the correct format for the Microsoft Excel file that they will load the values from. The most important parts of the Excel file are the parameter names in the first column, the values in the second column, the variable names in the fourth column, the table names in the seventh column, and the actual mat-file names of the tables in column eight. The tables do not correspond to the parameters in their specific row.

To warn the user that the GUI may not function properly without the correct Excel file format, the "Welcome Dialog Box" was created which briefly explains the purpose of the GUI, describes the correct Excel file format, and warns the user that the GUI will delete the variables in the MATLAB workspace and in the command window (see Figure 4, page 3.)

The welcome dialog box does not have any functions aside from informing the user, thus its code is fairly simple. As with all GUI code that is created by MATLAB, there is code that is written by GUIDE that is used to create, open, and run the GUI, but does not directly affect the way that the GUI works. The only code that was added closed the Welcome dialog box and opened the "Get Excel Data" dialog box (see Figure 5, page 4.) The name of the m-file for the Welcome dialog box is simply gui.m so that the

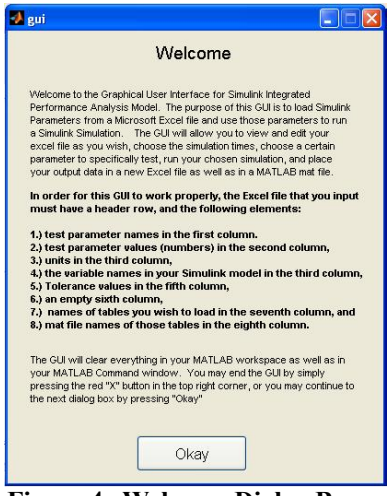


Figure 4. Welcome Dialog Box

user can use the GUI without having to remember a long descriptive name.

B. Get Excel Data Dialog Box

The GUI must be able to read and load a minimum of 100 parameters and 20 tables from a Microsoft Excel file. Before the GUI will run any of its main code, it will clear the variables in the MATLAB workspace and clear the MATLAB command window. While this feature is not entirely necessary (or required as per the task description,) it was added to simplify the process of running the GUI and to decrease the GUI's running time. If the variables were not erased from the workspace they could interfere with the GUI, especially if they have the same name as any of the variables used in the GUI. This feature of the GUI can be removed without any major consequences.

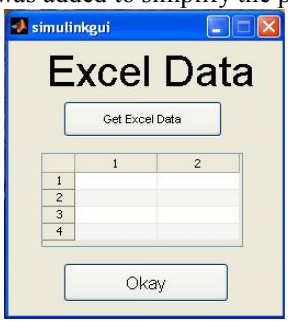


Figure 5. Empty Get Excel Data dialog box

To accomplish this GUI's main objective (load Excel data,) the built-in MATLAB function "uigetfile" is used to prompt the user to select the Excel file name. Uigetfile was used to get the Excel file as well as the Simulink model. It allows the user to browse through their drives (C:, D:, etc.) to find the file that they wish to use (see Figure 7, page 5.) In the case of the Get Excel Data dialog box, when the user pushes the "Get Excel Data" button it will allow them to find and select only Excel files (those files having an .xls or .xlsx extension.) This code was built into the callback (the function that runs when the button is pushed) for the Get Excel Data button. For aesthetics, the Get Excel Data dialog box begins small (see Figure 5, page 4,) but once the user has loaded data, the Get Excel Data dialog box will resize to show the data. If the data is larger than the built in size for the table, there will be scroll bars on both the bottom and right side. (see Figure 6, page 4.)

In addition to getting the Excel file, the Get Excel Data callback function takes the data that comes from the Excel file and manipulates it so that the names of the parameters, the variables, the values, and the table data all have their own array (or list) containing that data. This is necessary so that each data array (such as parameter names or parameter values) may be used later in the code to perform other operations. This code counts the number of rows in the Excel file so that the GUI knows how many parameters it is dealing with.

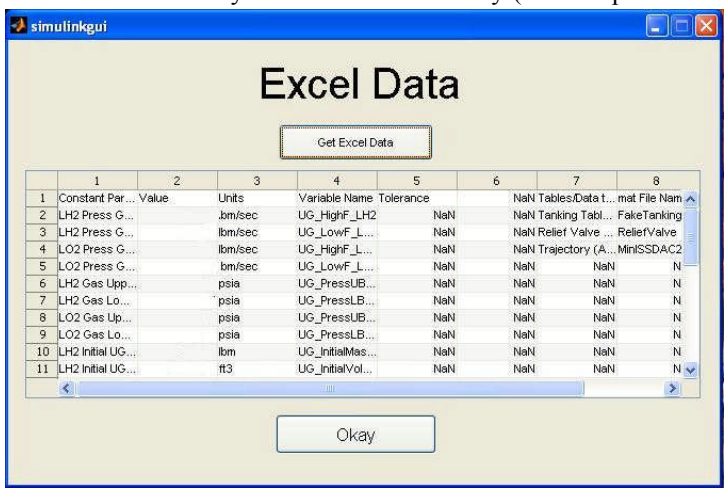


Figure 6. Full Get Excel Data dialog

The most important feature of the Get Excel Data m-file (and GUI) is that it sets the variable names in the Excel file equal to their respective values and places them in the MATLAB workspace so that they may be used to run the simulation. The simulation will not run if the variables are not set to their values.

Once the user has selected an Excel file, the data from the spreadsheet will load into the table in the Get Excel Data dialog box. Once the data is in the table, the user is free to edit as many values as they wish. When the numeric values are changed, their values will immediately change in the MATLAB workspace. If the user changes variable names, or other text values, their names will not be changed, but a new variable will be created which is equal to the value in column two (thus you will have two variables equal to the same number.)

When the user is happy with the values in the table, they can push the "Okay" button to move to the next dialog box in the sequence. Once the user pushes Okay, the values in the table at that time are the values that will be used to run the simulation. The Okay button will also close this dialog box and open the "Select a Model and Simulate" dialog box.

C. Select a Model and Simulate Dialog Box

The user has several choices for how they want to run the simulation: they can choose to run the simulation once with the values from the table, they can choose to run the simulation multiple times testing a range of values for a certain parameter, and in both case they can choose whether or not to input the simulation time data (start time, stop

time, and time step.) The GUI allows the user to decide which method to use in the "Select a Model and Simulate" dialog box (see Figure 8, page 5.)

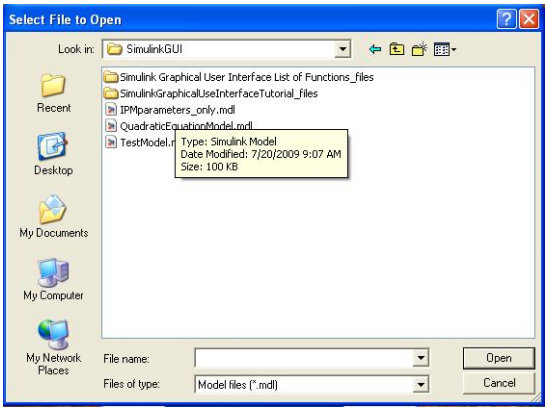


Figure 7. Uigetfile dialog box

At the top of the dialog box, there is a button that will allow the user to select the model using the MATLAB built in function uigetfile as was used to get the Excel file (see Figure 7, page 5.) Once the user has chosen a model, the model name will be shown next to the button so that the user may ensure that they have chosen the correct model. The dialog box also shows the name of the chosen Excel file. If the user pushes the Cancel button the GUI will go back to the Get Excel Data dialog box, but the user will have to reload any data that was in the table.

This dialog box has two optional buttons that will open more dialog boxes if the user pushes them, however, this can be the last dialog box before the simulation is run if the user chooses not to push either of those buttons.

If the user chooses to run the simulation only once, they will need to input a name for the output file. This can be done in the "Enter Save Name" edit box. If the user enters a save name and later chooses to test one variable at multiple values (via the "Select and Test Only One Parameter" button,) the GUI will take the save name that was entered in the "Test Parameter" dialog box (see Figure 10, page 6.)

If the user pushes the "Set Configuration Parameters" button, the "Simulink Configuration Parameters" dialog box will open while not closing the Select a Model and Simulate dialog box (see Figure 9, page 5.)

If the user pushes the "Select and Test Only One Parameter" button, the GUI will close the Select a Model and Simulate dialog box and open the Test Parameter dialog box (see Figure 10, page 6.)

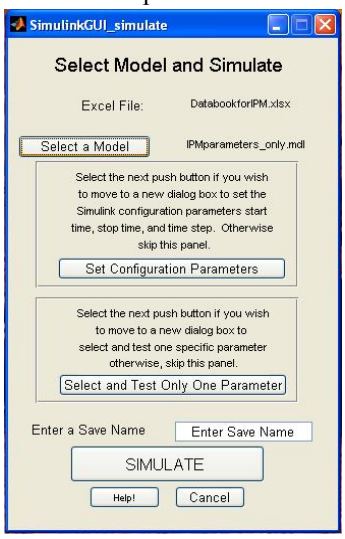


Figure 8. Select a Model and Simulate dialog box

D. Simulink Configuration Parameters Dialog Box

In a Simulink model, the simulation runs on simulation time. The default time data in Simulink starts at zero seconds and ends at 10 seconds (in simulation time.) When a new model is created, the user can enter the end time for the simulation as well as the time step (meaning you step forward in one second intervals, half second intervals, etc. based on what the user wants.) The GUI has a feature that will allow the user to enter a simulation start time, stop time, and time step each time the run the GUI.

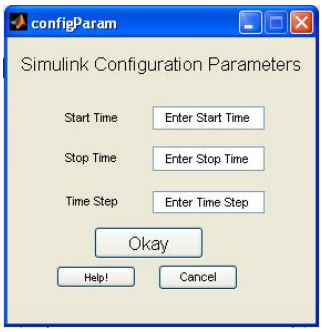


Figure 9. Simulink Configuration Parameters dialog box

When the user pushes the "Set Configuration Parameters" button in the Select a Model and Simulate dialog box, it will temporarily open the "Simulink Configuration Parameters" dialog box (see Figure 9, page 5.) Pushing this button will not close the Select a Model and Simulate dialog box. When the user pushes the Okay button, the Simulink Configuration Parameters dialog box will close.

The user may run into some issues when using this option, because the time step must be manually set in the model. The GUI outputs the variable name "timeStep" to the MATLAB workspace which will then be used in the model only if the user has manually set the step size to "timeStep" in the Configuration Parameters area, which can be found in the Simulation pane at the top of the Simulink Model. If the user has not set the time step as the "timeStep" variable, the model will simply accept the value that is in the model, and not the value that the user has entered. If the user does not enter a value for the time step, the GUI has a built in time step of 1 second.

This dialog box has some built in code to protect the GUI from any mistakes that the user may have made. One of those mistakes is opening the Simulink Configuration Parameters dialog box without meaning to enter the configuration parameters. In the early stages of this individual dialog box, the Select a Model and Simulate code told the main GUI when the user has chosen to open the Simulink Configuration Parameters dialog box. The GUI

would then determine how to run the simulation (with or without time values) based on whether the dialog box had opened or not. The issue here is that the user may open the dialog box and realize that they didn't want to. Running the GUI would result in a MATLAB error stating that there is no value for the start time etc. Additional code was added to this dialog box to confirm that the user has entered values in the boxes before continuing. If the user has not, the GUI is told to run the simulation without the time values. The GUI does not, at this time, confirm if the user has entered a valid real number into the box. This will probably be an easy improvement for the GUI (see Section III.G "Future Improvements.")

E. Test Parameter Dialog Box

The GUI has the ability to run the simulation multiple times using a series of values for one specific parameter. This ability gives the user an easy way to test a certain parameter and compare each of the outputs from the Simulink model. The data that comes from the simulation output is saved to a new Microsoft Excel file as well as the MATLAB native mat file. The user can do whatever they need with that data to observe the effects of the changing parameter value. This ability is included in the "Test Parameter" dialog box (see Figure 10, Page 6)

Like the Select a Model and Simulate dialog box (see Figure 8, Page 5,) the "Test Parameter" dialog box shows the user the Excel file and Simulink model that they have chosen.

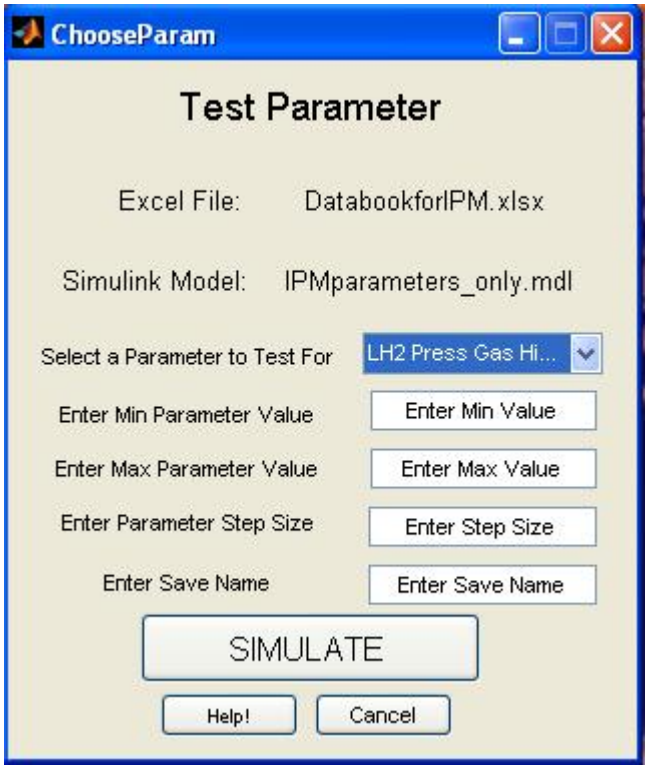


Figure 10. Test Parameter dialog box

The user will choose a parameter to test from the drop down menu called "Select a Parameter the Test for." The menu gets its data from one of the arrays that are created in the "Get Excel" dialog box (see Section III.B for more information,) specifically the parameter names array. Currently, if the user wants to select the first parameter in the list, in this case LH2 Press Gas High Flow, as opposed to simply leaving the pop-up-menu the way it is (showing that parameter,) they must drop down the menu and select the first parameter. This is an issue that will hopefully be changed in the near future.

To test one specific parameter at a range of values, the GUI must know what the user wants that range to be. In order to calculate the range, the GUI asks the user to input the minimum value of that range, the maximum value of that range, and the step value. An array is created which holds each of the test values so that they may be pulled from the array and used for each simulation. For example, if the user entered 4 in the minimum value box, 6 in the maximum value box, and 0.5 in the step box, the array of test values would be [4 4.5 5 5.5 6].

Before running the simulation, the user must enter a name in the Enter Save Name box. This name will be used as the file name for the Excel output file. If the user does not enter a name the program will "error Terminate" or stop running because of some internal coding error. This can be avoided by simple entering a name in the box. This error may also occur in the save name box in the Select a Model and Simulate dialog box (see Section III.C.) This error will be fixed in a future improvement without an exceptional amount of effort.

When the user pushes the "Simulate" button, the program will first check that the user has entered real numbers for the minimum value, maximum value, and step value for the test parameter. If the user has, the simulation will first create a matrix titled "answer" that will be used in the code to hold the simulation output. The first column and the second row of the answer matrix will hold the name "Time" as a header column. The run simulation command is held within a couple of nested for loops which create the answer matrix based on the simulation output. A loop is a programming statement that will run a certain bit of code multiple times based on either a counter or a condition. The GUI loops through each answer matrix column, and places the values from the simulation output in them. Once the GUI has made it through all of the loops in the code, it places the output matrix, answer, into a new Excel file and into a MATLAB matrix file.

If the user has not entered a real number, the GUI will display a warning dialog box that will tell the user "Input values may only be numeric, and must be real numbers." The user may then change the value until they stop receiving this warning message. This feature is very important in this GUI as well as in all of programming, because it protects the program from error termination caused by the user. More of this ability will be added in future improvements to the GUI.

F. Simulink Outputs

When the user runs the simulation, answer matrix is created which is then used to create a new Microsoft Excel spreadsheet. The spreadsheet has labels above most of the columns. These labels tell the user the name of the parameter that is being tested and the values that parameter is tested at above the outputs. An example of an output file is shown in Figure 12, page 7. This output may be used to graph the data and compare each of the iterations of the test parameter.

Figure 11 (page 7) is a sample graph that shows the output from the model that tested a parameter at a range of values. The user can observe the graph, check for any major issues, and confirm if this parameter will affect the temperature and pressure that must be within their respective start boxes. This graph is not created by the GUI automatically; the user must manipulate the Excel data to create the graph. In the future, an ability to graph the data automatically may be added.

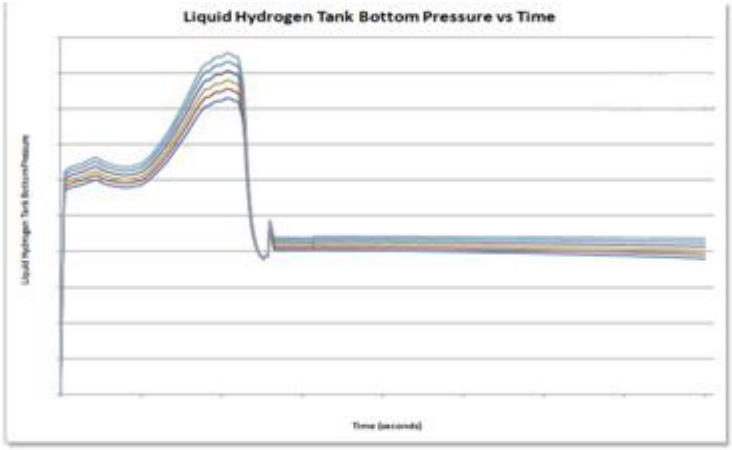
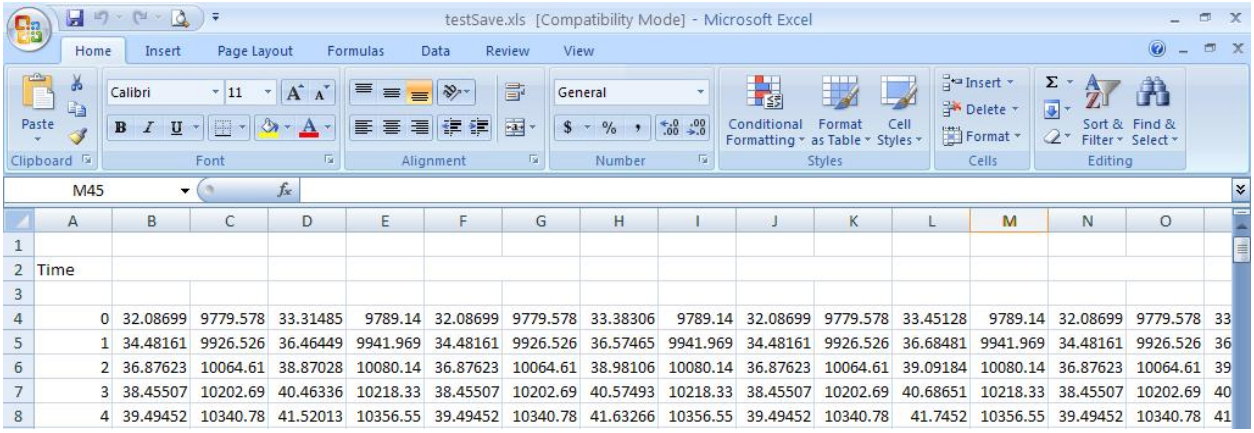


Figure 11. Liquid Hydrogen Tank Bottom Pressure graph

In most cases, each test iteration will have multiple outputs, all of which may be graphed separately. The output used for this individual graph was the Liquid Hydrogen Tank Bottom Pressure. A plot feature may be added in the future improvements that will graph data automatically.

G. Future Improvements



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	
1																
2	Time															
3																
4	0	32.08699	9779.578	33.31485	9789.14	32.08699	9779.578	33.38306	9789.14	32.08699	9779.578	33.45128	9789.14	32.08699	9779.578	33
5	1	34.48161	9926.526	36.46449	9941.969	34.48161	9926.526	36.57465	9941.969	34.48161	9926.526	36.68481	9941.969	34.48161	9926.526	36
6	2	36.87623	10064.61	38.87028	10080.14	36.87623	10064.61	38.98106	10080.14	36.87623	10064.61	39.09184	10080.14	36.87623	10064.61	39
7	3	38.45507	10202.69	40.46336	10218.33	38.45507	10202.69	40.57493	10218.33	38.45507	10202.69	40.68651	10218.33	38.45507	10202.69	40
8	4	39.49452	10340.78	41.52013	10356.55	39.49452	10340.78	41.63266	10356.55	39.49452	10340.78	41.7452	10356.55	39.49452	10340.78	41

Figure 12. Excel Output file

This GUI has many features that may be possible in the future, or are being implemented currently. The most important improvement is increasing the GUI's usability, or user friendliness. There a few times during the GUI's process when the user could inadvertently cause the GUI to error terminate. While this isn't a major issue, the user would have to restart the GUI. Adding more warning boxes and loops that would not allow the GUI to run until the user has corrected a problem that may cause GUI termination.

At this time, the pop-up-menu in the Test Parameter dialog box (see Section III.E) does not allow the user to leave the first parameter as the test parameter without manually clicking the parameter name. This issue may be changed by adding a "Select Parameter" value at the top of the pop-up-menu.

Next, the Simulink Configuration Parameters dialog box (see Section III.D) does not check that the user has entered real numbers for the start time, stop time, and time step in seconds. This could be a problem because of the GUI's callback functions that return the values in the edit boxes.

A feature that was wanted from the beginning of the GUI building process was the GUI's ability to plot the output data that it receives from the simulation. The user would not have to graph the data manually when this feature is implemented. The best thing would be for the GUI to make the graph in Excel, but a stepping stone to that may be to plot the data in a new MATLAB dialog box. This feature has some comments included in the GUI code which may be used to implement the plot feature in the beginning.

The output into Excel currently shows the parameter being tested as a column header and the values of that parameter above the output values. In the near future, an ability to read the names of the output data from Simulink and place those names above their data columns is wanted. An additional feature may also be added which will merge the cells that are headers for more than one column.

The ability to test values of a parameter within a certain tolerance will probably need to be included in later versions to increase the efficiency. When this feature is added, the GUI will automatically test values of a parameter within the tolerance.

H. Tutorial

The final requirement of this GUI was that it must have a help tutorial for users. The user can view this tutorial by opening the actual Microsoft Word document, or by pressing the help button on any of the dialog boxes. The tutorial explains how to run the GUI, what is required to run the GUI, as well as issues that are known at this time. In addition, the tutorial also has descriptions of each individual dialog box, a list of MATLAB functions used in the GUI, and a few frequently asked questions. The tutorial may be found in its entirety in Appendix B, page 11.

IV. Conclusion

This document contains details about the functions and abilities of the GUI, as well as some descriptions of code and methods used to create and run the GUI. This GUI will greatly decrease the time and effort spent to run the Simulink Integrated Performance Analysis Model. This GUI is not restricted to this Simulink model; it may be used on any Simulink model, as long as the user has followed the same pattern for the Excel file and the model.

Appendices

A. Task Description

B. GUI Tutorial

C. Acronym List

D. Simulink Integrated Performance Analysis Model

C. Acronym List

CaLV	Cargo Launch Vehicle
CEV	Crew Exploration Vehicle
CLV	Crew Launch Vehicle
GUI	Graphical User Interface
GUIDE	Graphical User Interface Design Environment
LAS	Launch Abort System
LH2	Liquid Hydrogen
LOX	Liquid Oxygen
MSFC	Marshall Space Flight Center
NASA	National Aeronautics and Space Administration

D. Simulink Integrated Performance Analysis Model

Acknowledgments

R. Caitlyn Durham thanks Don Krupp, MSFC EV82, for continuing support in all aspects of the project throughout the internship, Vicky Garcia, MSFC EV82, for continuing support in all aspects of the project throughout the internship, Mike Hannan, MSFC EV41, for coding support on GUI Version 1 and the initial stages of GUI Version 2, Peter Ma, MSFC EV82, for coding support on GUI Version 1, Fred Flack, MSFC EV82, for export control support, and all of the MSFC EV82 Branch for support throughout the internship.

References

[1] *The MathWorks*. The MathWorks, Inc., 1994. Web. June-July 2009. <http://mathworks.com>.

[2] MathWorks, The. *Learning MatLab 7- MathLab & Simulink, Student Version*. 14th ed. The MathWorks, 2005. Print.

[3] MathWorks, The. *Simulink: Model-Based and System-Based Design*. 5th ed. The MathWorks Inc., 2002. Print.

[4] MathWorks, The. *Simulink: Dynamic System Simulation for MATLAB*. 4th ed. The MathWorks Inc., 2000. Print.