Slycat™ User Manual

Patricia Crossno, Jaxon Gittinger, Warren Hunt, Matt Letter, Shawn Martin, Milosz Sielicki

Overview

Slycat™ is a web-based system for performing data analysis and visualization of potentially large quantities of remote, high-dimensional data. Slycat™ specializes in working with ensemble data. An ensemble is a group of related data sets, which typically consists of a set of simulation runs exploring the same problem space. An ensemble can be thought of as a set of samples within a multi-variate domain, where each sample is a vector whose value defines a point in high-dimensional space. To understand and describe the underlying problem being modeled in the simulations, ensemble analysis looks for shared behaviors and common features across the group of runs. Additionally, ensemble analysis tries to quantify differences found in any members that deviate from the rest of the group.

The Slycat™ system integrates data management, scalable analysis, and visualization. Results are viewed remotely on a user's desktop via commodity web clients using a multi-tiered hierarchy of computation and data storage, as shown in Figure 1. Our goal is to operate on data as close to the source as possible, thereby reducing time and storage costs associated with data movement. Consequently, we are working to develop parallel analysis capabilities that operate on High Performance Computing (HPC) platforms, to explore approaches for reducing data size, and to implement strategies for staging computation across the Slycat™ hierarchy.

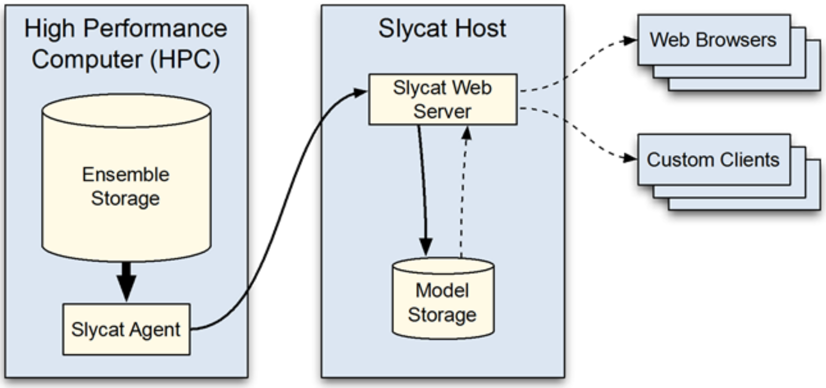


Figure 1: Slycat multi-tiered hierarchy, designed for large data analysis and exploration with minimal data movement.

Within Slycat™, data and visual analysis are organized around *projects*, which are shared by a project team. Project members are explicitly added, each with a designated set of permissions. Although users sign-in to access Slycat™, individual accounts are not maintained. Instead, authentication is used to determine project access. Within projects, Slycat™ models capture analysis results and enable data exploration through various visual representations. Although for scientists each simulation run is a model of real-world phenomena given certain conditions, we use the term model to refer to our modeling of the ensemble data, not the physics. Different model types often provide complementary perspectives on data features when analyzing the same data set. Each model visualizes data at several levels of abstraction, allowing the user to range from viewing the ensemble holistically to accessing numeric parameter values for a single run. Bookmarks provide a mechanism for sharing results, enabling interesting model states to be labeled and saved.

Getting Started

Slycat™ is accessed through a web browser from your desktop computer. Slycat™ currently supports Firefox, Chrome, and Safari browsers. We do not support Internet Explorer.

Since multiple Slycat™ servers are already in existence, you will need to obtain the URL for the Slycat™ server that you want to use. Enter this URL into the address bar of the browser. If the authentication mechanism for your institution relies on username and password, you will be taken to the Slycat™ login page, shown in Figure 2, where you will be prompted for your username and password. If your institution uses single sign-on, login will happen automatically and you will skip this step. Once your identity has been established, you will find yourself on the main *Projects* page.

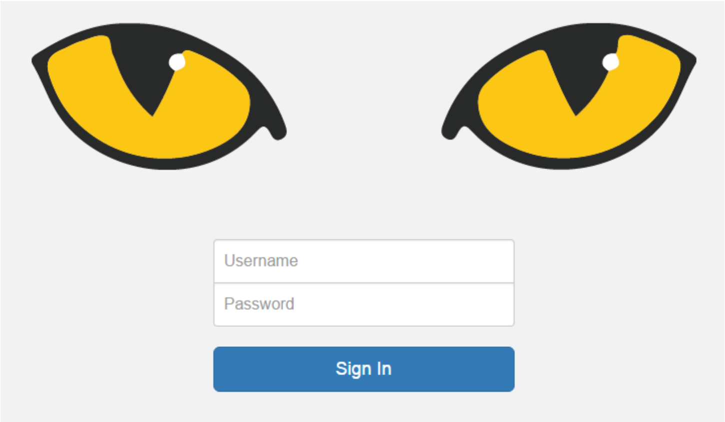


Figure 2: Slycat login page.

Slycat™ pages exist at one of three levels: the main Projects page, an individual project page, and an individual model page. The main Projects page displays all projects which you are authorized to access. This list of projects is unique to you. Clicking on a project name will take you to that project page, which will contain a list of all models that have been generated within the project. Clicking on a model name will take you to that model page, which will display a visualization of its data. At any level, clicking on the Slycat™ logo will return you to the main Projects page.

The first time that you access the Slycat™ website, your projects list will probably be empty, unless someone else has already created a project and added you as a project member. Since models cannot exist outside of a project, you must first create a project (see Project Creation below) before you can create a model. Project-specific information consists of the project name, a list of project members, a set of models, and a set of saved bookmarks for models within that project.

Slycat™ Navbar

At the top of every Slycat™ page is the Navbar. Working from left to right, we see the Slycat™ logo, a breadcrumb navigation path, and a set of colored buttons providing dropdown lists categorized by function. Depending on the type of page currently being viewed, the buttons and the contents of the dropdowns will vary. As shown in Figure 3, the Navbar for the main Projects page, the only button available is the green *Create* button for creating new projects. Since the main Projects page lies outside and above any projects, the breadcrumb navigation path points to the current page, which is simply labeled as *Slycat*. Figure 4 shows that for Navbars within a project or model page, there can be up to five buttons, including: *Create*, *Edit*, *Info*, *Bookmarks*, and *Delete.*

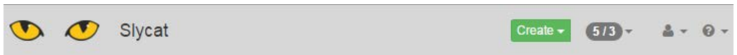


Figure 3: Slycat Navbar as seen on the main Projects page. At this level, the Navbar displays the title *Slycat* because we have yet to move to an individual project page.

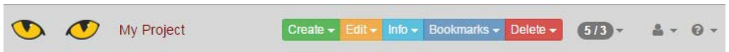
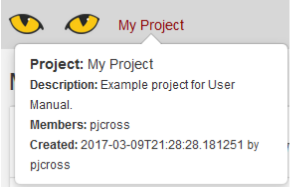


Figure 4: Navbar at the individual project page level. Here the name of the project is 'My Project'. Note that the *Bookmarks* button is hidden until at least one bookmark has been created.

As you move between pages at various levels, the breadcrumb path in the Navbar will change to reflect your current location. The path has the format *Project Name / Model Name*. The path can be used to navigate within the hierarchy. Clicking on the *Project Name* will take you to that project's page with its list of associated models. Hovering over the *Project Name* will display the project description, the project members, the date of creation, and who created it (Figure 5). Similarly, hovering over the *Model Name* will display the model description, the date of creation, and who created it.



Project: My Project
Description: Example project for User Manual.
Members: pjcross
Created: 2017-03-09T21:28:28.181251 by pjcross

Figure 5: Hovering over the project name will display more detailed project information.

Projects

Project-specific information consists of the project name, a list of project members, a set of models, and a set of saved bookmarks for models within that project.

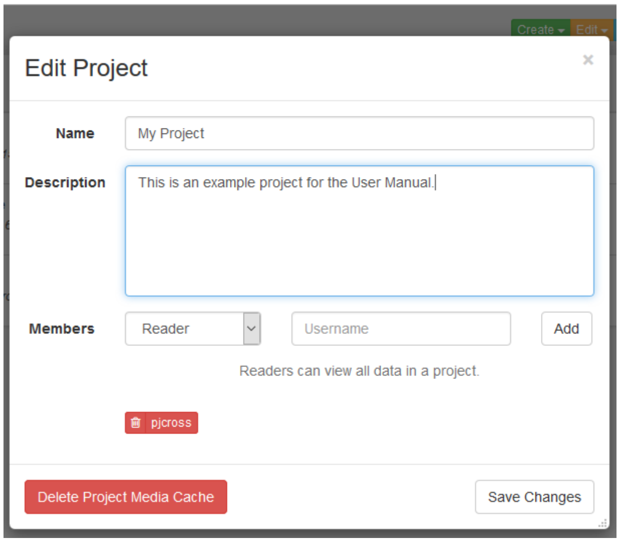
Project Creation

Projects are created by clicking the green *Create* button on the main Projects page and selecting *New Project* from the dropdown that appears. A dialog will pop up, providing editable regions for you to enter a *Name* and an optional *Description*. This brief text field allows you to provide more detailed comments or notes beyond the project name. Clicking the *Finish* button in the lower right corner creates the project and takes you to the newly created project page; clicking the *Cancel* button in the lower left corner aborts project creation.

Once the project is created, you will find yourself on the empty project page. As the project creator, you are automatically assigned the role of *Administrator* for that project (although there can be multiple project members with Administrator roles). A series of buttons appears to the right of the project name. Since you have an Administrator role, there will initially be four buttons: *Create*, *Edit*, *Info*, and *Delete*. Only project administrators can edit or delete the project. Otherwise, there will only be two buttons: *Create* and *Info*.

Editing Projects

Clicking on the yellow *Edit* button on a project page and selecting *Edit Project* from the dropdown list provides a means to change the project *Name*, *Description*, or project *Members*. An *Edit Project* popup will appear with the current project information (Figure 6). In a newly created project, the membership list consists solely of the project creator assigned the role of *Administrator*. The username of the creator will be shown within a red button (buttons are color-coded according to role and red is used for an *Administrator*) at the bottom.



Edit Project

Name: My Project

Description: This is an example project for the User Manual.

Members: Reader

Readers can view all data in a project.

Figure 6: *Edit Project* dialog allows you to change the project name, add or change a description, and add, remove, or change the roles of project members.

There are three different roles that project members can have: *Reader*, *Writer*, and *Administrator,* whose buttons are color-coded blue, yellow, and red, respectively. *Readers* can view all data in a project, but they cannot create new models, modify existing models, or delete models. *Writers* can both view and modify the contents of a project, but they are unable to add new project members or edit the project name or description. *Administrators* have full access to all aspects of the project, including adding new project members or deleting the project itself.

To add a project member, select a role from the dropdown list to the right of *Members* and type in the person's username. Note that the username is checked against a list of legitimate usernames and will be rejected if it is not found. If the username is found, a popup will translate the username into the person's full name and verify both the identity and the role selected. Click *OK* if both the person and role are as you intended, or *Cancel* if they are not. Now an additional button, color-coded by role and enclosing the newly added member's username, will appear in the member list below. Although the new member now appears to be in the project member list, this action has not been saved and will be discarded unless the *Save Changes* button is pressed.

To remove project members, click on the trashcan icon next to the name of the member to be removed. To change the role of a project member, add them as you would a new project member (you do not need to remove them first), but with the revised role. Note that as an Administrator, you have the power to delete yourself or reduce the level of your role (thereby losing your Administrator privileges), which is why we require you to first click the *Save Changes* button before we finalize any changes. If you find that you have accidently made a change that you do not want to execute, pressing the **X** button in the upper right corner of the *Edit Project* dialog cancels the edit and keeps the previous project state (*Name, Description, Members, and member roles*) intact.

Project Info

To see a non-editable version of a project's information, click on the cyan *Info* button on the project page and select *Project Details* from the dropdown. A popup will display the *Name*, *Description*, and project *Members* list. Click *Close* when you are finished viewing it. Note that this same information can be seen by hovering over the project name in the breadcrumb navigation path.

Deleting Projects

To remove a project, including ALL ITS MODELS AND DATA, click the red *Delete* button from within the project page of the project that you wish to delete. Select *Delete Project* from the dropdown. Note that only members with Administrator rights may delete the project. Project deletion is an irreversible operation, so deletion requires confirmation through a popup that asks if you really want to delete that project and all models within it. Press the red *Delete Project* button to confirm deletion, or the **X** button in the upper right corner of the dialog to cancel the operation and keep the project.

Models

In Slycat™, models combine analysis and visualization. Slycat™ provides three different types of models: Canonical Correlation Analysis (CCA), Parameter Space, and Time Series. The heart of every model is a data table. For each model type, there are predefined sets of linked views that provide different representations of the analysis results. Generally, the visualization for each model consists of three different representations, each showing the ensemble at a different level of abstraction. The highest-level view seeks to display the ensemble in a holistic manner. It seeks to show what high-level behaviors or trends can be seen across most, if not all, of the simulation runs. Slycat™ currently provides views showing correlations between inputs and outputs, or similarities between results. The intermediate-level view presents individually distinguishable runs in the context of the group, showing how well each member aligns with the high-level view of shared ensemble traits. The low-level view enables you to drill down to the raw data values, both to input parameters and to the results from individual runs.

Each model has a *Name*, a *Marking*, and an optional *Description*. Marking choices are defined as part of the server configuration, so they are specific to the institution that hosts the server. The intent is for

Slycat™ to facilitate clear labeling of data sensitivity through explicit choice of marking. The marking appears as part of the model description on the Project page list, plus it is shown in both header and footer bars when visualizing the model.

Creating Models

Models are created by clicking the green *Create* button from within a project page and selecting one of the model types from the dropdown list. The information needed to create a model varies depending on which model you choose, so a popup dialog specific to the selected model will step you through entering the necessary information for that type (the details for each are covered below). Model creation can be aborted at any stage by clicking the *X* button in the upper right corner of the popup.

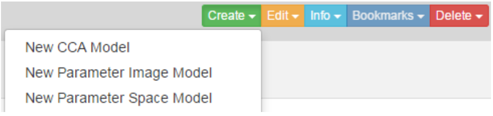


Figure 7: *Create* dropdown list of model choices, as seen from a project page.

Editing Models

Clicking on the yellow *Edit* button on a model page and selecting *Name Model* from the dropdown list allows you to change the model *Name*, *Description*, and *Marking*. A *Name Model* dialog will popup with the current model information. Click *Save Changes* to modify the model description on the server, or click the *X* button in the upper right of the popup to abort the operation.

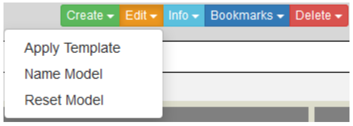


Figure 8: *Edit* dropdown list, as seen from a model page.

Reset Model

As you interact with a model and change various aspects of the visualization, Slycat™ keeps track of the current model state. If you leave that model and return to it later, Slycat™ will resume with the model rendered according to the most recent configuration state. Note that this is only true if you are returning to a model on the same computer using the same browser that you previously used to view it. However, sometimes you might want to start over with the default settings to produce the initial visualization. Clicking on the yellow *Edit* button on a model page and selecting *Reset Model* from dropdown list will return the model state to its initial configuration.

Deleting Models

To remove a model and ALL ITS DATA, click the red *Delete* button from within the model page of the model that you wish to delete. Select *Delete Model* from the dropdown. Model deletion is an irreversible operation, so deletion requires confirmation through a popup asking if you really want to delete that model. Press the red *Delete Model* button to confirm deletion, or click the *X* button in the upper right of the popup to abort the operation and keep the model.

View Regions

The inner part of each model’s visualization is subdivided into several views or regions, each separated from adjacent regions by a thick gray line. As you move the mouse over one of these region dividers, a double-headed arrow cursor perpendicular to the divider replaces the normal arrow cursor, the line extent (all but a darker gray center section) highlights in yellow, and the tooltip *Resize* pops up. If you click and drag the divider while this is enabled, the divider will move until you release it, resizing the regions on either side to reflect proportional changes created by the new divider location. The divider can be dragged to the very edge, effectively hiding the view.

Alternately, if you move the mouse over the darker center section of a divider, the center section highlights in yellow, the icon become a hand with a pointing finger, and the tooltip *Close* appears. Clicking the mouse button now will collapse one of the two adjacent regions. It collapses the region that is closer to the edge of the browser window. Clicking a second time on that same divider (now positioned along the edge of the model visualization) will restore the previous layout.

Download Data Table

Since data tables are at the core of each model type, all models provide a table download operation. The download can take one of several forms: download the entire table, download only selected items, or download only visible items. As will be described later (see **Selecting Points and Filter**), selection and filtering can be used to divide the data into sets using two approaches, either through highlighting or through visibility. Highlighting and visibility are independently defined sets, so selected items are not necessarily visible.

This functionality can be used to download a table or a table subset to your desktop, which can then be used to generate a new model. For example, if you had an ensemble where some of the runs failed to terminate properly, you could filter those runs out and download the subset of runs that finished correctly. Then you could use that subset to generate new models where the failed runs are not biasing the analysis results. Or alternately, you could download the subset that failed and use that table to create a Parameter Space model to explore what the failures have in common.

Color Themes

Color is used extensively in Slycat™ to encode information of various types. In the table views that appear within each model, green columns are associated with input variables, lavender designates output variables, and unspecified variables are not colored (they are rendered using an off-white).

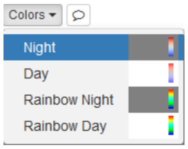


Figure 9: Dropdown list of color theme choices from the *Colors* button.

Slycat™ provides a set of predefined color themes, which are individually assigned to each model. A color theme consists of a bundled scatterplot background color and color palette for mapping numeric values to color-coded objects in Slycat™ views. Below the Navbar on the model page, there is an additional row of model-specific buttons. To change the current color theme, click the *Colors* button. As shown in Figure 9, there are four color themes available in the dropdown: *Night*, *Day*, *Rainbow Night*, and *Rainbow Day*. *Night* is the default choice. *Night* has a gray background and uses a diverging palette that maps low values to blue and high values to red, transitioning through white for values in the middle of the range [7]. *Day* has a white background and a similar blue to red mapping, though the palette is slightly shifted to transition through gray instead of white to enable you to distinguish points in the middle of the range from the background. *Rainbow Night* has a gray background and a conventional rainbow palette. *Rainbow Day* has a white background and a conventional rainbow palette. Although we provide *Rainbow* themes, we discourage their use since color order in the middle of the range is not intuitive.

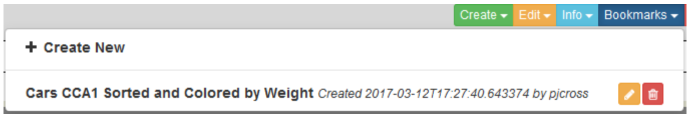
Bookmarks

Each time you interact with a model, changes in model state are preserved in a *Bookmark* and the URL in your browser's address bar is modified to incorporate the latest bookmark id. The id links to a description of the model state that is stored on the Slycat™ server. Although the contents of a bookmark are model dependent, all bookmarks capture the current visualization state so that it can be reproduced (though parameters such as view region sizes are not saved, since they are device dependent). Examples of the types of information stored in a bookmark include color-encoding, highlighted selections, filter values and limits, pinned media selections, and hidden points.

Bookmarking enables many useful functions. Dragging and dropping the URL from the address bar into an email, you can share a specific state of the visualization with other project members. If you save model pages as browser bookmarks, you can archive and recall interesting model states, though you will be limited to viewing them on the machine where you created them. The current bookmark id is stored locally in your browser's cache. This enables you to pick up where you left off when you begin a new session with a previously viewed model.

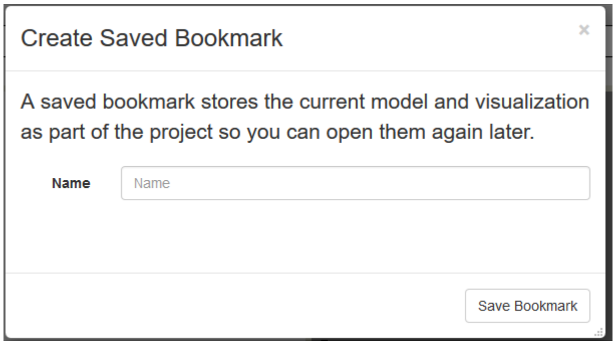
Within Slycat™ there is the concept of a saved *Bookmark*. This *Bookmark* is a persistent link to a model state that you explicitly save within a project. Slycat™ saves the current bookmark id along with a label that you provide. This provides a convenient, machine-independent mechanism for saving exploratory results. *Bookmarks* can be used to remember visualizations that reveal interesting patterns, to share findings with other team members, or to create a flipbook-style narrative for a demonstration.

To create a *Bookmark*, click the blue *Bookmarks* button from within a model page and select *Create New* (Figure 10). A *Create Saved Bookmark* popup will appear (Figure 11). Type in a *Name* and click the *Save Bookmark* button on the right to save it, or click on the *X* button in the upper right corner of the dialog to abort the operation. The *Bookmarks* button dropdown will display a list of all the bookmarks associated with the project. If you are on a model page, *Bookmarks* associated with that model are listed at the top, while those for other models appear below, each labeled with their model type. Clicking on a *Bookmark* in the list takes you to the associated model and visualizes it according to the saved state. To modify the name of a bookmark, click on the yellow pencil icon. To delete a bookmark, click on its red trashcan.



Cars CCA1 Sorted and Colored by Weight *Created 2017-03-12T17:27:40.643374 by pjcross*

Figure 10: Bookmarks dropdown, including one previously saved bookmark.



Create Saved Bookmark

A saved bookmark stores the current model and visualization as part of the project so you can open them again later.

Name: Name

Figure 11: Example of a *Create Saved Bookmark* dialog.

Note that changes to the *View Regions* are not currently preserved in bookmarks. Consequently, when the layout has been modified prior to the model being bookmarked, visualizing the *Bookmark* will render the model using the default *View Regions* layout.

Templates

Templates are essentially *bookmarks*, but they lack an associated model. *Templates* provide a means for applying the same visualization state to another model of that same model type. Note that the similarity between the template's original data and the new data set will determine the similarity of the resulting visualization. For dissimilar data, some portions of the saved state may not be applicable (in which case those attributes are ignored), or the results may significantly differ from your expectations.

To create a *Template*, click the green *Create* button, then select *Template* from the dropdown list. A *Create Template* popup will appear. Type in a *Name* and click the *Save Template* button on the right to save it, or click the *X* button to abort the operation.

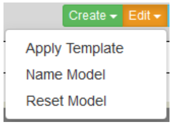
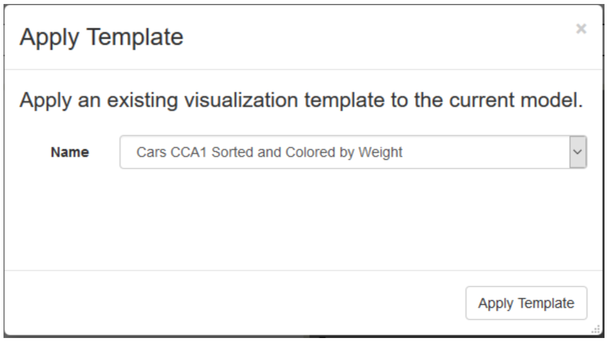


Figure 12: *Edit* dropdown for models, providing *Apply Template* functionality.

To apply a *Template* to a model, from within that model's page click the yellow *Edit* button and select *Apply Template* from the dropdown list (Figure 12). The *Apply Template* popup will appear. Select the name of the template that you wish to use from the list and click the *Apply Template* button on the right to execute, or *Cancel* to abort the operation. Note that bookmarks can also be used as templates, so they are included in the list of available templates (Figure 13). However, they are not interchangeable. *Templates* will not appear in the *Bookmarks* dropdown list, since they cannot be rendered without an associated model.



Apply Template

Apply an existing visualization template to the current model.

Name: Cars CCA1 Sorted and Colored by Weight

Figure 13: Apply Template dialog. Note that a bookmark appears in the template list.

Canonical Correlation Analysis Model

Canonical Correlation Analysis (CCA) was first proposed by Hotelling in 1936 [5]. Because CCA finds correlations between two multivariate data sets, CCA data structures are a good fit for exploring relationships between the input and output variables found in ensemble data sets (such as those generated for sensitivity studies, uncertainty quantification, model tuning, or parameter studies). Slycat™ uses CCA to model the many-to-many relationships between multiple input parameters and multiple output metrics. CCA is a linear method and a direct generalization of several standard statistical techniques, including Principal Component Analysis (PCA), Multiple Linear Regression (MLR), and Partial Least Squares (PLS) [1][3].

CCA operates on a table of scalar data, where each column is a single input or output variable across all runs, and each row consists of the values for each of the variables in a single simulation. Slycat™ requires the number of rows (samples) to be greater than the minimum variable count of the inputs or the outputs. A more meaningful result will be obtained if the ratio of runs to variables is ten or more. Additionally, columns cannot contain the same value for all runs. Slycat™ will reject such columns from being included in the CCA analysis, since they contribute no differentiating information. CCA cannot handle rows with missing data, Inf, -Inf, NAN, or NULL values. Slycat™ will remove rows from the analysis if any of the values in either the input or output variable sets include such data. However, if the bad values are only in columns that are not analysis variables, the row will be used.

For a concise description of CCA, we need the following definitions. Given n samples (n rows in the table), the input variables (presumed to be independent) will be referred to as the set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and the output (dependent) variables as the set $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$. Each vector $\mathbf{x}_i$ has p_1 components and each vector $\mathbf{y}_j$ has p_2 components. CCA attempts to find projections **a** and **b** such that $R^2 = \text{corr}\,(\mathbf{a}^T X, \mathbf{b}^T Y)$ is maximized, where corr (•,•) denotes the standard Pearson correlation.

The vectors $\mathbf{a}^T X$ and $\mathbf{b}^T Y$ are known as the first pair of canonical variables. Further pairs of canonical variables are orthogonal and ordered by decreasing importance. In addition to the canonical variables, the R^2 value for each variable pair is obtained, and various statistics can be computed to determine the significance of the correlation. A common statistic used in this context is the p-value associated with Wilks' λ [6]. Slycat™ provides both R^2 and p-values for each canonical component as part of the

Correlation View (see Figure 23 below). Note that these statistics assume that the data is normally distributed. If your data does not follow a normal distribution, be aware that these statistics will be suspect and adjust your interpretation of the results accordingly.

Once the canonical variables are determined, they can be used to understand how the variables in *X* are related to the variables in *Y*, although this should be done with some caution. The components of the vectors **a** and **b** can be used to determine the relative importance of the corresponding variables in *X* and *Y*. These components are known as canonical coefficients. However, the canonical coefficients are considered difficult to interpret and may hide certain redundancies in the data. For this reason, Slycat™ visualizes the canonical loadings, also known as the structure coefficients. The structure coefficients are generally preferred over the canonical coefficients because they are more closely related to the original variables. The structure coefficients are given by the correlations between the canonical variables and the original variables (e.g. corr ($\mathbf{a}^T X$, *X*) and corr ($\mathbf{a}^T Y$, *Y*)). These are calculated using Pearson's correlation between each column of *X* or *Y* and the corresponding canonical variable.

Cars Example Data Set

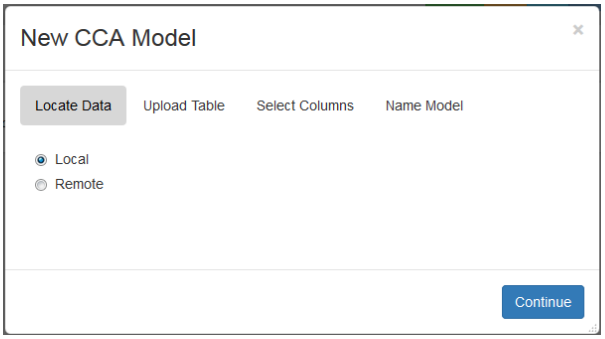
In the following sections, we will use the *cars* data set to illustrate model creation and CCA in general. *Cars* is not an ensemble of simulation data. Instead, it is a list of features for approximately 400 automobiles built between 1970 and the early 1980's. Selecting attributes which describe a car's physical system and labeling them as inputs, while grouping the performance-based variables as outputs, we can see the relationships between design choices and various performance metrics. Since CCA can only evaluate correlations between numeric variables, the analysis omits two columns, *Model* and *Origin*, which are string and categorical variables, respectively. Also note that *Acceleration* is a variable measuring the time required to reach a specific speed, so lower values represent greater acceleration.

This data set provides an intuitive introduction to CCA because most people already have some idea of how a car's manufacturing and performance features are related. Increasing weight, displacement, and number of cylinders all represent larger engines, which are in turn correlated with greater horsepower, lower miles per gallon (MPG), and faster acceleration. Due to the Arab oil embargos during the model years in this data set, engine sizes decreased over time to facilitate increased MPG.

Creating a CCA Model

Slycat™ accepts two file formats for table data, either Comma Separated Value (CSV) files, or Dakota tabular files (generated by Dakota [1], software which is frequently used to generate ensemble data sets). If your data is not currently in one of these two formats, Excel can be used to create CSV files from most common table formats. Note that if output metrics have been created separately in a post-processing step, they will need to be integrated with the inputs into a single file prior to model creation. In a CSV file, we expect to see only a single row of header information consisting of the column names.

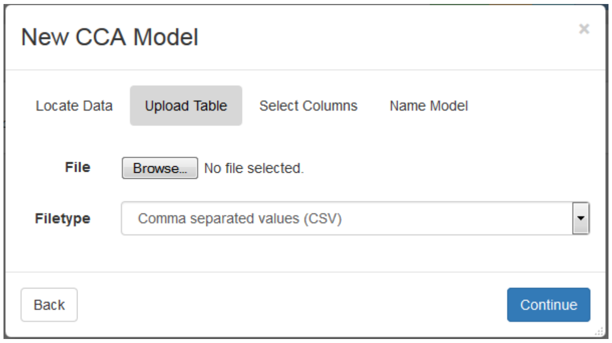
From your project page, click on the green *Create* button and select *New CCA Model* from the dropdown list. A dialog for walking you through the process will then pop up, as shown in Figure 14. The first page of the model creation wizard identifies whether the table is located on the local machine or whether the data is held on a remote machine. Select *Local* or *Remote,* followed by *Continue* to advance to the next page of the wizard.



● Local
○ Remote

Figure 14: Popup dialog in the CCA model creation wizard.

Local Files

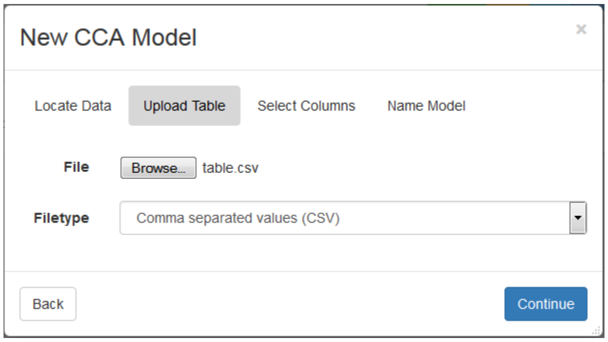


File: No file selected.

Filetype: Comma separated values (CSV)

Figure 15: Local file upload dialog in CCA model creation wizard.

As shown in Figure 15, if you selected *Local*, the next page will display two fields, *File* and *Filetype*. Adjacent to *File*, is the button *Browse*. Clicking *Browse* brings up a local file browser, which you can use to navigate to the location of your data table. After selecting a file, the file browser closes and the name of your selected file appears to the right of the *Browse* button, as shown in Figure 16. Depending on the format of the selected file, select either *CSV* or *Dakota tabular* from the *Filetype* dropdown, followed by *Continue* to read the file. Note, you can change your mind and read the table from a *Remote* host by clicking the *Back* button to return to the previous page.



File: table.csv

Filetype: Comma separated values (CSV)

Figure 16: Selected file, table.csv, shown in CCA model creation dialog.

Remote Files

As shown in Figure 17, if you select *Remote*, the *Choose Host* page enables you to log into a remote machine through Slycat™. First select a machine from the dropdown list, which is revealed by clicking the triangle to the right of *Hostname*. If the machine you wish to access is not on the list, type the machine name into the field. The name will be remembered and used as the default host for the next time. *Username* defaults to the username that you provided when logging into Slycat™, but this field can be manually edited if desired. Finally, enter your *Password* and click the *Continue* button in the lower right to connect to the remote host.

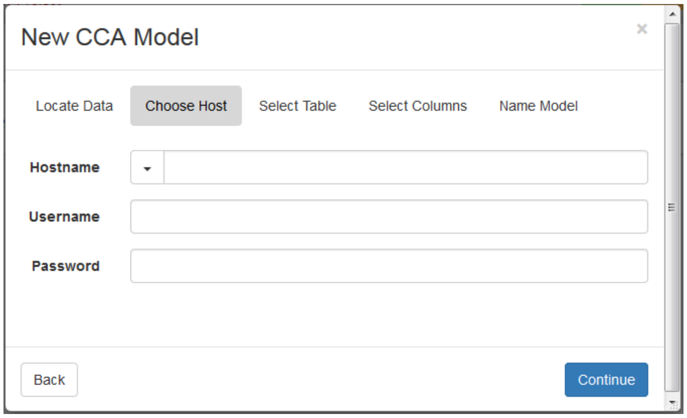


Figure 17: Remote system login for table ingestion in CCA model creation wizard.

Once you are connected, the model creation wizard will display a remote file browser. If you have previously accessed this machine through Slycat™, the browser directory will be initialized to your last location. Otherwise, the browser default directory will be the machine's root directory. There are two methods for navigating the remote directory structure to find your data: (1) if you know the full directory path, type it directly into the field at the top of the page (shown in Figure 18) and click the *Go* button;

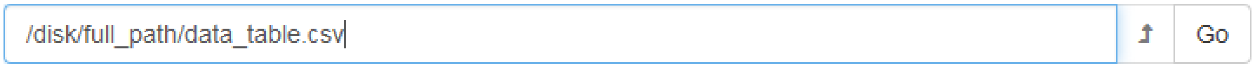

/disk/full_path/data_table.csv

Figure 18: File path field in remote file browser.

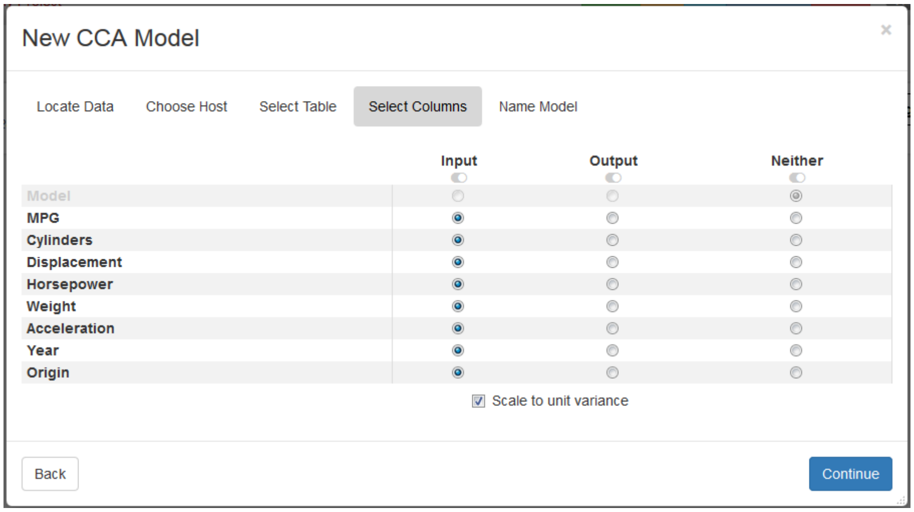
or (2) move up and down the directory hierarchy by clicking on folders in the list. Clicking on 📁 .. (the folder labeled '..' in the file list), or on the *Up Directory* ↥ button (to the right of the file path) moves you up a level in the hierarchy, while clicking on a named folder moves you down a level. Once you are in the directory that contains your table data, click on the file to select it. Ensure that the format shown in the *Filetype* dropdown matches the selected file's type, then click *Continue* to read the file. Note, you can change your mind and read the table from your *Local* host by clicking the Back button to return to the previous page.

Select Columns

Once the table has been read, either from a *Local* or a *Remote* source, the *Select Columns* page displays a list of the table's variable (column) names and asks you to categorize them as *Input*, *Output*, or *Neither* for the CCA analysis. Variables marked as *Neither* are omitted from the analysis altogether. Since CCA requires numeric values, strings are automatically excluded from consideration.

Looking at the variables in our *Cars* example in Figure 19, the faded variable name at the top of the list, *Model,* is the name for each car model. Because its values are all strings, it has been automatically set to *Neither* and cannot be changed. Although *Origin* is a numeric variable, the numbers are encoding categorical labels whose value order has no meaning (US = 1, Europe = 2, Asia =3). Because the values have no ordinal interpretation, *Origin* should also be removed from the analysis.

Since the number of inputs typically exceeds the number of outputs, we initialize all numeric variables to be inputs, leaving you to identify just the output and excluded variables. If variables shown for this table don't correspond to the ones you wanted or expected, you can click the *Back* button to select a different table file.

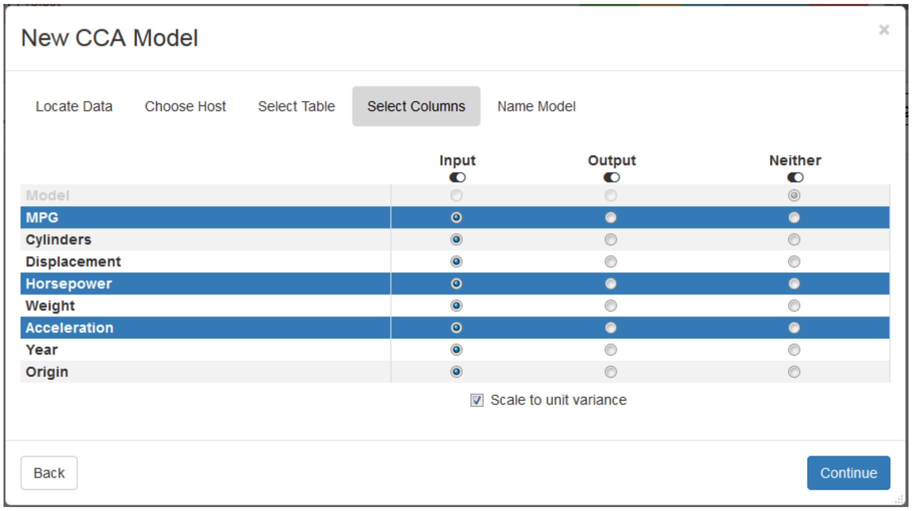


	Input	Output	Neither
Model	○	○	●
MPG	●	○	○
Cylinders	●	○	○
Displacement	●	○	○
Horsepower	●	○	○
Weight	●	○	○
Acceleration	●	○	○
Year	●	○	○
Origin	●	○	○

☑ Scale to unit variance

Figure 19: Initial configuration in the *Select Columns* dialog for the cars data set.

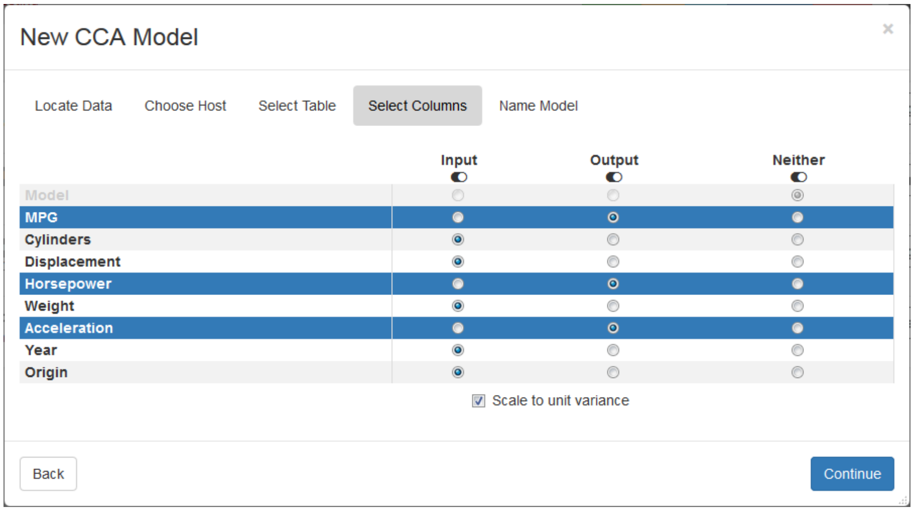
Variables can be marked one at a time by clicking the radio buttons, or they can be marked in larger groups by using either shift-click to select a contiguous group of variables, or by using control-click to pick a scattered set of rows (as demonstrated in Figure 20). For group selections, you must click on the rows near the variable names instead of near the radio buttons. Once you have highlighted a set of lines for joint assignment, click on the ◐ icon under the desired category to set the radio buttons for the group, as shown in Figure 21. Since CCA can be performed on any subset of variables, you can also use it to calculate correlations between multiple inputs and a single output, or between any two individual variables.



	Input ◐	Output ◐	Neither ◐
Model	○	○	◉
MPG	◉	○	○
Cylinders	◉	○	○
Displacement	◉	○	○
Horsepower	◉	○	○
Weight	◉	○	○
Acceleration	◉	○	○
Year	◉	○	○
Origin	◉	○	○

☑ Scale to unit variance

Figure 20: Click on the ◐ icon beneath *Output* to label the highlighted variables as outputs.



	Input ◐	Output ◐	Neither ◐
Model	○	○	◉
MPG	○	◉	○
Cylinders	◉	○	○
Displacement	◉	○	○
Horsepower	○	◉	○
Weight	◉	○	○
Acceleration	○	◉	○
Year	◉	○	○
Origin	◉	○	○

☑ Scale to unit variance

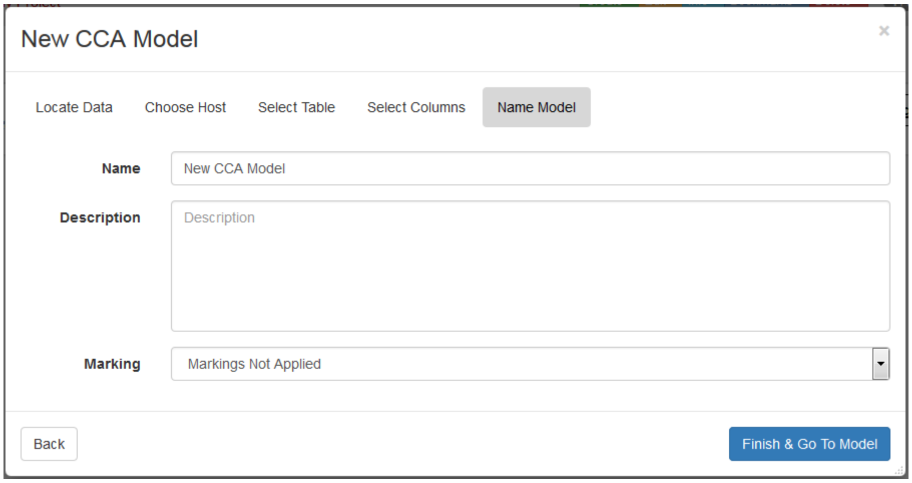
Figure 21: Result of using shift-click and the group assignment icon to select *Output* variables.

Sometimes value ranges between variables differ by many orders of magnitude, which can bias the analysis. The checkbox, *Scale inputs to unit variance*, permits you to normalize the values prior to running CCA. This feature is enabled by default. If you wish to perform the analysis using the original unscaled values, click within the box to remove the checkmark.

Once you have finished defining the input/output variables for the CCA analysis and have determined whether you want the values to be scaled, click *Continue* to go to the final step where you provide a name for your model.

Name Model

The final page of the CCA model creation wizard, shown in Figure 22, provides editable fields to enter a *Name* and an optional *Description* for the model. A default name of *New CCA Model* is provided, but the model list on the project page will become uninterpretable if you use this for all your models. Additionally, you should select a *Marking* from the dropdown list of choices. These markings are specific to your institution and the Slycat™ server you are using. The selected marking identifies the sensitivity of the data that is being analyzed, both for your own benefit and for other team members. This marking is used to label the model, both on the project page and within banners at the top and bottom of the model visualization. Once this information has been entered, click the *Finish & Go To Model* button in the lower right. Slycat™ will then transfer you to the model visualization page. When the analysis has completed, the CCA model will be displayed. If processing is still ongoing, the message "The model is being computed. Patience!" will be shown.



New CCA Model

Name: New CCA Model

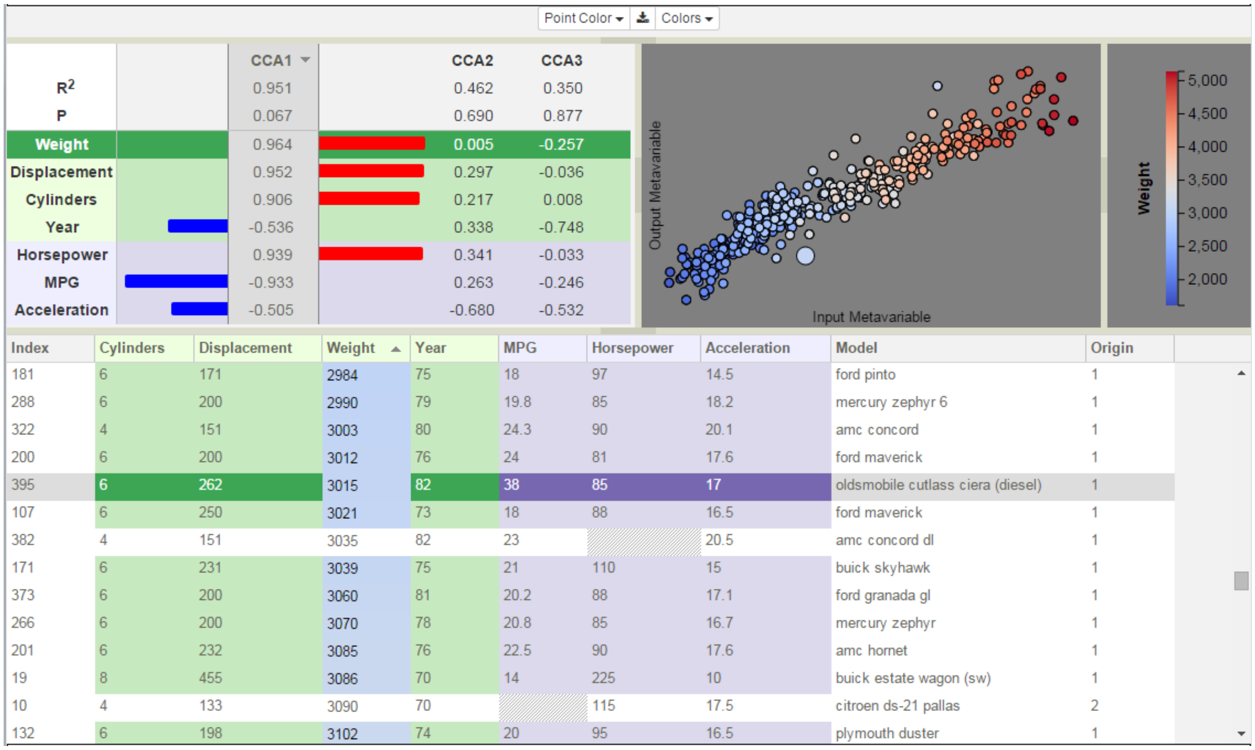
Description: Description

Marking: Markings Not Applied

Figure 22: The final step in CCA model creation is to name the model and apply markings.

CCA Model Visualization

As shown in Figure 23, visualization of a CCA model consists of three linked views, each providing a different level of abstraction. The most abstract level is the *Correlation View*, where each column displays the structure coefficients for one of the canonical components. The scatterplot in the *Simulation View* shows how well each individual run is described by the correlations found in the ensemble overall. The least abstract view is the *Variable Table*, which provides the raw data values contained in the original table file. The views are all linked, so changing the selection in one view will modify the selection in one or more of the other views. As with most Slycat™ models, the views are arranged with the ensemble level view in the upper left, the midrange view in the upper right, and the lowest level view at the bottom.



	CCA1	CCA2	CCA3
R^2	0.951	0.462	0.350
P	0.067	0.690	0.877
Weight	0.964	0.005	-0.257
Displacement	0.952	0.297	-0.036
Cylinders	0.906	0.217	0.008
Year	-0.536	0.338	-0.748
Horsepower	0.939	0.341	-0.033
MPG	-0.933	0.263	-0.246
Acceleration	-0.505	-0.680	-0.532

Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
181	6	171	2984	75	18	97	14.5	ford pinto	1
288	6	200	2990	79	19.8	85	18.2	mercury zephyr 6	1
322	4	151	3003	80	24.3	90	20.1	amc concord	1
200	6	200	3012	76	24	81	17.6	ford maverick	1
395	6	262	3015	82	38	85	17	oldsmobile cutlass ciera (diesel)	1
107	6	250	3021	73	18	88	16.5	ford maverick	1
382	4	151	3035	82	23		20.5	amc concord dl	1
171	6	231	3039	75	21	110	15	buick skyhawk	1
373	6	200	3060	81	20.2	88	17.1	ford granada gl	1
266	6	200	3070	78	20.8	85	16.7	mercury zephyr	1
201	6	232	3085	76	22.5	90	17.6	amc hornet	1
19	8	455	3086	70	14	225	10	buick estate wagon (sw)	1
10	4	133	3090	70		115	17.5	citroen ds-21 pallas	2
132	6	198	3102	74	20	95	16.5	plymouth duster	1

Figure 23: CCA model of cars data set displaying the first canonical component.

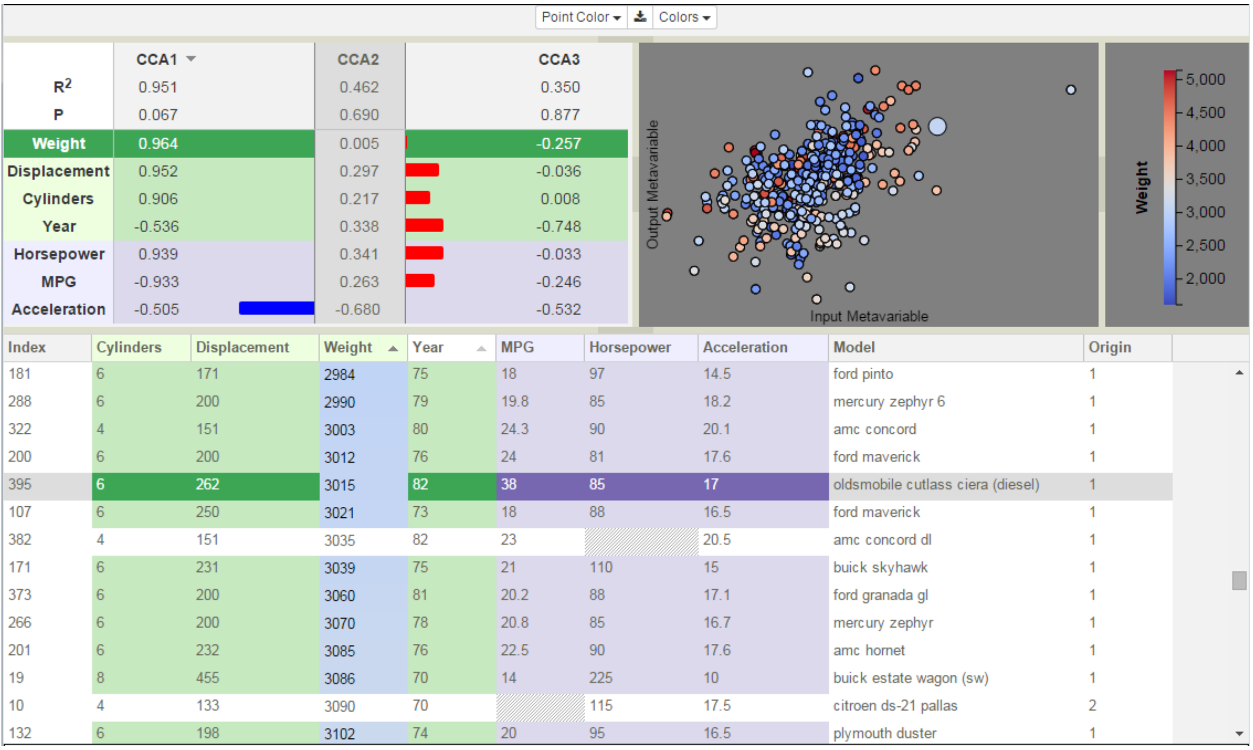
Correlation View

The *Correlation View* displays the relationships found between variables when the ensemble is viewed holistically. Each column of the *Correlation View*'s bar chart represents a different canonical component. These orthogonal components are ordered from left to right in decreasing importance, as shown by the decreasing R^2 and increasing *p*-values in the first two rows of each component column. Variable names are shown along the left edge with rows for input variables colored in green, and rows for output variables colored in lavender. The number of components returned by CCA is equal to the minimum of the number of inputs versus the number of outputs. So, for the example in Figure 23, where there are four inputs and three outputs, CCA will return three canonical components.

In the bar chart, only one canonical component is expanded at a time (e.g. in Figure 23, *CCA1* is expanded, while in Figure 24, *CCA1* is collapsed and *CCA2* is expanded). Clicking on a CCA column name changes the selected canonical component. This collapses the bar chart from the previously selected component and expands that of the new component.

The horizontal bars in the expanded bar chart visually encode the relationships between variables, both in terms of the magnitude of the structure coefficients, and in terms of the correlation type (positive or negative). Numeric values for the coefficients are displayed in the center of each column. Positive values are drawn as red bars extending towards the right. Negative values are drawn in blue extending to the left. The orientation combined with the color-coding acts to visually reinforce the relationship information. At a glance, you can see correlative relationships between variables and their strength by comparing the color, direction, and length of the bars. Positively correlated variables will display the same color and bar orientation, while negatively correlated variables will be opposed.

The bar chart rows can be sorted by variable strength. To the right of the CCA column name is a small triangular icon. Clicking on this icon sorts the columns by the unsigned magnitudes of the structure coefficients in the expanded column, though all columns will reflect the order returned by this sorting operation (i.e. the rows are sorted using this column as the key). The initial sort is descending and the orientation of the triangle reflects this by rendering the triangle with the wide edge at the top and the point at the bottom (e.g. *CCA1* in Figure 23). The sorting order is reversed if you click the triangle again. Ascending sorts are signified by rendering the triangle with the point at the top. Inputs and outputs are sorted independently. For long lists of variables, the input and output variable sets are independently scrollable. Note that in Figure 24, although *CCA2* is selected, the decreasing sort order from *CCA1* is still maintained. Sorting column is independent of component selection. Hovering over any of the CCA column headers, the sorting icon for that column becomes visible and can be clicked without needing to expand the component.



	CCA1 ▾	CCA2	CCA3
R²	0.951	0.462	0.350
P	0.067	0.690	0.877
Weight	0.964	0.005	-0.257
Displacement	0.952	0.297	-0.036
Cylinders	0.906	0.217	0.008
Year	-0.536	0.338	-0.748
Horsepower	0.939	0.341	-0.033
MPG	-0.933	0.263	-0.246
Acceleration	-0.505	-0.680	-0.532

Index	Cylinders	Displacement	Weight ▲	Year ▲	MPG	Horsepower	Acceleration	Model	Origin
181	6	171	2984	75	18	97	14.5	ford pinto	1
288	6	200	2990	79	19.8	85	18.2	mercury zephyr 6	1
322	4	151	3003	80	24.3	90	20.1	amc concord	1
200	6	200	3012	76	24	81	17.6	ford maverick	1
395	6	262	3015	82	38	85	17	oldsmobile cutlass ciera (diesel)	1
107	6	250	3021	73	18	88	16.5	ford maverick	1
382	4	151	3035	82	23		20.5	amc concord dl	1
171	6	231	3039	75	21	110	15	buick skyhawk	1
373	6	200	3060	81	20.2	88	17.1	ford granada gl	1
266	6	200	3070	78	20.8	85	16.7	mercury zephyr	1
201	6	232	3085	76	22.5	90	17.6	amc hornet	1
19	8	455	3086	70	14	225	10	buick estate wagon (sw)	1
10	4	133	3090	70		115	17.5	citroen ds-21 pallas	2
132	6	198	3102	74	20	95	16.5	plymouth duster	1

Figure 24: CCA model of *Cars* data set displaying the second canonical component.

Clicking on a row in the bar chart selects that variable for color-coding the points in the *Simulation View* (i.e. each simulation point is color-coded according to its value for that variable). The variable row is highlighted by darkening the background color and changing the font color to white. The color palette, shown in the *Legend* alongside the value range for the color-coding variable, corresponds to the currently selected theme (see Color Themes). This same color-coding is applied to the cell backgrounds of this column of the *Variable Table*, as is demonstrated by the *Weight* column in both Figure 23 and Figure 24.

Simulation View

The *Simulation View* is a scatterplot, in which each point represents an ensemble member. The axes of the scatterplot are the canonical variables, $\mathbf{a}^{T}X$ and $\mathbf{b}^{T}Y$, which are labeled as *Input Metavariable* and *Output Metavariable*. The *x* and *y* coordinates of each point are weighted sums of that point's input and output variable values, respectively. Because the values of the canonical variables differ for each canonical component, changing the selected component in the *Correlation View* changes the point coordinates, which are then re-rendered in the scatterplot. Comparing the scatterplots in Figure 23 and Figure 24, you can see how the point locations shift from a loose diagonal for *CCA1*, to a ball of points for *CCA2*. Given the low R^2 and high *p*-value for *CCA2*, the scatterplot point placement visually reveals the poor quality of the result (all points would be on the diagonal in an ideal result).

Color-Coding Points

The first time a model is rendered, the points are colored by their index number. There are three mechanisms for changing the variable that is used to do the color-coding: clicking on a variable in the *Correlation View*, clicking on the column header in the *Variable Table*, or selecting a variable from the *Point Color* dropdown list. Irrespective of the interface used, changing the variable selected for color mapping will lead to changes in all three views and the legend, including: highlighting the newly selected variable's row in the bar chart, recoloring the scatterplot using the new variable's values, coloring the cell backgrounds in its table column, returning the cell backgrounds of the previously selected variable's column to its default color (green, lavender, or white depending on its type), and relabeling and redefining the value range in the *Legend* (see below). Note that the table may include variables that are not present in the *Correlation View*, columns that are neither inputs nor outputs. These columns are drawn on the right end of the table against white backgrounds. These provide additional color-coding options (numeric variables only), however, the bar chart will not be highlighted because it only includes variables passed to CCA.

Legend

To the right of the scatterplot is the *Legend*. The *Legend* is in its own view, which can be resized or closed altogether. The *Legend* displays information about the current color-coding variable, including its name, range of values, and the mapping between values and colors. The color palette is defined by the current theme (see Color Themes).

Selecting Points

Points in the scatterplot may be selected through several mechanisms. The simplest is to place the mouse cursor over a point and click the left mouse button. The selected point is redrawn in the plot with a larger radius, while simultaneously in the *Variable Table*, the row corresponding to the selected point is darkened and scrolled to be visible.

For groups of adjacent points that lie within a rectangular region, rubber banding can be used to draw a rectangle around the desired point set. Position the mouse at one corner of the region. Press the left mouse button down while simultaneously moving the mouse towards the opposite corner of the region. A yellow rectangle will be drawn between the location of the initial button-press and the mouse's current position. Move the mouse until the rectangle encloses all the desired points, then release the mouse button to finish the selection.

Holding the control key while selecting new points, either through clicking or rubber banding, will add these additional points to the previously selected set. Alternately, scatterplot points can be selected by picking rows in the *Variable Table* (see Simulation Selection).

Clicking in the background (not on any point) deselects all previously selected points.

Variable Table

The *Variable Table* provides access to the original numeric variable values for each simulation run. It is essentially an interactive version of the original table data, where each column represents a single variable and each row contains the variable values for a single ensemble member. Within the table, the cell backgrounds take on one of four color-encodings: input variables are green, outputs are lavender, non-designated variables are white, and the elements of the selected variable are individually colored by their value using the current color map (see Color Themes). Coloring table elements by value highlights the selected color-coding variable, while concurrently providing color correspondence between rows and scatterplot points. The interactive capabilities of the table include: sorting within columns, column (variable) selection, and row (simulation) selection.

Rows that have not been used in the CCA analysis have white backgrounds. CCA requires that all rows have values for each of the columns that are included in the analysis. As shown in Figure 24, the cars in rows 382 and 10 are missing values for *Horsepower* and *MPG*, respectively. The missing data are shown with no numeric value and a hatched gray background in the table. Since the columns with missing values were declared as outputs during model creation, rows 382 and 10 have been entirely excluded from the analysis and are drawn in white. If CCA is later rerun, and if *Horsepower* and *MPG* are removed from the analysis (if the columns are marked as *Neither* when creating the new model), then these rows will be included in the calculation and will be color-coded.

Sorting

Sorting allows rapid identification of simulations whose variable values are extrema. Additionally, sorting facilitates comparison between simulations whose values are similar within one variable, but whose values for other variables might differ significantly. For instance, in Figure 23 and Figure 24 the tables are sorted by weight. The highlighted row appears adjacent to cars having similar weights, yet the selected car is notable because it gets much better gas mileage.

Although the sorting order is defined by values within a single column, the full row moves in the reordering. To sort on a variable, left-click the small triangular icon in the column header. The initial sort is ascending and the orientation of the triangle reflects this by rendering the triangle with the tip at the top and the wide edge at the bottom. If you click the triangle again, the sorting order is reversed to be descending and the triangle is redrawn with the point at the bottom.   The sorting column is independent of variable selection (see Color-Coding Points). Hovering over any of the column headers, the sorting icon for that column becomes visible and may be clicked to initiate sorting on that variable.

Variable Selection

Left-clicking the variable name in a column header selects that variable to color-code the scatterplot points in the *Simulation View*, to color the cell backgrounds in its associated table column, and to highlight that variable's row in the *Correlation View*.

Simulation Selection

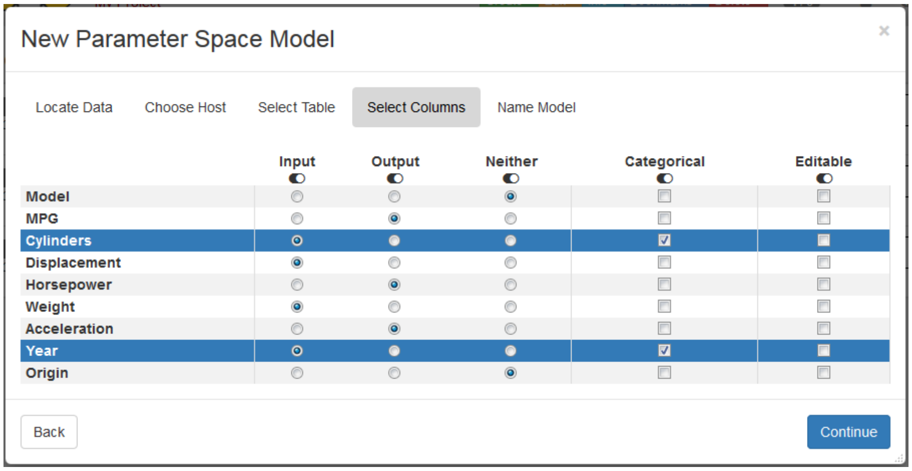
Left-clicking within a table row selects that simulation, which is then highlighted in both the table and the scatterplot. Multiple row selection, as is commonly performed on lists, consists of clicking on a starting row, then using shift-click to select the ending row (either above or below the starting row). This selects both the starting and ending rows along with all rows in between. Individual rows may be added to an existing selection by clicking on them while pressing the control-key. Selected rows are highlighted in the table by increasing the saturation of cell backgrounds, except for cells in the color-coding column (see Variable Table). Selected scatterplot points are highlighted by being redrawn with an increased radius.

Parameter Space Model

Unlike the CCA and Time Series models, the Parameter Space model does not perform an analysis step. Instead, the Parameter Space model is an exploratory visual interface that combines a filterable scatterplot representation with remote access to images, videos, and other media-based ensemble data.

Creating a Parameter Space Model

Like the CCA model, the core of the Parameter Space model is table data. Up until the stage where inputs and outputs for the model are selected, the model creation steps are identical to CCA (see Creating a CCA Model). Instead of initializing all variables as *Input*, the variables default to being assigned as *Neither*. As with CCA, group selection operations using shift-click and/or control-click allow rapid assignment of variable types (see Select Columns). However, a central difference in the Parameter Space model is that variables can also be designated as being *Categorical* and/or *Editable*.



	Input	Output	Neither	Categorical	Editable
Model	○	○	●	□	□
MPG	○	●	○	□	□
Cylinders	●	○	○	☑	□
Displacement	●	○	○	□	□
Horsepower	○	●	○	□	□
Weight	●	○	○	□	□
Acceleration	○	●	○	□	□
Year	●	○	○	☑	□
Origin	○	○	●	□	□

Figure 25: Parameter Space model creation wizard dialog for designating variable attributes, including *Categorical* and *Editable*.

Categorical variables are those with a limited number of discrete values. During scatterplot filtering, it is often advantageous to be able to turn on or off points that are associated with specific values or combinations of values. *Categorical* variables are filtered using labeled buttons, which can individually be set *on* or *off*. Continuous variables are filtered using a slider, which is limited to defining a single range of values to be included or excluded. By declaring *Year* and *Cylinders* as *Categorical* variables during model creation, the example in Figure 26 shows how we can filter the scatterplot display to be only those cars having 3, 5, or 8 cylinders that were manufactured in even-numbered years. This fine-grained filtering of the data would be impossible using a range slider.

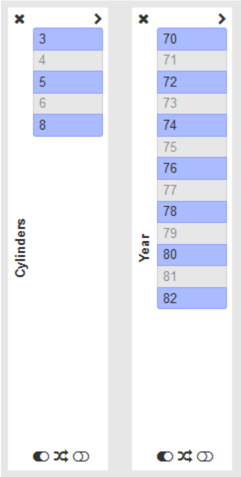


Figure 26: Filters for *Categorical* variables enumerate each discrete value as a labeled button, enabling filtering operations that would not be possible using range sliders. The values selected here (shown in dark blue) limit the scatterplot to display just those cars with 3, 5, or 8 cylinders that were manufactured in even-numbered years

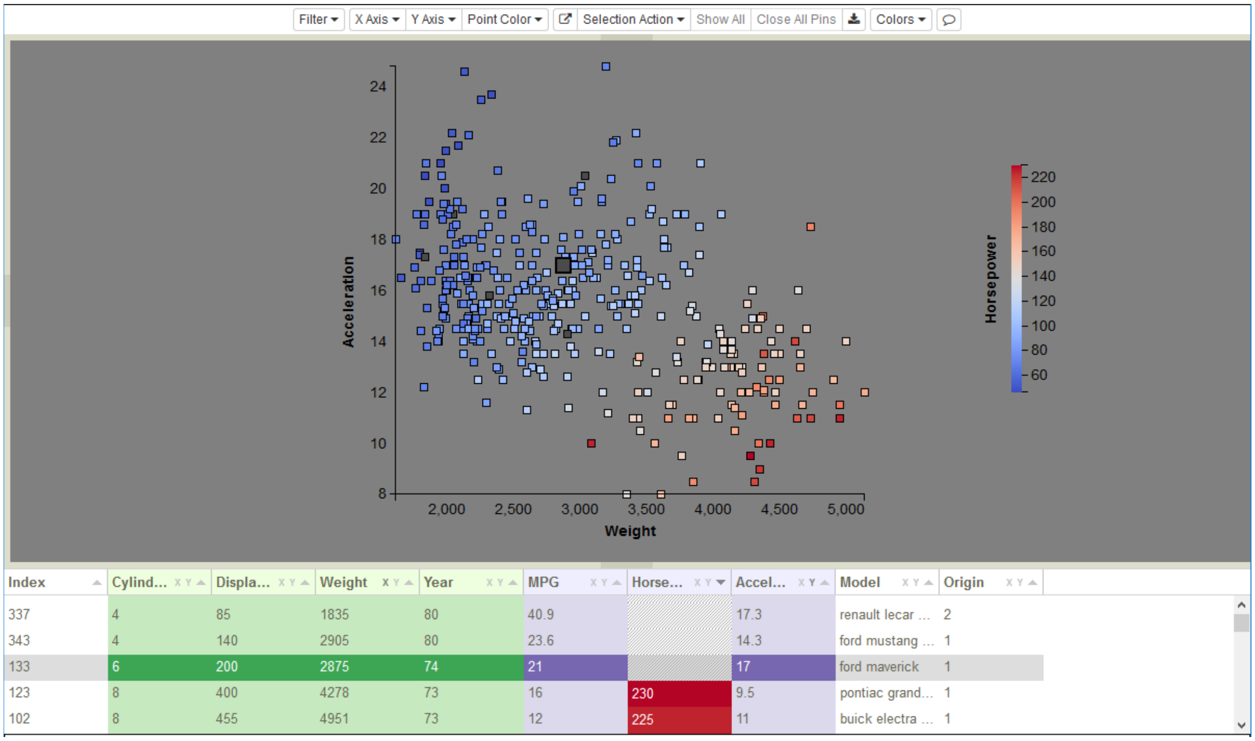
An *Editable* variable is one that can be modified by the user. The type of values originally in the variable define the type of values that can be substituted (i.e. numeric variables cannot be changed into text strings). A variable can only be defined as *Editable* during model creation. The mechanism for changing variable values is through selection (see Selecting Points). Note that value modification actions in the *Selection Action* dropdown list are only enabled when an *Editable* variable has been declared.

Parameter Space Model Visualization

As shown in Figure 27, the Parameter Space model page consists of linked scatterplot and data table views, combined with interactive filtering, data manipulation, and remote viewing of images and other media. The *Scatterplot View* provides an abstract representation of the ensemble members and their value distributions within the variables selected for the axes. The *Variable Table* provides the raw data values contained in the original table file. Changes made to the plot or the table will cause corresponding changes in the other.

As with the CCA Model Visualization example, we will be using the *Cars* data set in the following sections to illustrate some of the Parameter Space Model features. We will also be using the Taylor Anvil Impact Scenario (TAIS) ensemble, which includes image data, to demonstrate some of the media-based functionality. TAIS was generated using Sierra/SolidMechanics [8] (a Lagrangian, three-dimensional code for problems with large deformations and nonlinear material behaviors) in combination with ParaView/Catalyst [3]. The images were generated in situ (at the same time as the physics simulation) using Catalyst. The simulation is of an Oxygen Free High Conductivity (OFHC) copper cylinder, 2.54 cm long with a diameter of .762 cm and an initial velocity of 190 m/sec, impacting a rigid wall. The ensemble is a sensitivity study that evaluates the effects of changing four parameters of the Johnson-Cook inelastic constitutive law: ajo, bjo, njo, and beta. The height and radius of the cylinder after the impact are compared to experimental photographic results. Two output metrics are calculated for each run, ndrf_last and ndhf_last. These variables are the **n**ormalized **d**ifferences between the **r**adius/**h**eight of the **f**inal cylinder state from the **last** timestep of the simulation and the final radius/height of the cylinder in the experiment, respectively. Since the differences would decrease in those cases where the simulation more closely matched the experimental results, the optimal case would be a simulation where the values of these two metrics were zero (i.e. there is no difference between the simulation and the experiment).

Scatterplot View



Index	Cylind...	Displa...	Weight	Year	MPG	Horse...	Accel...	Model	Origin
337	4	85	1835	80	40.9	▨	17.3	renault lecar ...	2
343	4	140	2905	80	23.6	▨	14.3	ford mustang ...	1
133	6	200	2875	74	21	▨	17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand...	1
102	8	455	4951	73	12	225	11	buick electra ...	1

Figure 27: Parameter Space model visualization of the *Cars* data set.

The *Scatterplot View* represents each ensemble member as a point in a two-dimensional plot, where the variables that are used for the x-axis, y-axis, and point color-coding are interactively selected. As with the CCA Model, when individual points or groups of points are selected within the plot (see Selecting Points), corresponding rows in the *Variable Table* are simultaneously selected, and vice versa.

However, the Parameter Space Model's *Scatterplot View* possesses additional capabilities that the CCA Model's *Simulation View* lacks. If the *Variable Table* contains media columns (URIs for images, videos, time series tables, or STLs), hovering over a point with the mouse can be used to retrieve media items from remote HPC systems. The media variable to be retrieved is selected through a *Media Set* dropdown list, which only appears in the controls (as shown in Figure 28) when media columns are present in the table.

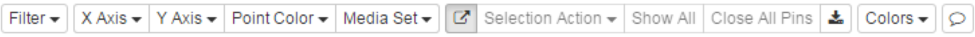


Figure 28: Full set of Parameter Space model-specific controls.

Another key feature in the *Scatterplot View* is the ability to reduce the number of visible points through filtering or point hiding. This is important for interacting with large ensembles, since point overlap and occlusion increase with ensemble size. Additionally, the *Scatterplot View* enables the inclusion of points with missing data. Although a point still requires values to exist in both axis variables to define its coordinates, none of the other variables need to have a value for the point to be displayed. The points colored dark gray in Figure 27 are examples of rows with missing values for *Horsepower*. One of these is row 133, which is highlighted in both the scatterplot (enlarged point) and the table (darkened row).

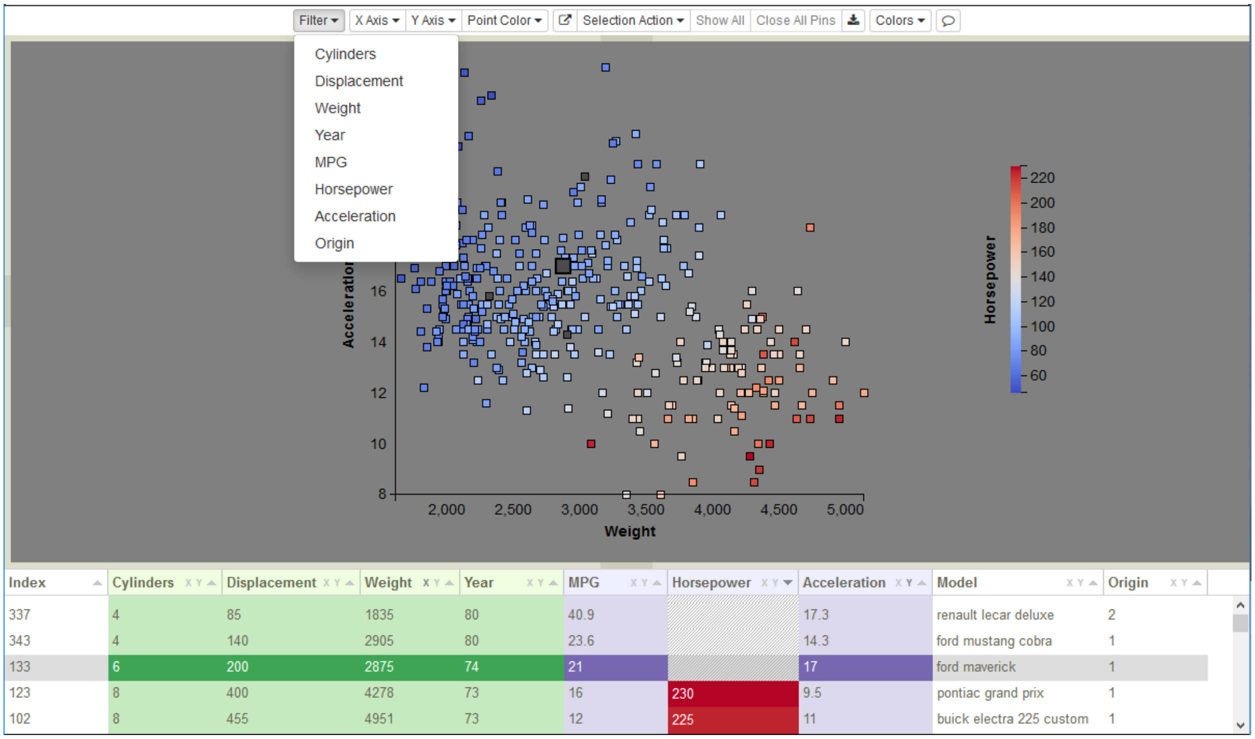
Filter

There are two mutually exclusive mechanisms in the Parameter Space scatterplot for removing points from the view, filtering and hiding selected points. Once a filter is present, point hiding operations in the *Selection Action* dropdown are disabled. If points had previously been hidden using point hiding, they immediately become visible again as soon as any filter is selected. This ensures that during filtering, visible points only reflect the filtering choices. We did this because we wanted screenshots of the model that included filters to be self-documenting as to which values were retained and which had been removed. This is useful for interpreting slides or figures outside of the Slycat™ interface.

However, sometimes manually removing points through selection is the only way to define a set of points matching criteria that are not achievable through filtering. Since multiple selections may be required to construct this set, we want to avoid discarding it unintentionally. Consequently, the last visibility state is saved when filters are enabled. Once all filters have been removed, the scatterplot returns to its previous state of visibility (i.e. any points that were previously hidden through the selection mechanism will return to being hidden). To explicitly return to a state of total point visibility, click the *Show All* button.

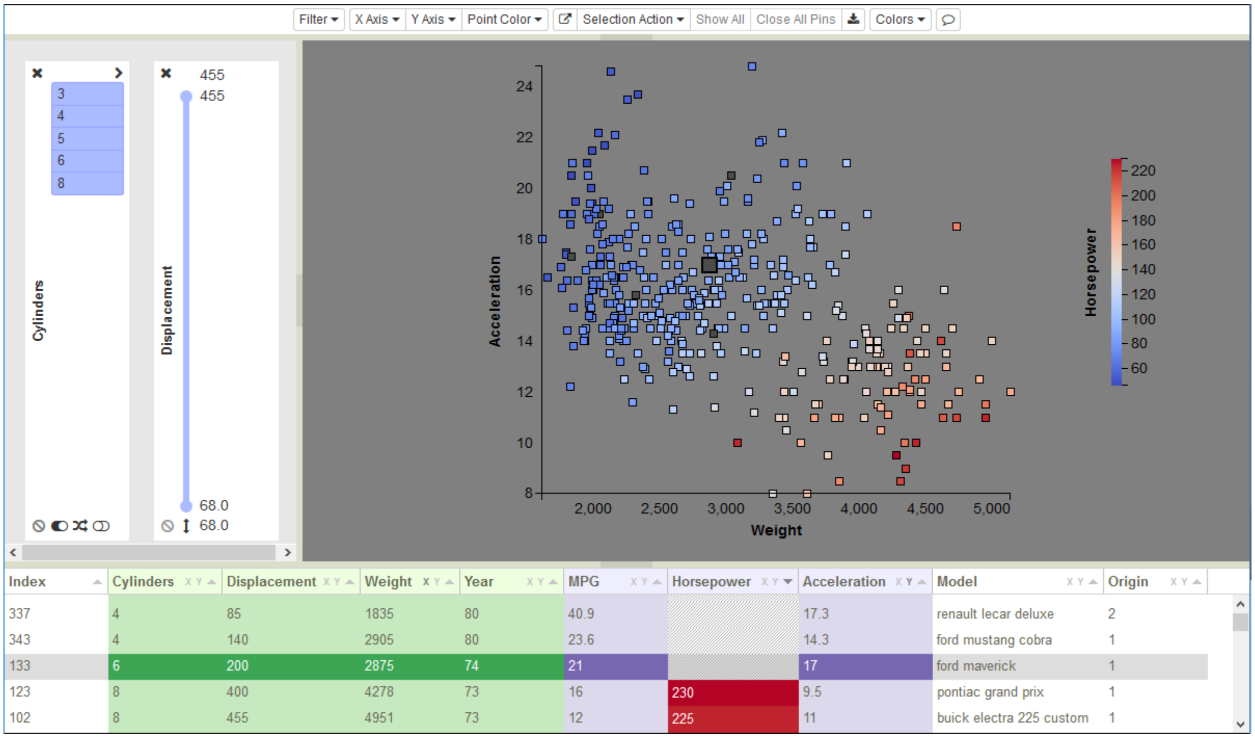
Similarly, filter states are saved, so if you remove a filter and then reinstate it, the filter will resume with its previous settings (button states or slider range). This way, if you accidentally remove a filter and don't remember the settings, you can instantly restore your state.

To add a filter, click on the *Filter* button to drop down a list of variables, such as those for the cars data shown in Figure 29. Selecting a variable from the list displays a filter consisting of either a set of buttons for categorical variables (see Creating a Parameter Space Model for how to define a categorical variable), or a slider for continuous variables (Figure 30 shows examples of each). To remove a filter, click the ✖ icon in the upper left corner of the filter. In both types of filters, blue coloration of the buttons or the slider range indicates values that are visible, whereas gray is used for values that are not visible. As filters are changed to remove values, corresponding points are removed from the scatterplot and their rows in the table are grayed out. Conversely, as filters add values back into the visible set, corresponding points will reappear in the plot and their table rows will be re-colored.



Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
337	4	85	1835	80	40.9		17.3	renault lecar deluxe	2
343	4	140	2905	80	23.6		14.3	ford mustang cobra	1
133	6	200	2875	74	21		17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand prix	1
102	8	455	4951	73	12	225	11	buick electra 225 custom	1

Figure 29: *Filter* dropdown variable list for cars data.



Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
337	4	85	1835	80	40.9		17.3	renault lecar deluxe	2
343	4	140	2905	80	23.6		14.3	ford mustang cobra	1
133	6	200	2875	74	21		17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand prix	1
102	8	455	4951	73	12	225	11	buick electra 225 custom	1

Figure 30: Examples of both a categorical and a continuous filter.

Categorical Filters

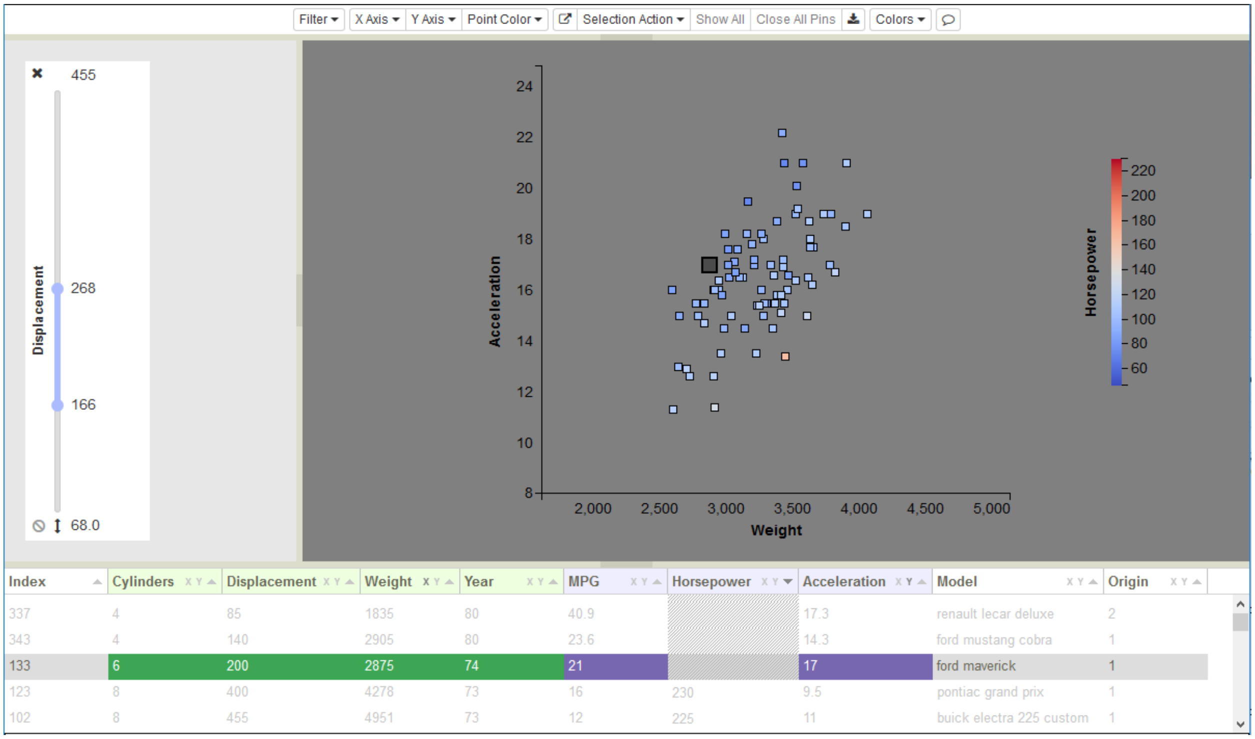
Initially, all filter buttons are on (visible values are colored blue), so all values are visible the first time that a filter is instantiated. Clicking a blue button sets its state to off, colors the button gray (hidden values are colored gray), hides points with that value in the scatterplot, and grays/fades the corresponding rows in the table. Buttons are toggles, so clicking the button again restores that value's point visibility in the plot and re-colors those rows in the table.

In addition to individual buttons, group operations are also available through the three icons ◐🔀○ at the bottom of the filter. Use ◐ to turn all buttons on, ○ to turn all buttons off, and 🔀 to flip the states of all buttons. As an example of using group operations, imagine that you wanted to see only cars with 5 cylinder engines. You could turn all buttons off with ○, then click button 5 to show just those points. This would be faster than individually turning off the 3, 4, 6, and 8-cylinder buttons.

For categorical filters, sometimes the width of the button label exceeds the width of the button. To expand the filter width, click the › icon in the upper right corner of the filter. This will widen the filter and replace the › icon with the ‹ icon. To collapse the filter back to its original width, click the ‹ icon.

Continuous Filters

The first time a filter is used, the entire range of the variable is visible (drawn in blue), as in Figure 30. The maximum and minimum variable values are at the top and the bottom of the filter, while the max/min values for the visible range are shown next to the slider endpoints. These max/min values interactively track the slider endpoints, changing both the value and position. The excluded portions of the range appear in gray, as demonstrated in Figure 31, where the maximum value has been dragged down to 268 and the minimum value has been pulled up to 166. In addition to grabbing and dragging the endpoints, you can drag the visible range as a unit, creating a sliding window of fixed length. The scatterplot and table are only redrawn when you stop moving the mouse or release the button.



Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
337	4	85	1835	80	40.9	▨	17.3	renault lecar deluxe	2
343	4	140	2905	80	23.6	▨	14.3	ford mustang cobra	1
133	6	200	2875	74	21	▨	17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand prix	1
102	8	455	4951	73	12	225	11	buick electra 225 custom	1

Figure 31: Slider after dragging adjusting *Displacement* max/min values.

Sometimes the resolution of slider increments can be a problem. The values associated with the set of unique slider positions may not include the value that you want to use as your threshold. So, the minimum and maximum threshold values are editable. To edit the threshold extrema, hover over the value (which will then turn orange as in Figure 32), type in the new value, and hit enter.

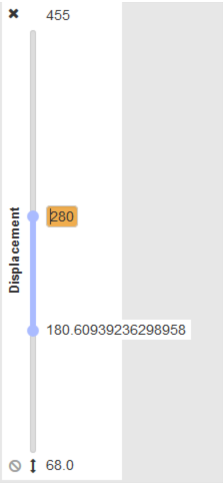
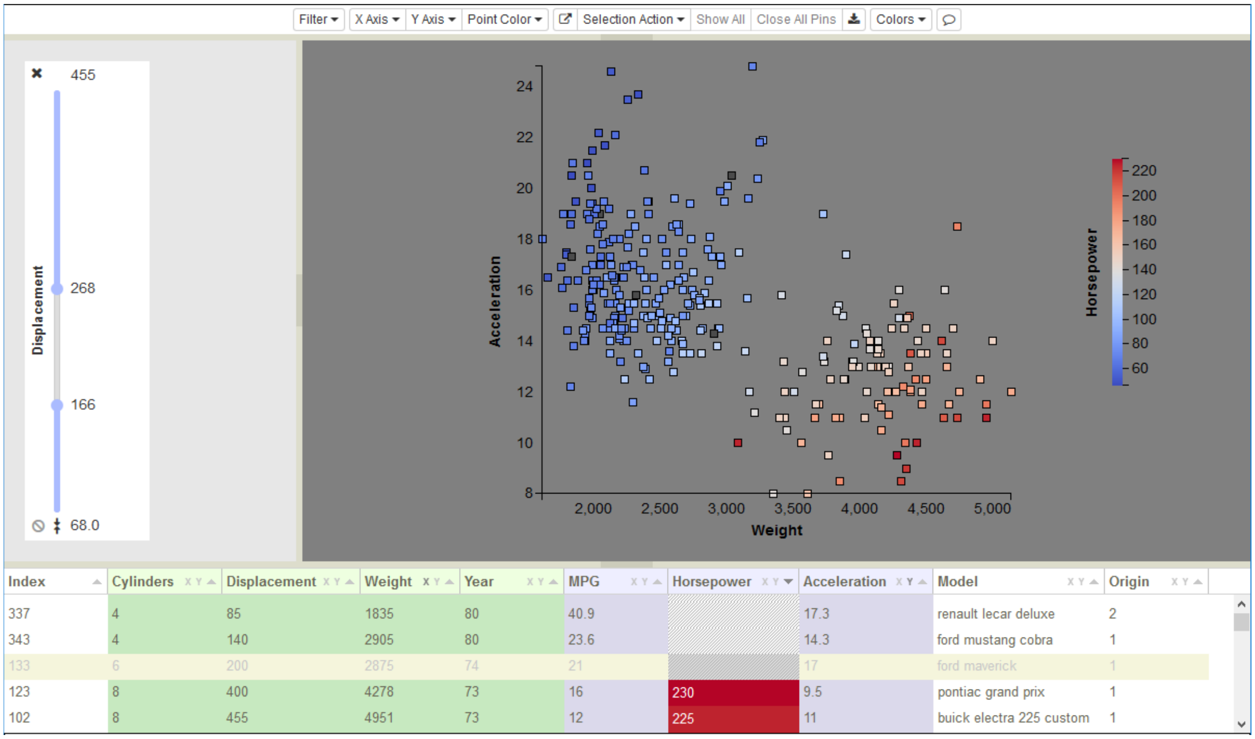


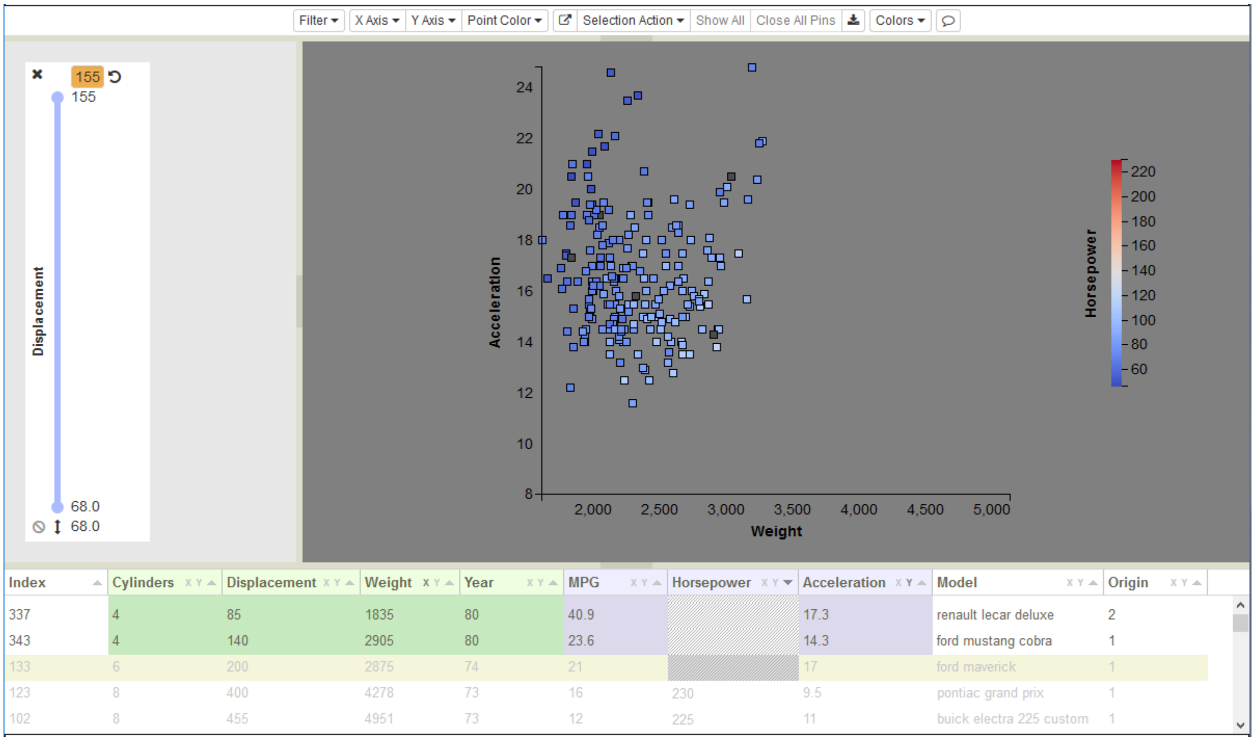
Figure 32: Editing a slider maximum threshold value.

On startup, the slider defines a single region of values that are visible. The ↕ icon beneath the slider indicates that this is the current mode of operation. Clicking on that icon inverts the mode, so values in the middle region become hidden and values at the ends are visible. The icon changes to ‡, the gray end regions become blue, and the central blue region is drawn in gray. Figure 33 shows the result of inverting the *Displacement* slider selection shown in Figure 31.



Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
337	4	85	1835	80	40.9		17.3	renault lecar deluxe	2
343	4	140	2905	80	23.6		14.3	ford mustang cobra	1
133	6	200	2875	74	21		17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand prix	1
102	8	455	4951	73	12	225	11	buick electra 225 custom	1

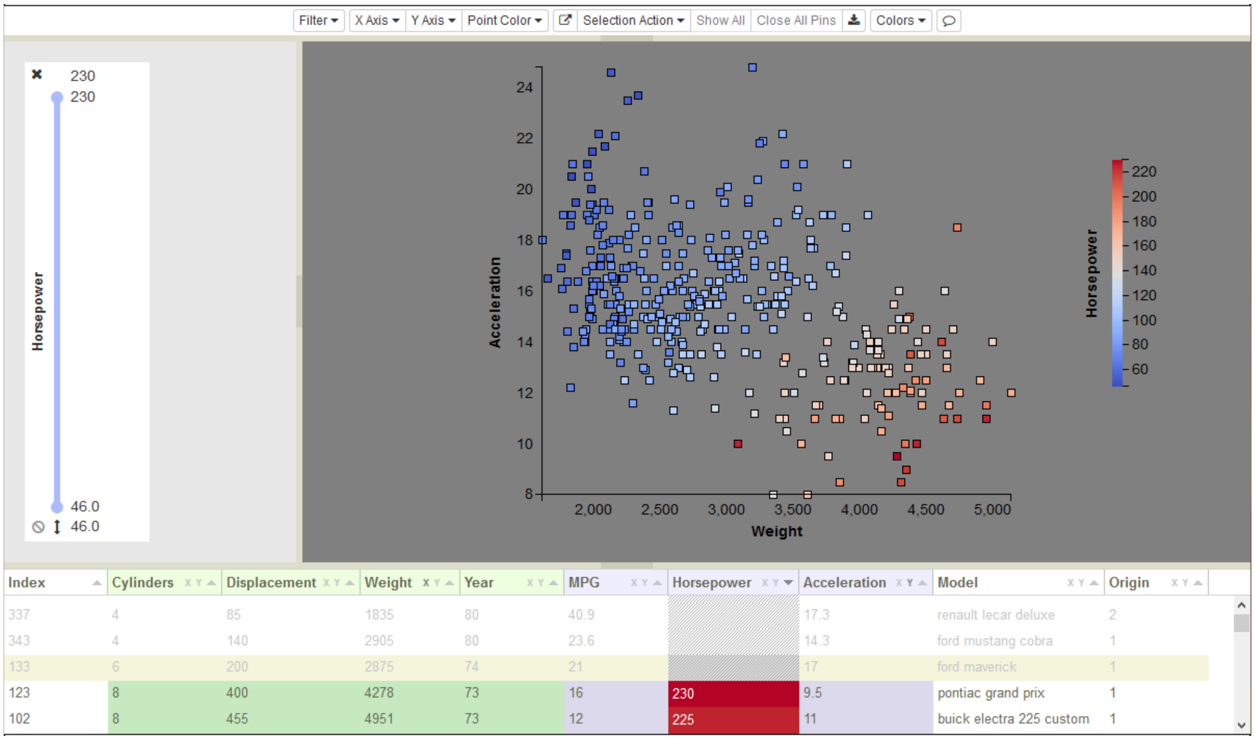
Figure 33: Invert slider selection to display only values outside the selected range.



Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
337	4	85	1835	80	40.9		17.3	renault lecar deluxe	2
343	4	140	2905	80	23.6		14.3	ford mustang cobra	1
133	6	200	2875	74	21		17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand prix	1
102	8	455	4951	73	12	225	11	buick electra 225 custom	1

Figure 34: Resetting slider maximum to increase resolution within slider range.

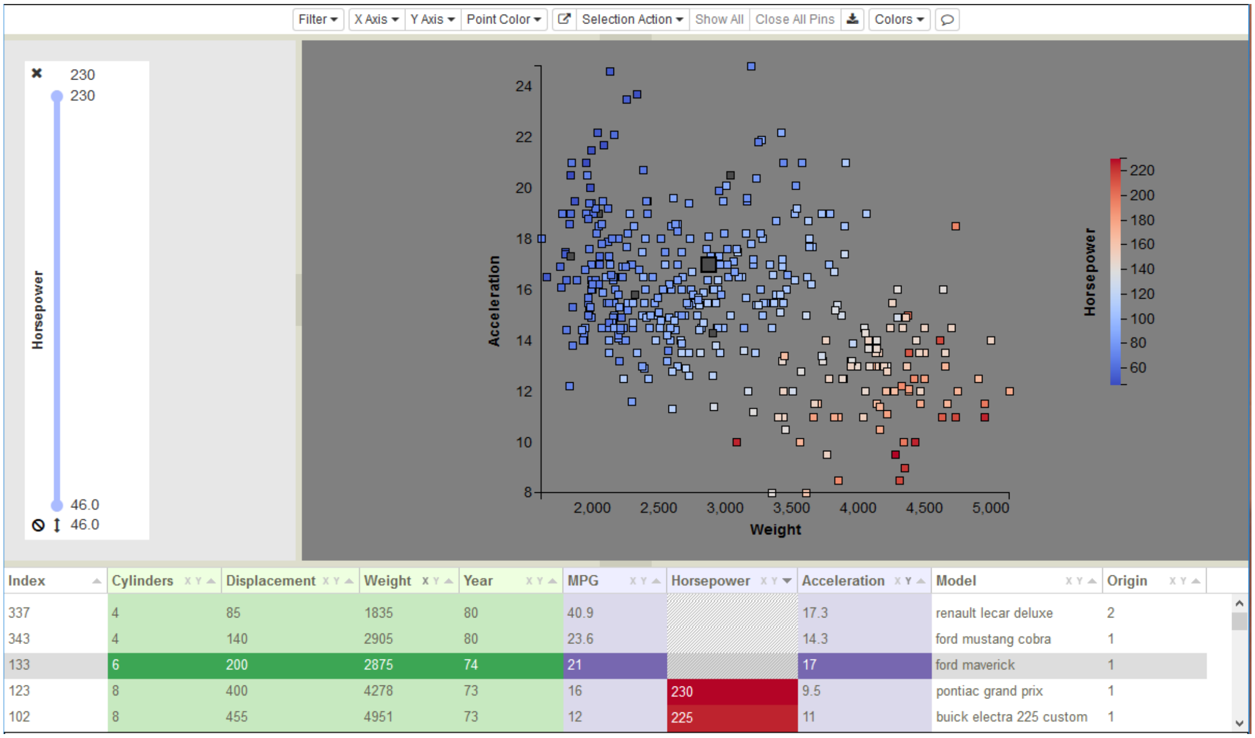
Another type of problem arises when the slider range itself is skewed by anomalous values that are far outside of the normal range for a variable (say points with values of 10,000, when the normal values are between 0 and 1). This forces most of the slider's value range to be empty, with all the points on the extreme ends. To eliminate this value bias, you can reset the overall slider range by editing the maximum and/or minimum range values at the top or bottom of the slider. Values outside of the revised range are hidden in the scatterplot and grayed out in the table. To edit the range extrema, hover over the value (which will then turn orange), type in the new value, and hit enter. In Figure 34, we have reset the filter maximum from 455 to 155. The maximum retains an orange background as a reminder that it has been modified. To reset the value back to the original max/min, click the ↺ icon.



Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
337	4	85	1835	80	40.9		17.3	renault lecar deluxe	2
343	4	140	2905	80	23.6		14.3	ford mustang cobra	1
133	6	200	2875	74	21		17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand prix	1
102	8	455	4951	73	12	225	11	buick electra 225 custom	1

Figure 35: Filtering normally excludes points with missing values.

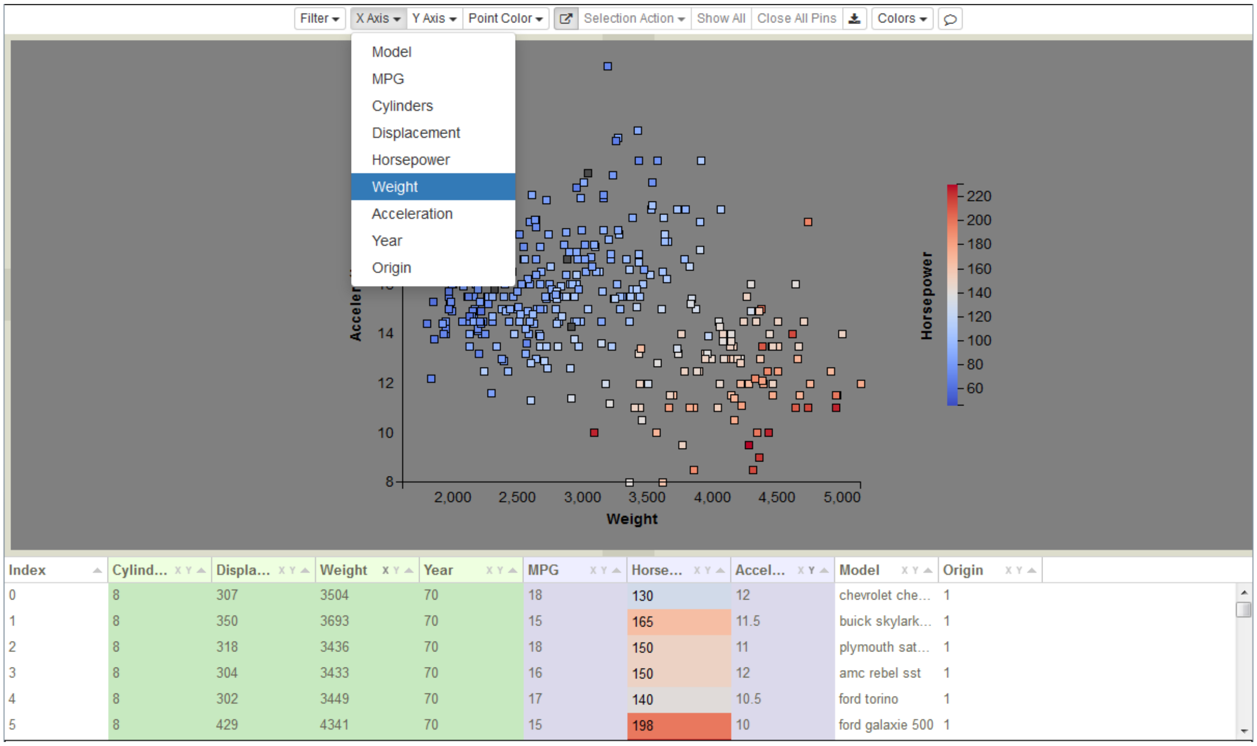
When filtering is performed on a variable where there are missing or non-defined data values (e.g. NULLs, NANs, Inf, -Inf), such as the *Horsepower* variable in Figure 35, those points corresponding to missing data are eliminated, as they are undefined and therefore not part of the range. This presents a problem if you want to compare points with missing data to points from a subset of that variable's value range. The ⊘ icon at the bottom of each filter enables the inclusion of points with missing data relative to that variable into the scatterplot. When a filter is first instantiated, the missing values are filtered out and the ⦸ icon is gray. Click ⦸ to make those points visible, as shown in Figure 36. The icon acts as a toggle, so clicking ⊘ will hide those points again.



Index	Cylinders	Displacement	Weight	Year	MPG	Horsepower	Acceleration	Model	Origin
337	4	85	1835	80	40.9	▨	17.3	renault lecar deluxe	2
343	4	140	2905	80	23.6	▨	14.3	ford mustang cobra	1
133	6	200	2875	74	21	▨	17	ford maverick	1
123	8	400	4278	73	16	230	9.5	pontiac grand prix	1
102	8	455	4951	73	12	225	11	buick electra 225 custom	1

Figure 36: Filter plus missing-valued points.

X Axis and Y Axis



Index	Cylind...	Displa...	Weight	Year	MPG	Horse...	Accel...	Model	Origin
0	8	307	3504	70	18	130	12	chevrolet che...	1
1	8	350	3693	70	15	165	11.5	buick skylark...	1
2	8	318	3436	70	18	150	11	plymouth sat...	1
3	8	304	3433	70	16	150	12	amc rebel sst	1
4	8	302	3449	70	17	140	10.5	ford torino	1
5	8	429	4341	70	15	198	10	ford galaxie 500	1

Figure 37: Selecting the *X Axis* variable using the dropdown list. *Horsepower* is used to color-code the points.

There are two mechanisms for selecting variables for the X and Y axes. They can be selected using the *X Axis* and *Y Axis* dropdown lists. Or, they can be selected by clicking on the **X** or **Y** icons in the *Variable Table* column headers. The icons are found in each column to the right of the variable name. The two mechanisms are linked, so regardless of which one is used to make the choice, the selection is shown in both. For example, in Figure 37, *Weight* is highlighted in the dropdown list and the **X** icon is darkened in the *Weight* column of the table. Once the selection is changed, the corresponding axis is immediately redrawn displaying the name and value range for the new variable. Points are also re-rendered as each point's coordinates are changed to reflect the values of the new variable.

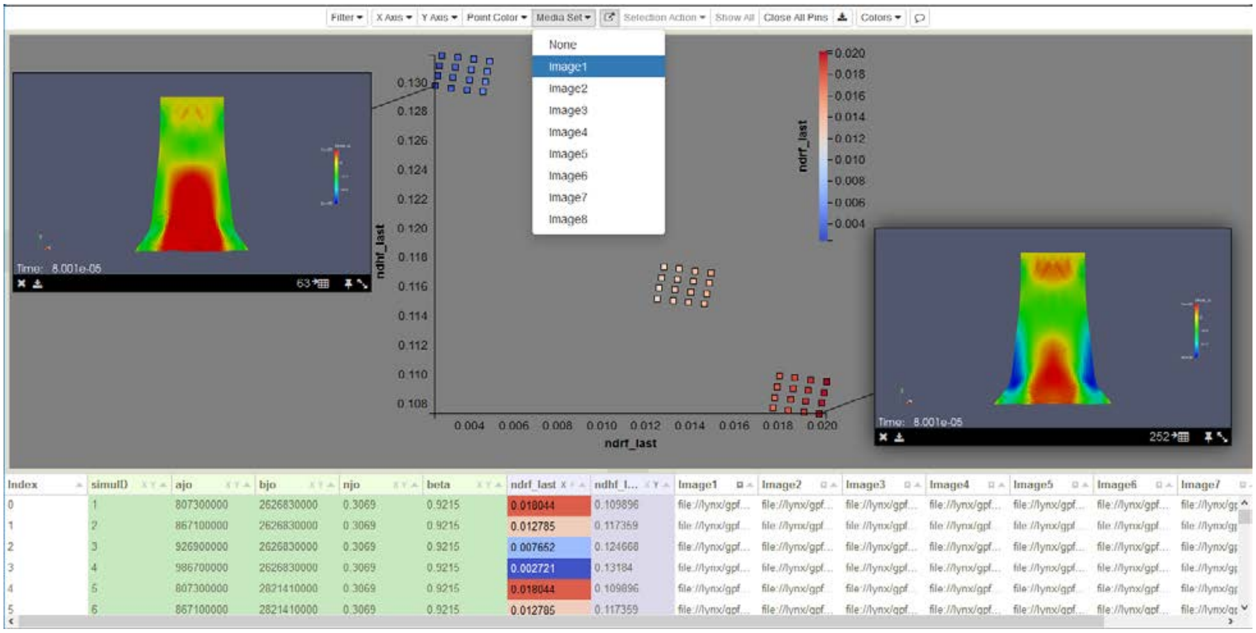
Point Color

Similarly, there are two mechanisms for selecting the variable for color-coding individual points. Either select the variable in the *Point Color* dropdown list, or click on the variable name in the *Variable Table* column header. The two mechanisms are linked, so regardless of which one is used to make the choice, the selection is shown in both. The variable name is highlighted in the dropdown, and the backgrounds of that column's elements in the table are color-coded to match the corresponding point's color in the scatterplot. Once the selection is changed, the points are re-rendered using the new encoding and the legend is changed to reflect the name and value range for the new variable. In Figure 37, the points are color-coded by each car's *Horsepower* value. Although the variable name is not fully visible in the table (we initially display each column at a uniform width, which tends to truncate many of the names), you can immediately tell which column has been selected for color-coding because the background coloring of the cells in that column makes it stand out.

Media Set

If media variables are present in any of the columns of the input data table, a *Media Set* button will appear to the right of the *Point Color* button. Media files are broadly defined. We currently support viewing various image formats, videos, and STLs (geometry files for a 3D printer). Each media variable in the input data table consists of a column with a shared file type, though blank entries are permitted within a column (i.e. not all ensemble members are required to have an image or video for media viewing to be used). Typically, the files in a column share a common theme, such as images showing the final state of a simulation, or an event-triggered animation, or the geometry of a specific part at a specific time. In all cases, a media file is described by a URI that provides a full path to a source file. Each URI must have the format: file://machine_name/absolute_directory_path/filename.ext.

Initially, media retrieval is disabled because the *Media Set* selection is set to *None*. Once a media variable has been selected, hovering over a scatterplot point will retrieve that point's remote media file as defined by the URI. For the first remote access, authentication will be required. Hovering provides a convenient means for rapidly examining and comparing outputs generated by in situ visualization codes, such as Catalyst and SpyPlot. Viewers can be repositioned within the scatterplot by dragging. Each viewer is connected to its associated point in the scatterplot by a line, one end of which always tracks the viewer position (see Figure 38). For videos and STLs, the viewers include interaction controls (e.g. play buttons, or rotation axis selectors).



Index	simulD	ajo	bjo	njo	beta	ndrf_last	ndhf_l...	Image1	Image2	Image3	Image4	Image5	Image6	Image7
0	1	807300000	2626830000	0.3069	0.9215	0.018044	0.109896	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gp
1	2	867100000	2626830000	0.3069	0.9215	0.012785	0.117359	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gp
2	3	926900000	2626830000	0.3069	0.9215	0.007652	0.124668	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gp
3	4	986700000	2626830000	0.3069	0.9215	0.002721	0.13184	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gp
4	5	807300000	2821410000	0.3069	0.9215	0.018044	0.109896	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gp
5	6	867100000	2821410000	0.3069	0.9215	0.012785	0.117359	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gp

Figure 38: *Media Set* Selection of *Image1*.

As with *Point Color*, there are two mechanisms for selecting which *Media Set* variable to display. Either select a variable in the *Media Set* dropdown list, or click on the small square to the right of the variable name in the *Variable Table* column header. The two mechanisms are linked, so regardless of which one is used, the selection is shown in both. As seen in Figure 38, the selected variable name (*Image1*) is highlighted in the dropdown, and the square to the right of the variable name is darkened in the column header. Once the selection is changed, hover will reference the new column's URIs when retrieving remote files, but currently visible images will not be affected. This allows you to compare different media variables from one or more simulations. Selecting *None* (the first choice in the dropdown list) disables the hover response.

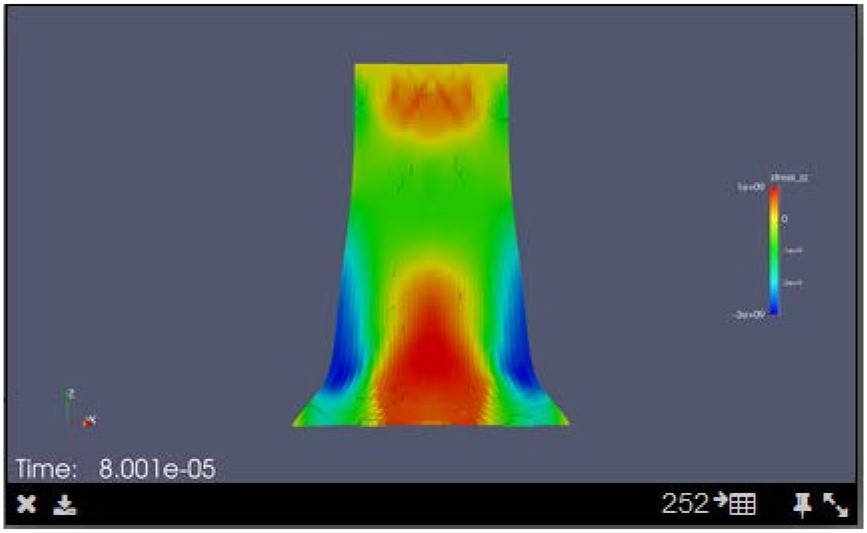


Figure 39: Image viewer with close, download, index row, pinning, and resizing icons.

The viewer in Figure 39 shows the set of standard viewer icons in the strip across the bottom of the image. From left to right, each viewer has a close icon ☒, a download icon ⤓, an index row icon ▤, a pinning icon 📌, and a resize icon ⤡. Viewers remain visible while the mouse remains within the viewer boundaries. Either clicking on the 📌 icon, resizing, playing a movie, or moving the viewer acts to *pin* the viewer in the scatterplot. A *pinned* viewer remains visible until your explicitly close it, either by clicking its ☒ icon, or by simultaneously closing all pinned views using the *Close All Pins* button (see Figure 28). To simultaneously retrieve and pin media for a group of points, first select the points, then select *pin* in the *Selection Action* dropdown list. Using the 📌 icon within a viewer both *pins* and shrinks it to a standard thumbnail size. For arbitrary viewer sizing, press and hold the left mouse button on the ⤡ icon while dragging the corner. To download a copy of the media object to your local machine, click the ⤓ icon.

Note that Slycat™ video functionality is not available for all movie formats. In fact, due to technical issues related to our web-based delivery, videos must be created in a very specific way to be viewable in Slycat™.

Video Source Files

There is no standardized support for videos between browsers. We have found that the h264 codec in combination with an mp4 container format is compatible with Firefox, Chrome, and Safari on both Windows and Mac platforms. In our testing, we have found that initial key frames are frequently lost, rendering the following compressed frames useless. This requires explicit key frame forcing during movie creation. We use the ffmpeg utility to convert images into videos. Make sure that your version of ffmpeg was built with the h264 library, since some versions of ffmpeg don't include this codec by default.

If you are within Sandia, we provide this custom version of the library on the cluster machines. You can generate Slycat™ compatible movies as follows:

```
> module load slycat
```

If your images are PNGs, they must be first converted to JPG format (ffmpeg won't complain about the input images being PNG, but the movie that it generates won't play). If you already have JPG images, skip this step:

```
> mogrify -format jpg myImageName.0*
```

This last step generates the mp4. Don't forget to enclose the image path in single quotes:

```
>  ffmpeg -pattern_type glob -i '/someDisk/someUser/someDirectoryPath/myImageName.0*.jpg' -force_key_frames 0.0,0.04,0.08 myMovieName.mp4
```

Automatic Scaling

The auto-scale button ⧉ (to the right of *Media Set*) enables/disables automatic scaling of the scatterplot axes when points are hidden or filtered out (these two mechanisms are mutually exclusive, see *Selection Action* and *Filter*). As points are removed, the locations of the remaining points shift to span the available space and the axes are redrawn to reflect the reduced value range. The mapping between point colors and the values shown in the legend is also modified to reflect the reduced value range (i.e. the new maximum value is now mapped to the brightest red and the new minimum value maps to the deepest blue, maximizing the color distinctions between the remaining points). Compare the scatterplots in Figure 40 and Figure 41 to see the effects of using auto-scaling.

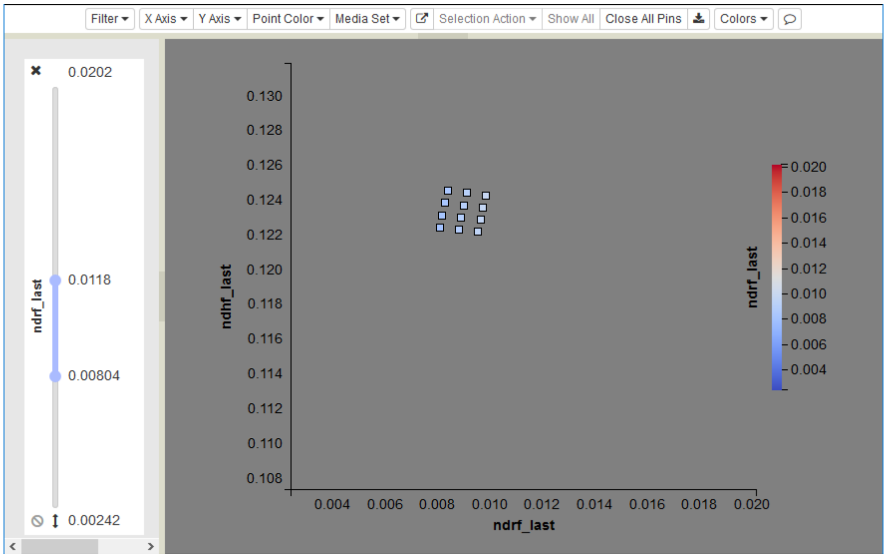


Figure 40: Filtered scatterplot with auto-scaling off. Note that most of the scatterplot is empty.

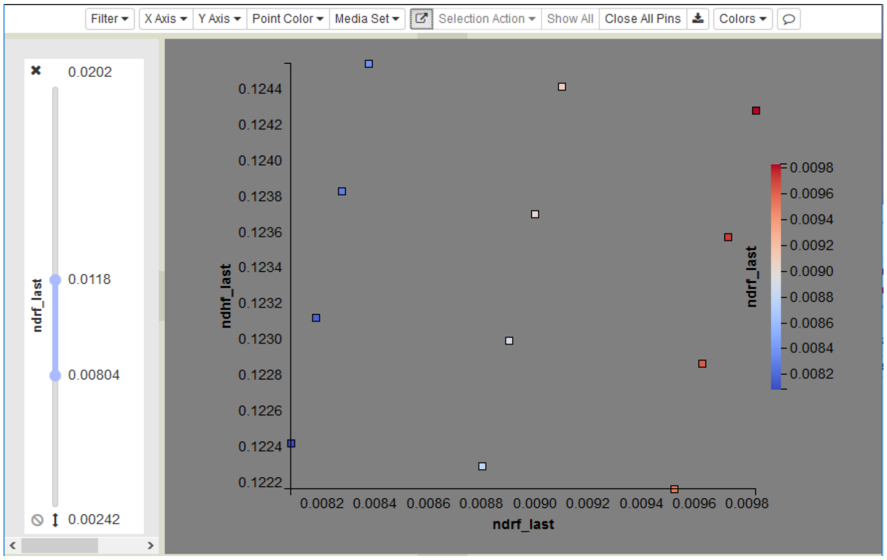


Figure 41: The same filtered scatterplot with auto-scaling enabled. The color range is scaled as well.

This feature is especially useful if a variable's value range is skewed by a few anomalous points. For example, imagine a data set where most points have x-coordinates in the range from 0 to 1, but a few points have extreme values of 10,000. Including the extreme points means that most of the plot consists of empty space, while the majority of points are drawn on top of one another near the origin. By filtering and rescaling, you can remove the extreme points and have the remaining points fill the plot.

Note that automatic scaling is enabled as the default, indicated by the darkened state of the button (Figure 41).

Selection Action

Selected points (see Selecting Points) create a set that is used as the designated input for any of the group operations listed in the *Selection Action* dropdown, as shown in Figure 42. Since filtering and hide/show selections are mutually exclusive within the Slycat™ interface, if there are any filters enabled (visible), all of these actions, except *Pin,* will be disabled.

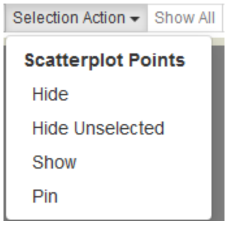


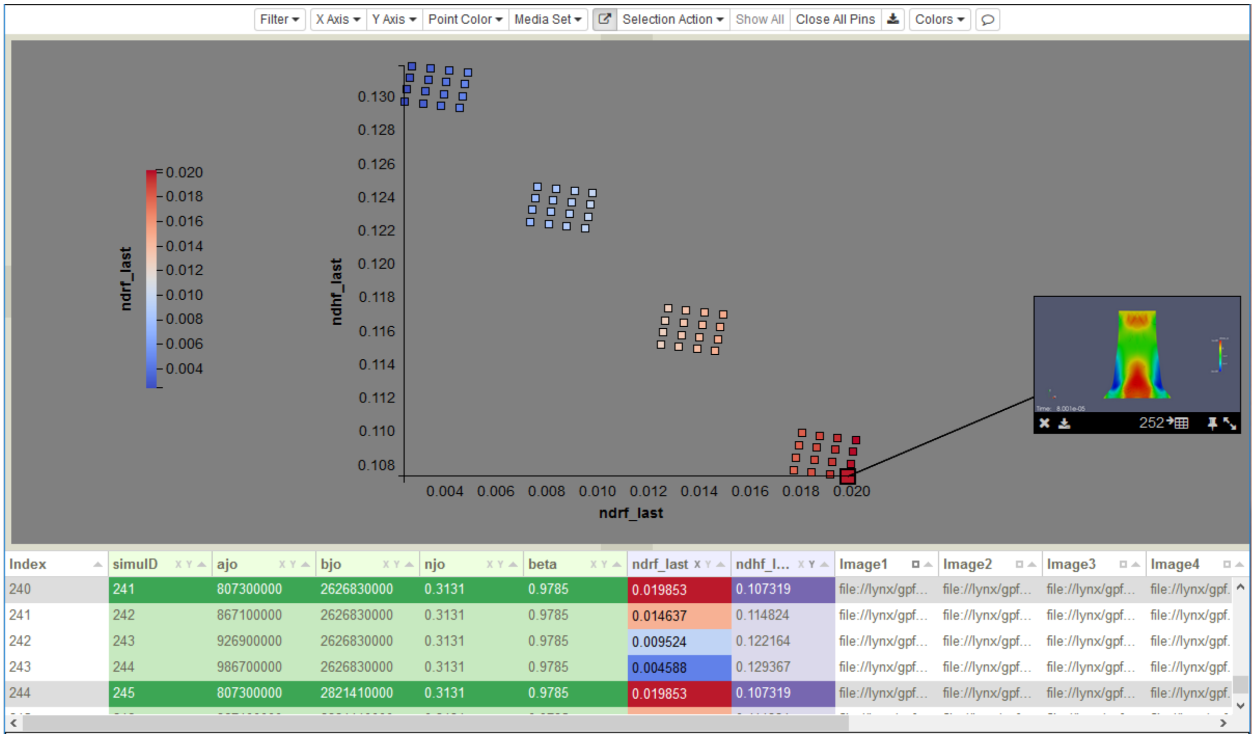
Figure 42: *Selection Action* dropdown list of operations on selected groups of points.

Hide removes the selected points from the scatterplot and yellows out the associated rows from the table. The points are still selected, but they are no longer visible, which is why their rows are colored yellow instead of white. If automatic scaling is enabled (see Automatic Scaling), the remaining visible points in the scatterplot will be shifted and recolored to reflect any changes in the value range.

Hide Unselected performs the same operation on the set of points that are not selected, thereby reducing the visible points to just the selected set.

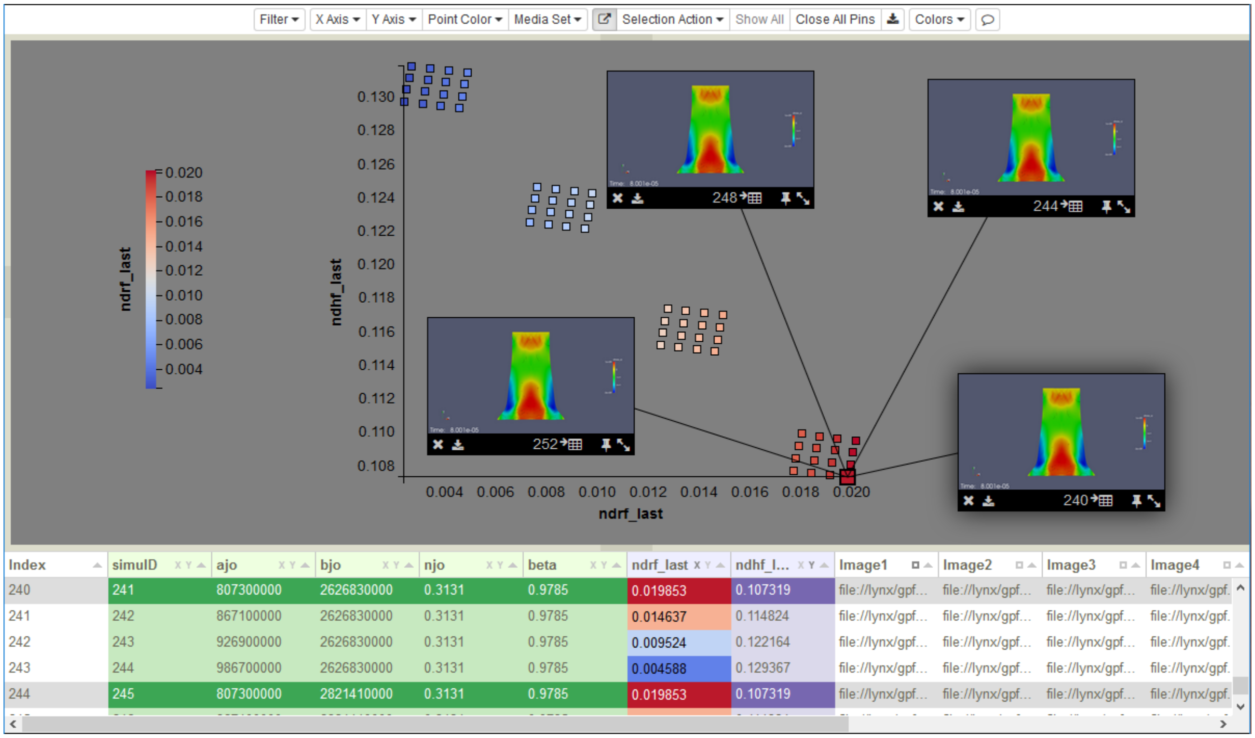
Show restores the last hidden selection to visibility, both in the scatterplot and in the table. If the selection has been lost by clicking elsewhere within the scatterplot, *Show* will not be able to restore the previously hidden set, since the selected set is now empty. For the same reason, there is not a function to restore points hidden using *Hide Unselected*. However, the *Show All* button (to the right of the *Selection Action* dropdown) can be used to make all hidden points visible again.

Using the currently selected media variable, *Pin* retrieves and pins items for each of the points in the selection set (only if visible), performing the equivalent of a group hover and pin operation. This is much more efficient than doing individual retrieval when there are large numbers of runs to compare. Note that the images may be stacked on top of one another if the associated points are coincident, as is the case in Figure 43. However, you can separate the images by dragging them apart, as shown in Figure 44.



Index	simuID	ajo	bjo	njo	beta	ndrf_last	ndhf_l...	Image1	Image2	Image3	Image4
240	241	807300000	2626830000	0.3131	0.9785	0.019853	0.107319	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
241	242	867100000	2626830000	0.3131	0.9785	0.014637	0.114824	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
242	243	926900000	2626830000	0.3131	0.9785	0.009524	0.122164	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
243	244	986700000	2626830000	0.3131	0.9785	0.004588	0.129367	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
244	245	807300000	2821410000	0.3131	0.9785	0.019853	0.107319	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.

Figure 43: Four selected runs are coincident in the scatterplot, so the associated pinned images are stacked.



Index	simuID	ajo	bjo	njo	beta	ndrf_last	ndhf_l...	Image1	Image2	Image3	Image4
240	241	807300000	2626830000	0.3131	0.9785	0.019853	0.107319	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
241	242	867100000	2626830000	0.3131	0.9785	0.014637	0.114824	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
242	243	926900000	2626830000	0.3131	0.9785	0.009524	0.122164	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
243	244	986700000	2626830000	0.3131	0.9785	0.004588	0.129367	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.
244	245	807300000	2821410000	0.3131	0.9785	0.019853	0.107319	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf...	file://lynx/gpf.

Figure 44: Separating pinned images by dragging. Note that the legend can also be repositioned by dragging.

Show All

The *Show All* button restores all points that were hidden using *Selection Action* to visibility. *Show All* is disabled whenever a filter is present, so it cannot be used to make filtered points visible again.

Close All Pins

The *Close All Pins* button closes all pinned media in the view. It provides a quick method to clear a Parameter Space model of all viewers.

Video Synchronization 📹

Beyond the video functionality described earlier (see the Media Set section), Slycat™ provides additional fine-grained and group-based video controls. Once the first video is pinned, the interface shown in Figure 45 will appear in the model-specific controls to right of the Download Data Table ⭳ icon. These video controls will remain visible until all videos are closed. For a single video, these global controls provide single step accuracy in advancing or rewinding the animation, which is not possible using the limited controls of an individual viewer. However, the larger goal of this interface is to enable synchronized playback of multiple videos from a single set of controls.

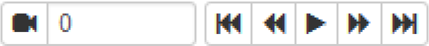


Figure 45: Video controls: enable video synch, current video location (seconds from start), go to first frame, step back a frame, play, step forward a frame, and go to last frame, respectively.

From left to right the controls are as follows: the video synchronization button 📹, a numeric field providing the current video location in seconds from the video start, a button to go to the start of the video ⏮, a button to step backward by one frame ⏪, play ▶ / pause ⏸ buttons (the icon changes to pause once play is pressed), a button to step forward by one frame ⏩, and a button to go to the end of the video ⏭.

The video synch button is a toggle that enables/disables shared control of multiple videos. The background color of the icon shows the state of the synchronization. The background is gray when it is enabled 📹, and white when it is disabled 📹. When video is synched, the playback buttons operate on all pinned videos. When synch is disabled, the playback operates only on the *current* video. The current video is highlighted by drawing a shadow behind it, making it appear to float above the other videos, such as the middle video in Figure 46. At all times, the video location field shows the current value of the video's elapsed playback time (note that this is not the same as the simulation time stamp for a particular frame). You can directly edit this field to align all videos to the frame that is closest to a specific time of interest in the playback.

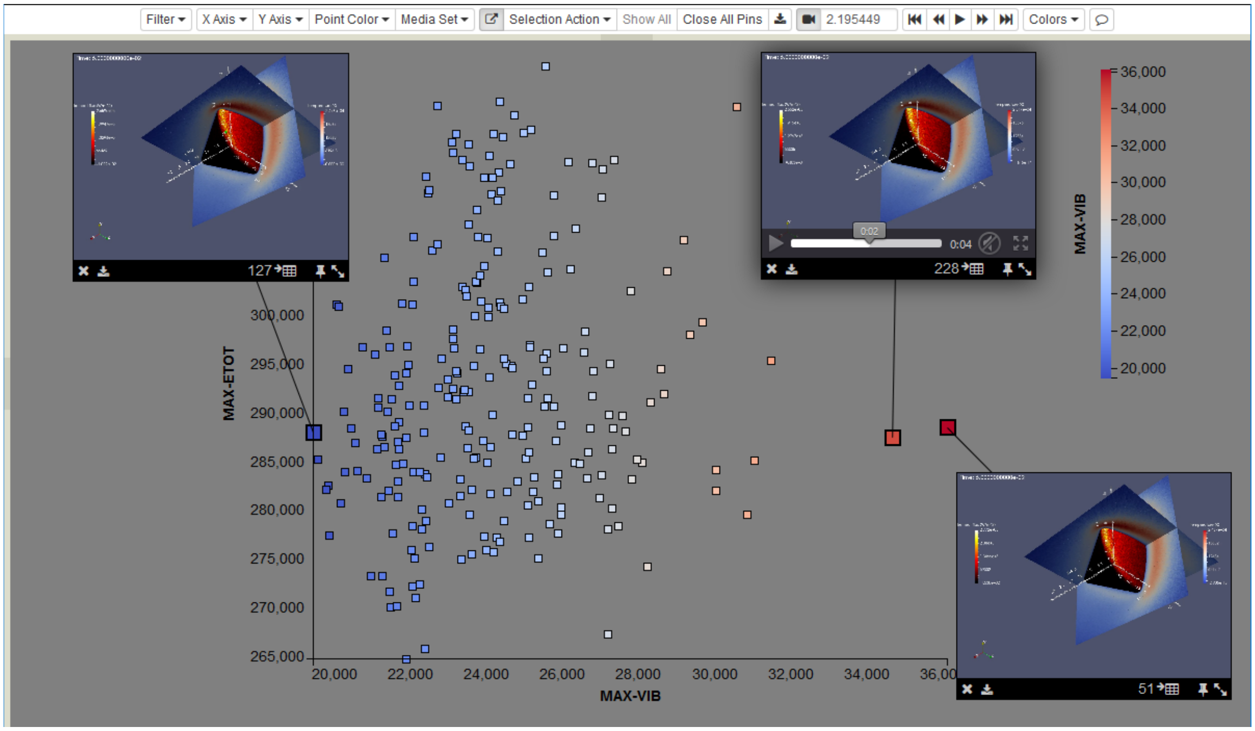
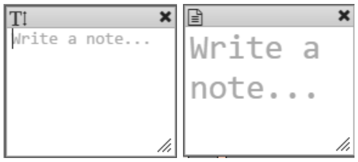


Figure 46: Three synched videos, where the middle video is the *current* video.

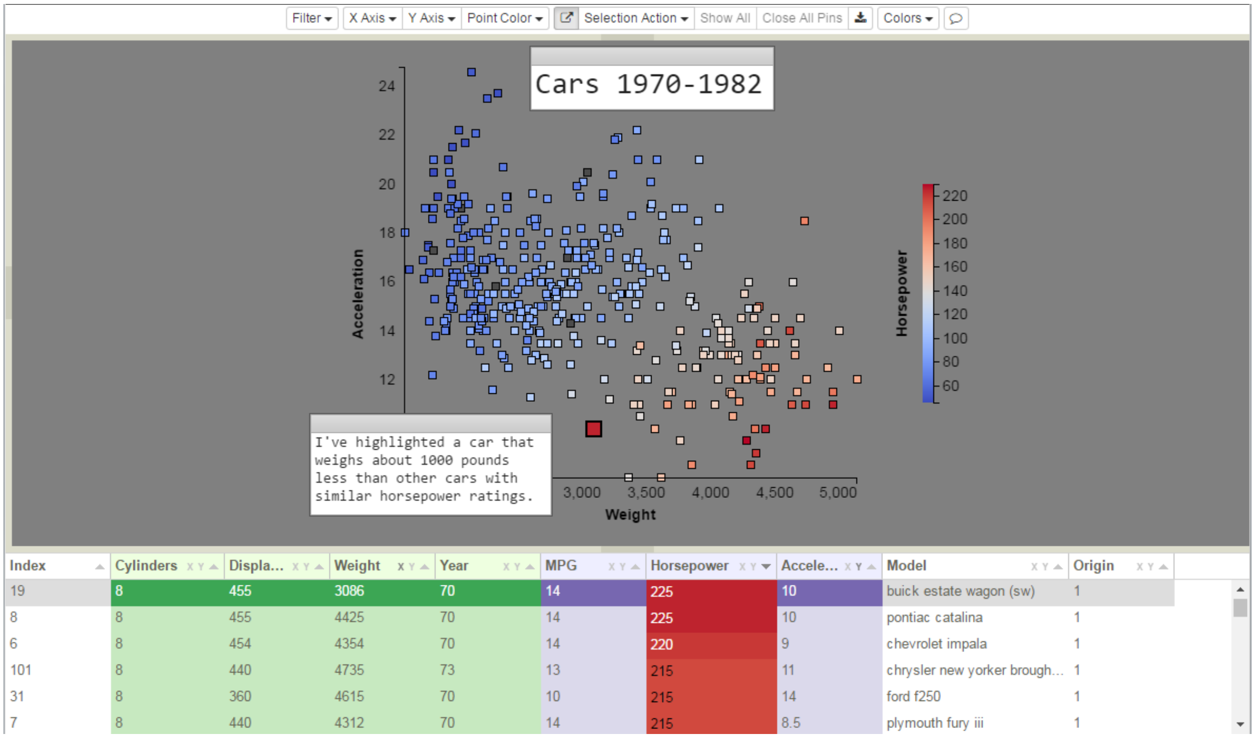
Adding Text 🗨

To add text, click the note icon 🗨 on the right end of the controls. A text window will popup. The intent was to provide two types of annotation: notes and titles. The icon in the upper left toggles between TI and 🗎 to switch between text sizes, as shown in Figure 47. The TI icon increases the text size to generate a title. The 🗎 icon decreases the text size back to a font more suitable for notes. The text window can be closed and resized using the ☒ and ◿ icons, respectively. The icons are only visible when the mouse is in the window. Figure 48 provides an example showing both types of annotation. Notes in combination with bookmarks can be used to draw the attention of project members to discoveries or generate explanatory visualizations that can be self-contained.


Write a note...

Write a note...

Figure 47: Text windows for adding notes (on left) and titles (on right).



Cars 1970-1982

I've highlighted a car that weighs about 1000 pounds less than other cars with similar horsepower ratings.

Index	Cylinders	Displa...	Weight	Year	MPG	Horsepower	Accele...	Model	Origin
19	8	455	3086	70	14	225	10	buick estate wagon (sw)	1
8	8	455	4425	70	14	225	10	pontiac catalina	1
6	8	454	4354	70	14	220	9	chevrolet impala	1
101	8	440	4735	73	13	215	11	chrysler new yorker brough...	1
31	8	360	4615	70	10	215	14	ford f250	1
7	8	440	4312	70	14	215	8.5	plymouth fury iii	1

Figure 48: Example of adding text as both a title and a note.

Variable Table

The *Variable Table* is the same as the table in the CCA model (see Variable Table) with one small difference. '**X**' and '**Y**' icons to the right of variable names in the column headers provide an alternate method to the dropdown lists for selecting the scatterplot axes.

Time Series Model

The Time Series model was originally developed to evaluate similarities between waveforms generated by electrical circuit simulations. However, this model can be used more generally to compare any set of time series data, so long as the starting and ending times for each sequence of values are the same. Although the time series inputs do not need to be sampled identically, our initial step is to resample since the analysis ultimately requires corresponding samples for comparison. Points in each sequence are binned, with the bin size calculated by dividing the time range into equal intervals. The values of the points within each bin are then averaged. The underlying assumption is that there are enough samples in the sequence to have at least one sample per bin. The number of bins is a user-supplied parameter to the analysis.

Slycat™ calculates a table of the distances between each pair of resampled time series by summing the differences between corresponding points. The distance table is used to create an agglomerative, hierarchical model of similarities between the sequences. The resulting model is a tree, in which the set of time series contained within each subtree are more and more similar as successive subtrees get closer to the leaves. Because our distance metric was developed for electrical simulation data, both amplitude (y value) and timing (x value) must match for a pair of sequences to be regarded as similar (i.e. identical shapes that are shifted in time are not seen as similar).

Time Series Data

Slycat™ accepts two different time series data formats, which we will call *Xyce* and *CSV*. Each input format consists of two parts, a table file describing the entire ensemble, and a time series data file for each simulation in the ensemble. Like the CCA and Parameter Space models, the table is at the heart of the model. For each simulation (for each row in the data table), there must be a file with time series data. Within each of these time series files are sequences of values, sampling one or more output variables over the course of the simulation. It is not necessary that each simulation write the same number of samples into their time series files, but it is required that each simulation have a corresponding data file with matching output variables that cover the same time range.

The *Xyce* format consists of Xyce-generated time series files stored within a fixed directory hierarchy. The hierarchy is rooted within a single high-level directory where there must be a *dakota_tabular.dat* file (providing the data table). It is not that the file must be named *dakota_tabular.dat*, but rather the file format must correspond to the *dakota_tabular.dat* files generated by Dakota. Additionally, a set of subdirectories (one per run) must be located in the same directory as the *dakota_tabular* file. These subdirectories should all be named using a template like *workdir.n*, where *n* is the simulation number. Within each subdirectory, there must be a time series file generated by Xyce that is formatted as a *.prn* file. The time series files must all be named identically (the subdirectory defines which simulation generated them), and each file must contain a shared set of time series variables (columns with matching headers within each of the *.prn* files).

The *CSV* format (such as *heartbeat.dat* files produced by Sierra, or *.csv* outputs from Catalyst), is less structured than the *Xyce* format. The individual time series files need not be stored in the same directory hierarchy as the data table, nor does the directory structure need to follow any structure or naming conventions. Instead, the data table is a *CSV* file, which contains a column of URIs providing full paths to each of the time series files, which must also be *CSV* files (no *.prn* files). Each URI must have the format: file://machine/absolute_directory_path/timeseries_filename.csv.

Whether we are using *.prn* files or *CSV* files, both formats are essentially tables in which each column is a separate variable and each row is a set of concurrent samples for each of the variable columns. The first line of a time series file contains headers, which provide the names of the time series output variables. Note that in a *CSV* file, we expect to see only a single row of header information consisting of the column names (some physics codes output two rows of header information, with the variable names in the first row and the units in the second row – this is not a legal *CSV* format). At least one column must be a time value (typically the first column).

If your data is not currently in one of these two formats, Excel can be used to create *CSV* files from most common table formats. Note that if output metrics have been created separately in a post-processing step, they will need to be integrated with the inputs to form a single file prior to model creation.

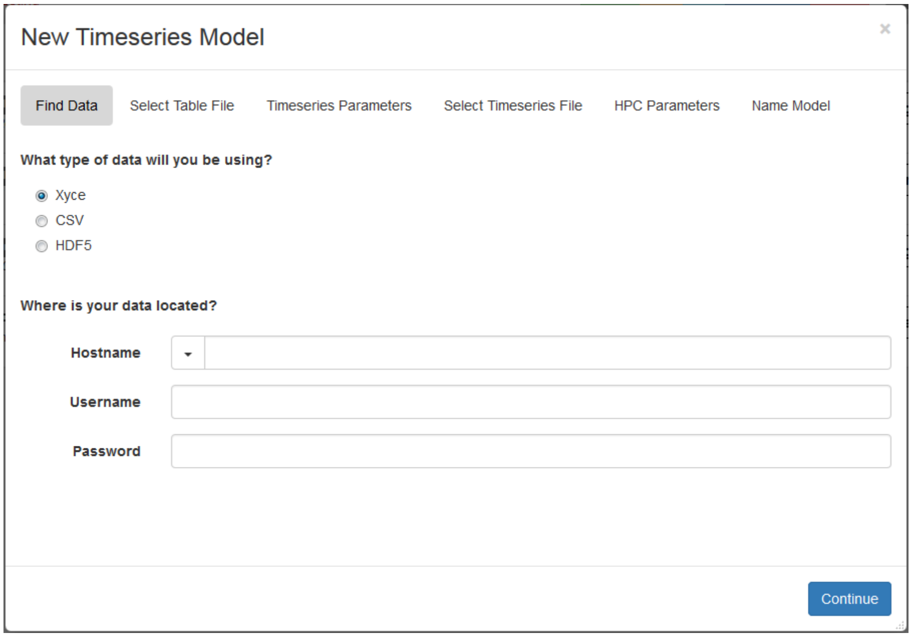
In the time series creation wizard, both formats are rewritten as *HDF5* files in a temporary Slycat™ directory (we have found that this significantly speeds up our processing compared to working with the originally-formatted files). If you opt to keep these *HDF5* files, they constitute a third data format that the wizard will accept, though be aware that *HDF5* files created through other means are not interchangeable since their internal structures will be different.

Creating a Time Series Model

Creating a Time Series model is more complicated than the models that we have described in previous sections. This is due to the size and structure of the data, combined with the computationally intensive nature of the analysis. The data is stored in multiple files, typically in multiple directories. This data complexity and scale compels the use of parallel processing to reduce the model creation time. Unfortunately, our cluster's batch environment increases complexity through the need for additional High Performance Computing (HPC) parameter choices, uncertain wait times in the job queue, and potentially long processing times. All these factors are at odds with an interactive interface. Consequently, our time series wizard is designed to collect all the necessary information, then autonomously launch the analysis and finish the model creation. You are free to do other things while it completes, although we do provide a means to remotely check on the status of your job through the Slycat™ interface.

To access the wizard, go to your project page, click on the green *Create* button and select *New Timeseries Model* from the dropdown list. A dialog for walking you through the process will then pop up, as shown in Figure 49. The first page identifies the format of the time series data (see Time Series Data above) and the location of the ensemble's table file. The assumption is that time series data is large and difficult to move, so it will be located on the same remote HPC machine where it was generated. Consequently, we do not provide a *Local* option, as we do for other model types.

Using the radio buttons, select the data input type. Next, select a remote host from the *Hostname* dropdown. Unlike CCA or Parameter Space, only hosts included in the dropdown list may be used. This is because time series analysis requires parallel job launching functionality found in the Slycat™ agent, which must be running on the remote machine. Although the Slycat™ agent assists you in remotely submitting the analysis job to the cluster queue, you will be running the job under your own user credentials. Consequently, the host must be a machine on which you already have a user account. Enter your username and password into the associated fields. Hitting the *Enter* button after typing in your password will both log you into the remote machine and take you to the next screen in the wizard. If you have recently accessed the selected host through Slycat™, the *Username* and *Password* fields will not be shown. This is because Slycat™ maintains remote sessions for a fixed period of time after your initial login to reduce the number of login requests. In this case, click the *Continue* button to advance the wizard.



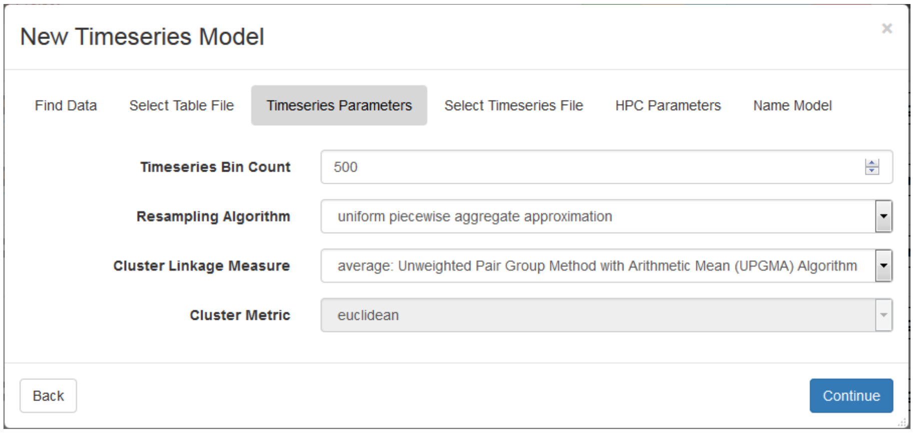
What type of data will you be using?

● Xyce
○ CSV
○ HDF5

Where is your data located?

Figure 49: Popup dialog in Timeseries model creation wizard.

The next screen of the wizard will depend on which data format you selected. If you selected *Xyce* or *CSV*, the next screen will be a file browser on the remote host. Navigate to the location of the ensemble data table (a *dakota_tabular* file within the directory hierarchy described above for *Xyce* inputs, or a *CSV* file containing full paths to each time series file for *CSV* inputs). Navigation is identical to that described in the earlier section on Remote Files. Click on the data table file in the remote file browser to select it, then click *Continue*. If you selected *HDF5,* the wizard skips this step since there is no need to select a table file.



Timeseries Bin Count: 500

Resampling Algorithm: uniform piecewise aggregate approximation

Cluster Linkage Measure: average: Unweighted Pair Group Method with Arithmetic Mean (UPGMA) Algorithm

Cluster Metric: euclidean

Figure 50: Timeseries Parameters for Xyce data sets.

For all three input types, the next step is setting the parameters to be used for binning and clustering the time series. *Xyce*, *CSV*, and *HDF5* have slightly different interfaces for this step, which are shown in Figure 50, Figure 51, and Figure 52, respectively.

The *CSV* screen includes two additional fields that are not needed by the other formats, *Table File Delimiter* and *Timeseries Column Name*. *Table File Delimiter* allows you to use other delimiters besides commas in the data table, such as tabs or spaces. Tabs are difficult to specify because the web interface uses tabs to move between fields, but if you cut-and-paste a tab into the field, enclosing it with single quotes, Slycat™ will accept a tab delimited table. To designate a *space* as a delimiter, enclose it with single quotes, since otherwise the field is interpreted as being empty. Commas do not require quotes.

Since *CSV* data tables can have multiple columns of time series data (e.g. if you sampled a set of variables over time at various locations within the simulation), the *Timeseries Column Name* identifies which time series data set to analyze. Type in the column name, taking care to exactly match the header as it appears in the table.

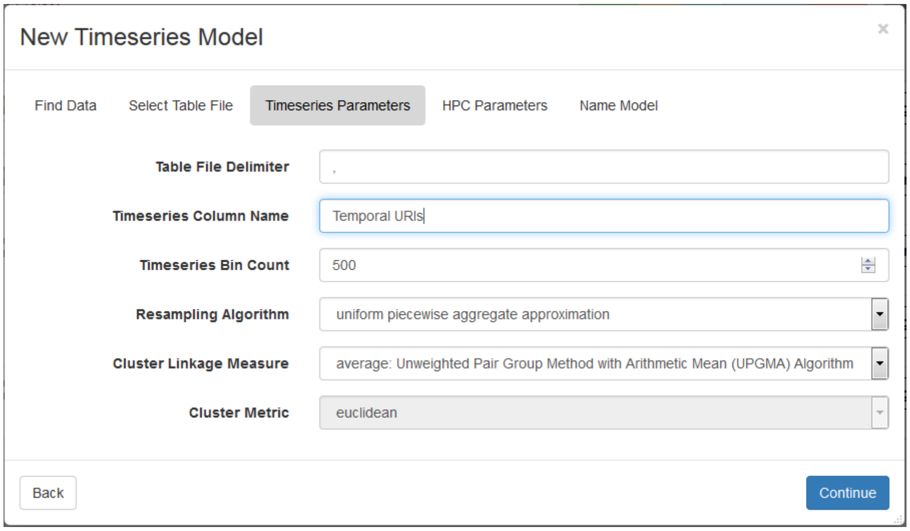


Table File Delimiter: ,

Timeseries Column Name: Temporal URIs

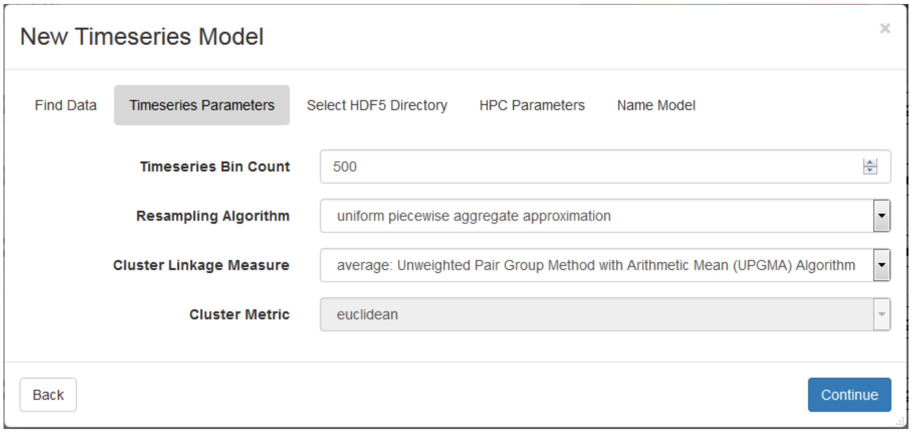
Timeseries Bin Count: 500

Resampling Algorithm: uniform piecewise aggregate approximation

Cluster Linkage Measure: average: Unweighted Pair Group Method with Arithmetic Mean (UPGMA) Algorithm

Cluster Metric: euclidean

Figure 51: Timeseries Parameters for CSV data sets.



Timeseries Bin Count: 500

Resampling Algorithm: uniform piecewise aggregate approximation

Cluster Linkage Measure: average: Unweighted Pair Group Method with Arithmetic Mean (UPGMA) Algorithm

Cluster Metric: euclidean

Figure 52: Timeseries Parameters for HDF5 data sets.

The remaining parameters are shared by all three input types. *Timeseries Bin Count* controls how finely the time series is sampled. The resulting binned sequences are used for calculating similarities and the reduced representations are drawn in the model visualization. Generally, bin counts between 500 and 1000 produce a reasonable tradeoff between speed and accuracy. Although increasing the number of bins increases both the analysis and rendering times, a greater bin count also helps preserve spikes or other localized features that could be lost when using a smaller number.

The *Resampling Algorithm* dropdown has two options. Both algorithms use a uniform set of bins, with the choice between using *uniform piecewise linear approximation* or *uniform piecewise aggregate approximation* as the resampling method. *Uniform piecewise aggregate approximation* is the default.

The *Cluster Linkage Measure* dropdown selects the metric used when evaluating distance between groups of elements. There are four choices:

- *single: Nearest Point Algorithm*
- *complete: Farthest Point Algorithm*
- *average: Unweighted Pair Group Method with Arithmetic Mean (UPGMA) Algorithm*
- *weighted: Weighted Pair Group Method with Arithmetic Mean (WPGMA) Algorithm*

Single evaluates the distance using the closest elements/minimum linkage; *complete* uses the farthest elements/maximum linkage; *average* uses the distance between the group averages; and *weighted* uses the values from the distance matrix. *Average* is the default. We are using SciPy to perform the clustering, so a more complete description of the linkage choices can be found at https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.linkage.html.

The *Cluster Metric* currently only has a single choice, *Euclidean*, so it cannot be changed (hence the field is grayed out). This field is provided to inform you that we are using Euclidean distances in our algorithms.

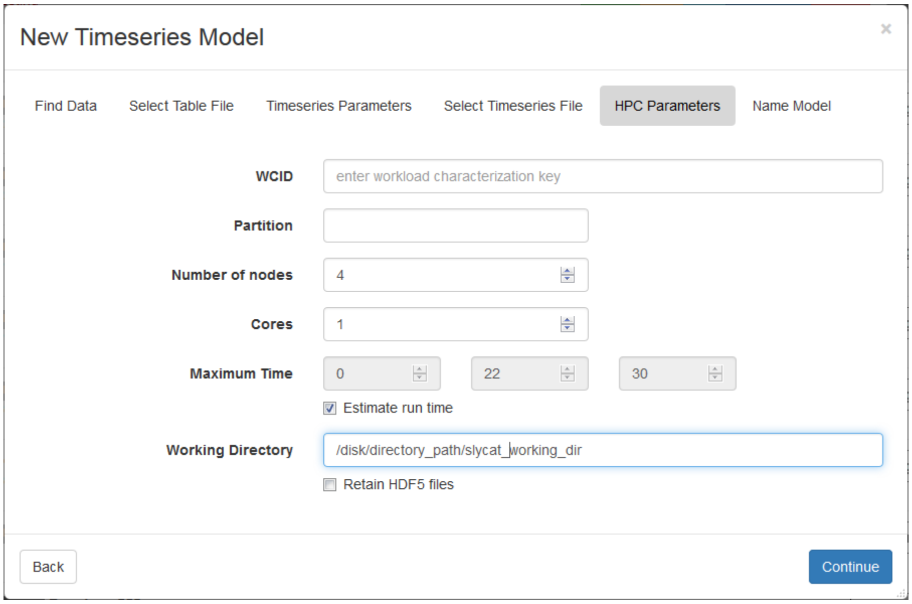
Once you are satisfied with these parameter choices, click *Continue* to go to the next screen.

For *Xyce* data sets, the following screen, *Select Timeseries File*, is a file browser on the remote host. Navigate to one of the time series files within any of the working directories and select it. This provides a template for the file name and path that is used to access all the other time series files in the data set. Click *Continue* to advance to the next screen.

For *HDF5* data, the following screen, *Select HDF5 Directory*, also is a file browser on the remote host. However, instead of selecting a time series file, you should navigate into the directory where Slycat™ previously created and stored HDF5 files for a Time Series model (i.e. you should be looking at the contents of the directory and be able to see the HDF5 files). Click *Continue* to advance to the next screen.

The *HPC Parameters* screen is specific to your institution. Figure 53 shows the parameters for Sandia's cluster systems. Other than differences in the list of steps shown along the top, the same screen is used for all three data formats (*Xyce*, *CSV*, and *HDF5*).

WCID stands for Workload Characterization ID, which is required for job submission on the clusters. Because Slycat™ uses parallel processing to speed up Time Series model creation, users are required to have obtained a *WCID* prior to creating a Time Series model. Your *WCID* is associated with an Strategic Management Unit/Program Management Unit (SMU/PMU), which is used to specify the *Partition* or queue-name. At Sandia, the choices are *nw*, *ec*, *dsa*, *ihns*, *ldrd*, *cee*, *viz*, or *viz batch*).



WCID: enter workload characterization key

Partition:

Number of nodes: 4

Cores: 1

Maximum Time: 0 22 30

☑ Estimate run time

Working Directory: /disk/directory_path/slycat_working_dir

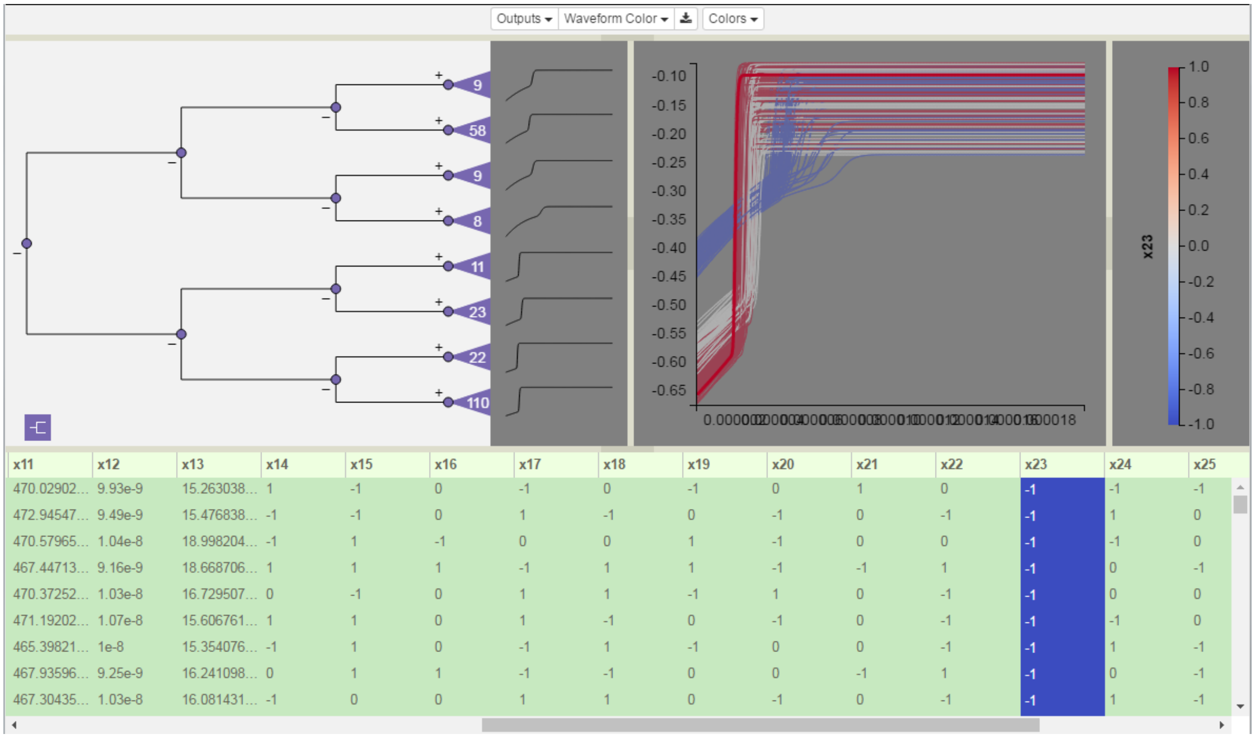
☐ Retain HDF5 files

Figure 53: HPC Parameters for Sandia clusters.

Time Series Model Visualization

Our examples in this section come from an ensemble of 250 electrical circuit simulations, where our time series outputs are current and voltage variables sampled over time. A Time Series model of this ensemble is shown in Figure 54. The model provides three linked views, each providing a different level of abstraction. The *Dendrogram View* in the upper left provides a high-level view that groups waveforms by similarity. In the upper right, the *Simulation View* displays a line plot for each ensemble member. The plots are superimposed in a shared coordinate space to facilitate comparisons. The lowest level view is the *Variable Table*, which provides the raw data values from the original table file. It is drawn across the bottom of the display and may be scrolled both vertically and horizontally if the number of rows or columns exceeds the available space. Although the *Variable Table* may contain columns with scalar output metrics, or columns of file URIs that would be categorized as being neither input nor output, the ingestion wizard assumes all table columns are inputs and colors them green.

The *Dendrogram View* controls visibility of ensemble members in both the *Simulation View* and the *Variable Table*. Selections can be made in any one of the views, after which they are propagated to the other two. Color-encoding is shared by all views.



x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	x21	x22	x23	x24	x25
470.02902...	9.93e-9	15.263038...	1	-1	0	-1	0	-1	0	1	0	-1	-1	-1
472.94547...	9.49e-9	15.476838...	-1	-1	0	1	-1	0	-1	0	-1	-1	1	0
470.57965...	1.04e-8	18.998204...	-1	1	-1	0	0	1	-1	0	0	-1	-1	0
467.44713...	9.16e-9	18.668706...	1	1	1	-1	1	1	-1	-1	1	-1	0	-1
470.37252...	1.03e-8	16.729507...	0	-1	0	1	1	-1	1	0	-1	-1	0	0
471.19202...	1.07e-8	15.606761...	1	1	0	1	-1	0	-1	0	-1	-1	-1	0
465.39821...	1e-8	15.354076...	-1	1	0	-1	1	-1	0	0	-1	-1	1	-1
467.93596...	9.25e-9	16.241098...	0	1	1	-1	-1	0	0	-1	1	-1	0	-1
467.30435...	1.03e-8	16.081431...	-1	0	0	1	1	0	-1	0	-1	-1	1	-1

Figure 54: Time Series model with *Dendrogram View* in upper left, *Simulation View* in upper right, and *Variable Table* below.

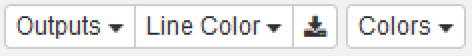


Figure 55: Time Series model specific controls. The *Outputs* dropdown switches the views between different time series variables. *Line Color* selects from the scalar input variables for color-coding the lines. The *Download Table* icon enables download of the entire *Variable Table* or subsets thereof. *Colors* selects the color theme. The latter two functions are shared by all model types.

The model specific controls for a Time Series model are shown in Figure 55. Within each time series file, multiple output variables may be present. Each variable is input as a column of values, and each row provides the values for all variables at a single instant in time. Each time series variable generates a distinctly different dendrogram and has a set of unique plots. The *Outputs* dropdown provides a list of the time series variables so you can select which one is currently being displayed in the *Dendrogram View* and the *Simulation View*.

To facilitate discovering relationships between input parameters and groups of output plots, it is often useful to color-code the lines by input variable values. Select a color-coding variable either through the *Line Color* dropdown or by clicking on a table column header (see Color-Coding Lines below).

Dendrogram View

The *Dendrogram View* displays a tree that clusters line plots from a single temporal variable by similarity. The analysis begins by calculating distances between each pair of time series vectors, an O(n^2) calculation. Then the distance matrix is used to build the dendrogram using agglomerative clustering. Each time series output variable generates a different tree.

The dendrogram is drawn with the root on the left and the leaves on the right. To reduce visual clutter, the tree is not drawn at full resolution at every level down to the leaves. Instead, only the first four levels of the tree are initially rendered (as shown in Figure 56 below), with the last level on the right consisting of collapsed subtrees for the remaining sections of the tree down to the leaves. The subtrees are represented by purple triangular icons, each labeled with the number of nodes in its subtree. Non-collapsed nodes in the tree are drawn as purple dots.

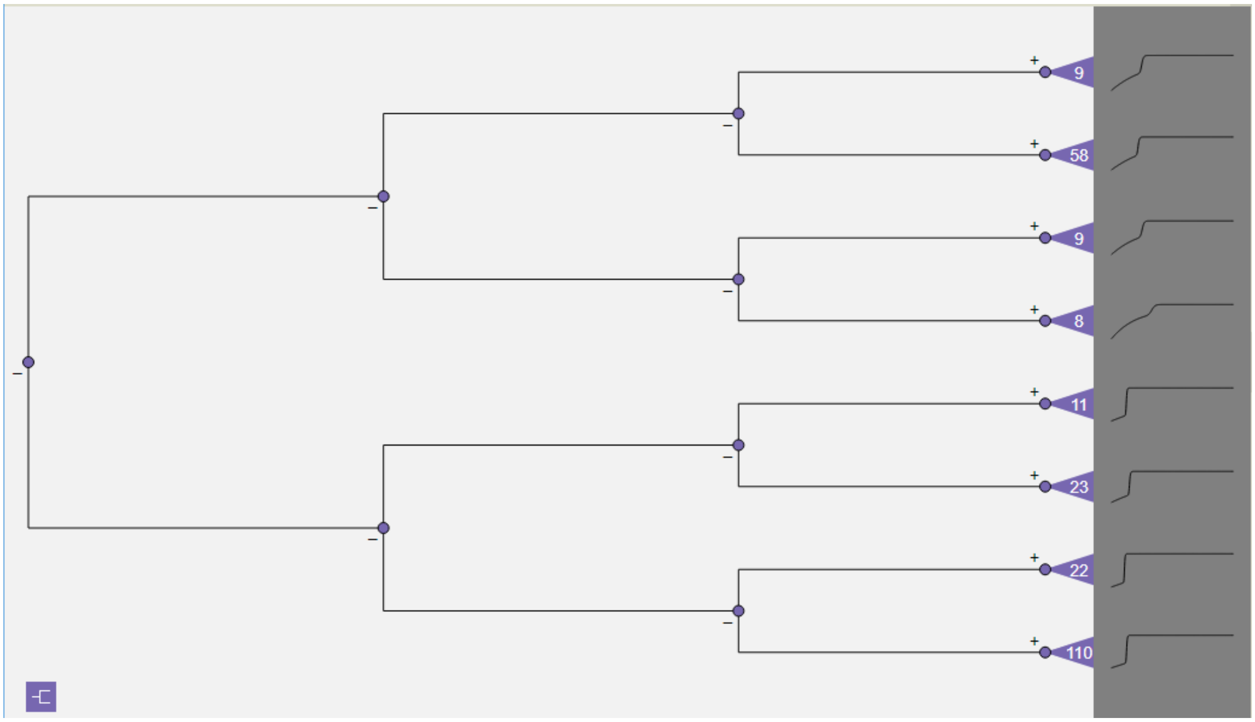


Figure 56: Dendrogram tree compression.

The expansion or contraction of each node is individually controlled through the '+' or '-' icons that appear to the left of each node, respectively above or below the line connecting the node to its parent. In Figure 57, the subtree with 58 nodes in Figure 56 has been expanded by clicking the '+'. Each expansion adds two levels of the tree to the dendrogram. Note that leaf-level nodes are drawn as dots.

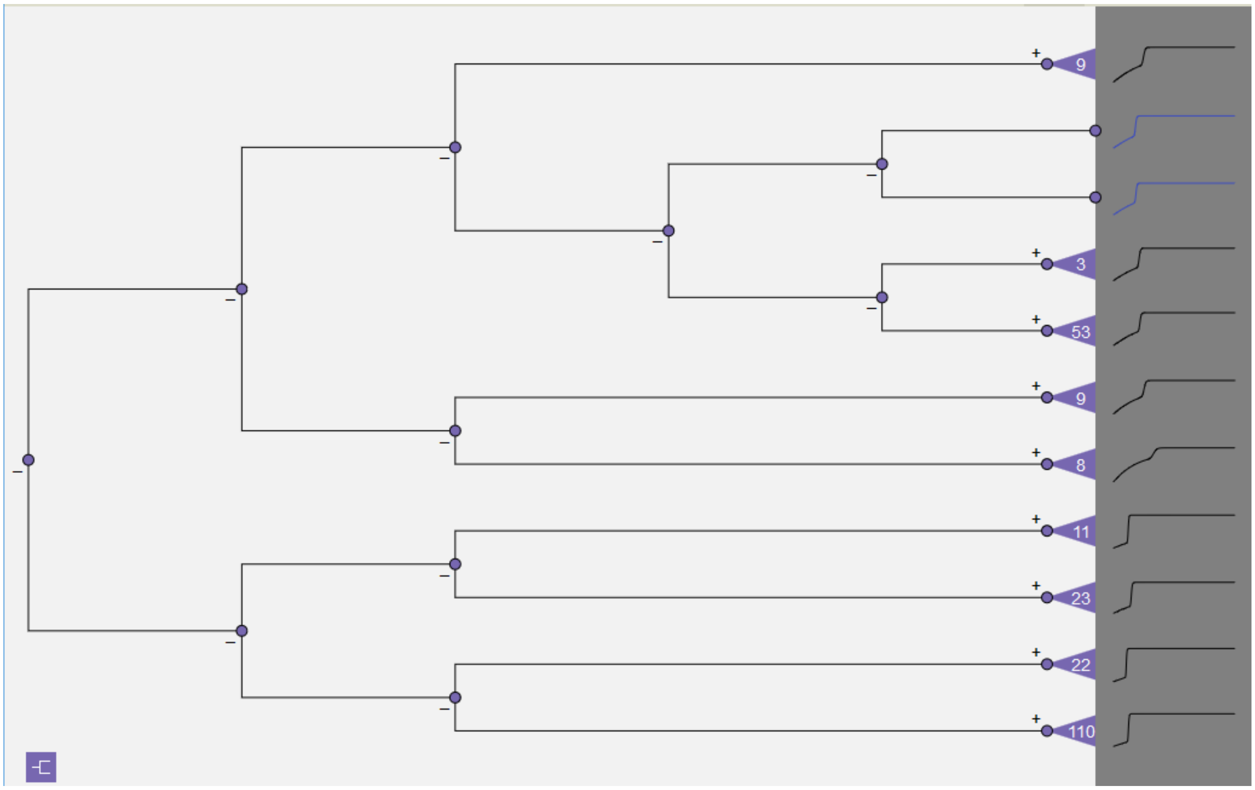


Figure 57: Expansion of second subtree from the top (previously shown as the subtree with 58 nodes in Figure 56). Each expansion adds two additional levels.

To expand a subtree all the way down to the leaves in a single action, click on the subtree triangle itself. In Figure 58, the subtree with 9 nodes at the top of the dendrogram in Figure 56 has been expanded to the leaf level with this one-click operation. Caution should be exercised when doing a full subtree expansion for subtrees over 20 or so nodes, since the dendrogram can become cluttered and largely unintelligible. Figure 59 demonstrates the results of clicking on the subtree with 110 nodes at the bottom of the dendrogram in Figure 56.

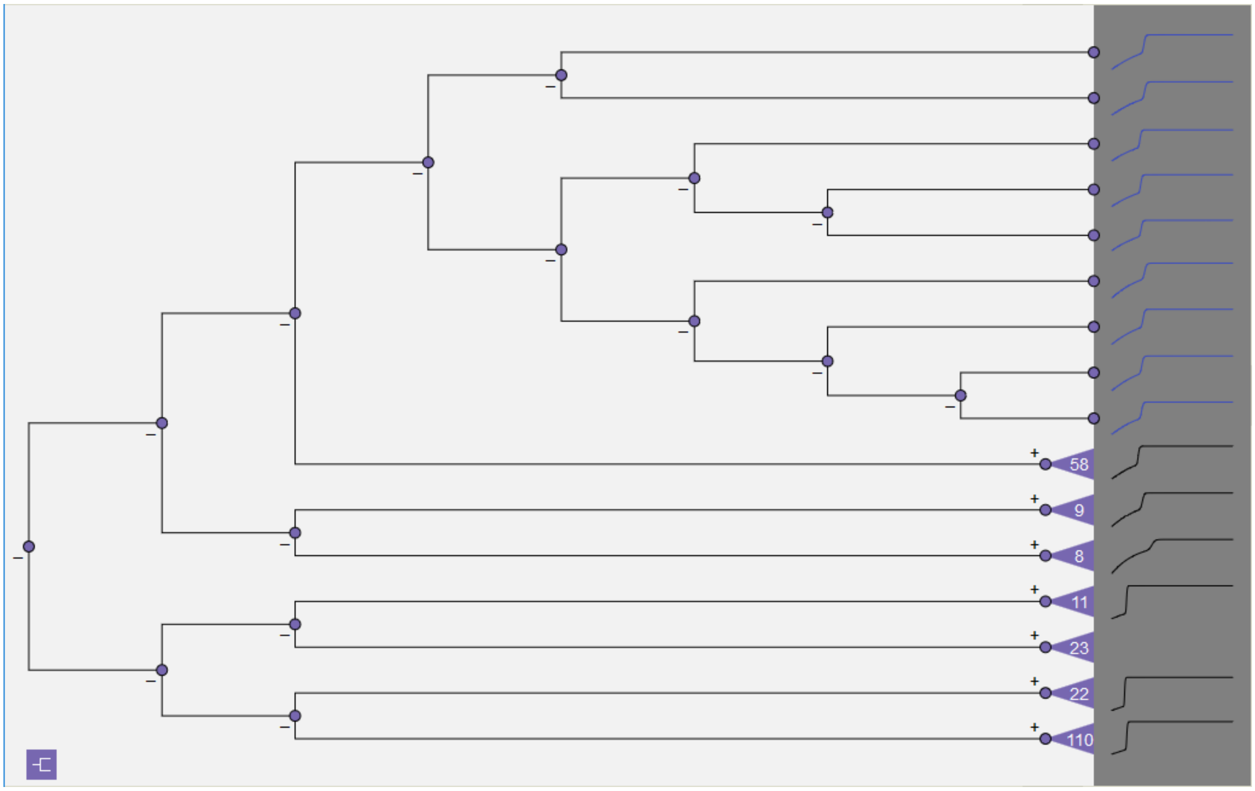


Figure 58: Subtree with 9 nodes expanded to leaf level by clicking triangular subtree icon.

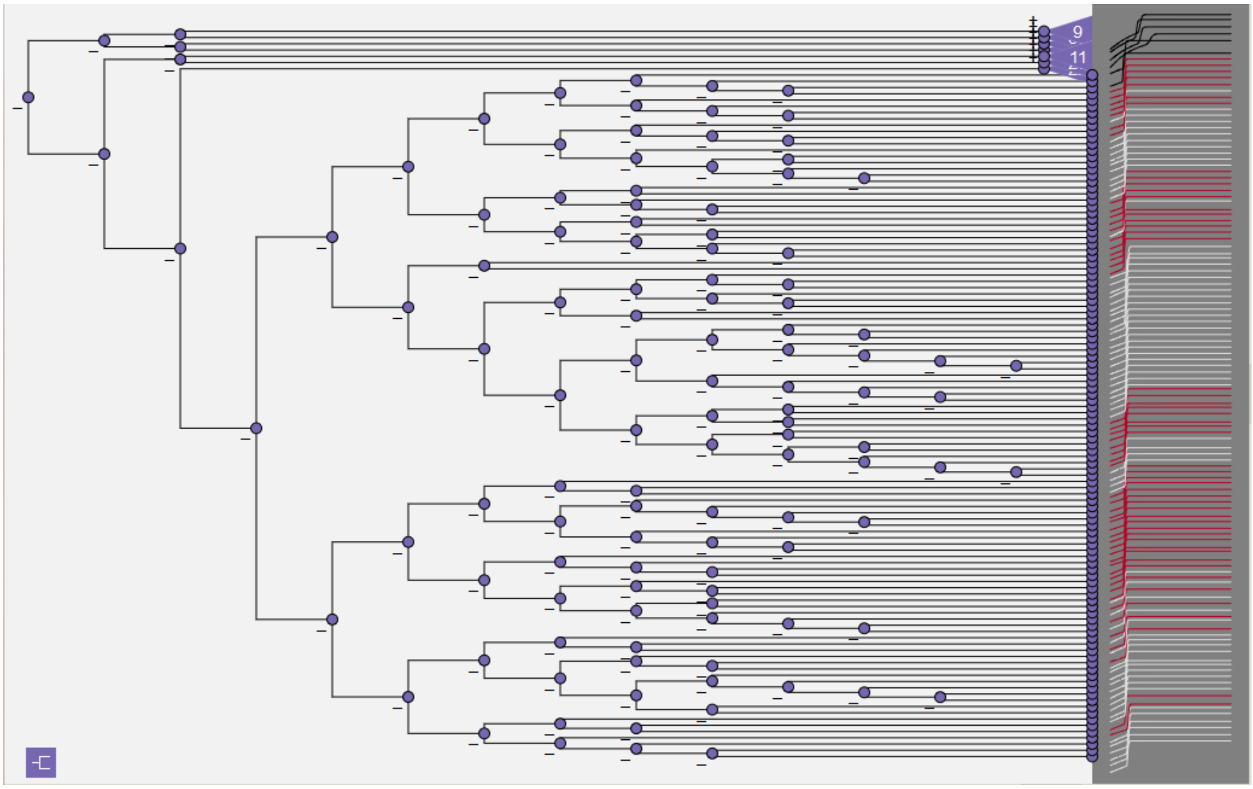


Figure 59: One-click expansion of subtree with 110 nodes.

The operation of collapsing nodes reduces all of the nodes below the designated node (the node whose '-' icon is clicked) into a single subtree, regardless of the number of levels that are currently visible. In Figure 60, the bottom half of the dendrogram in Figure 57 has been collapsed into a single subtree with 166 nodes. This ability to expand and contract sections of the dendrogram controls the level of detail, maximizing the rendered portion of the tree around areas of interest.

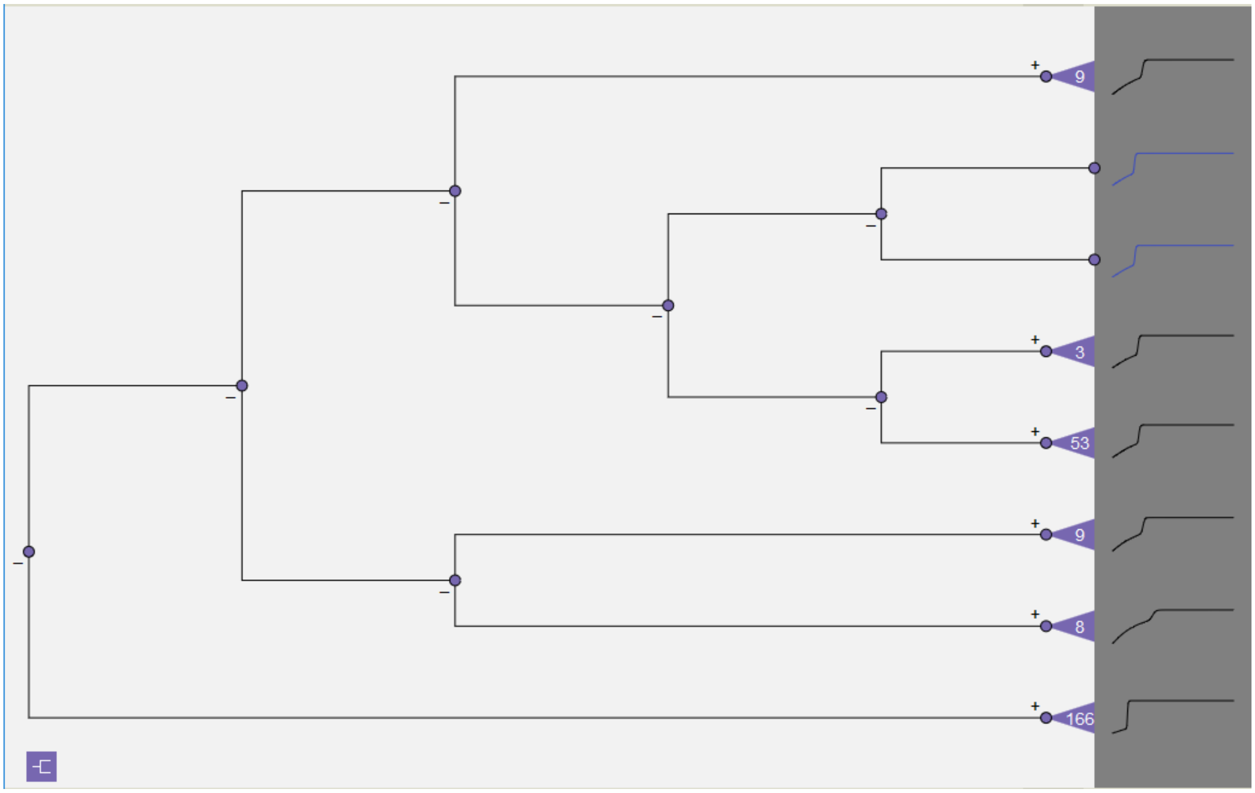
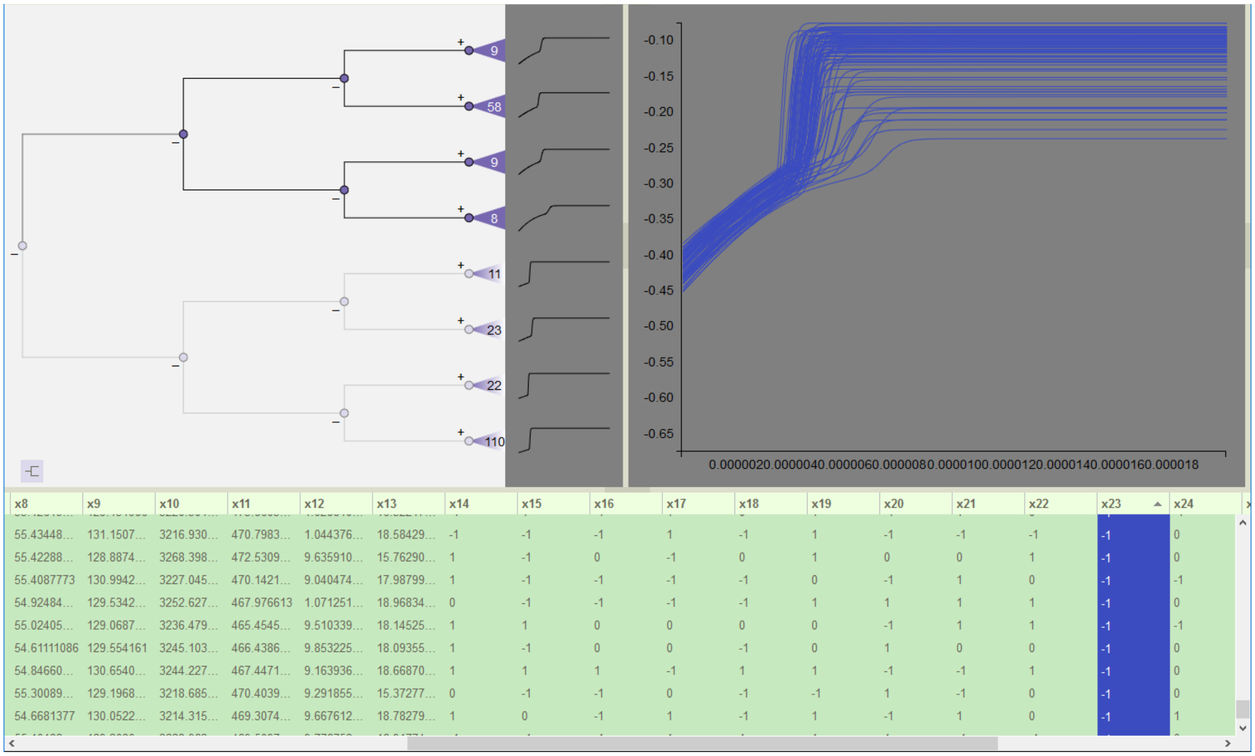


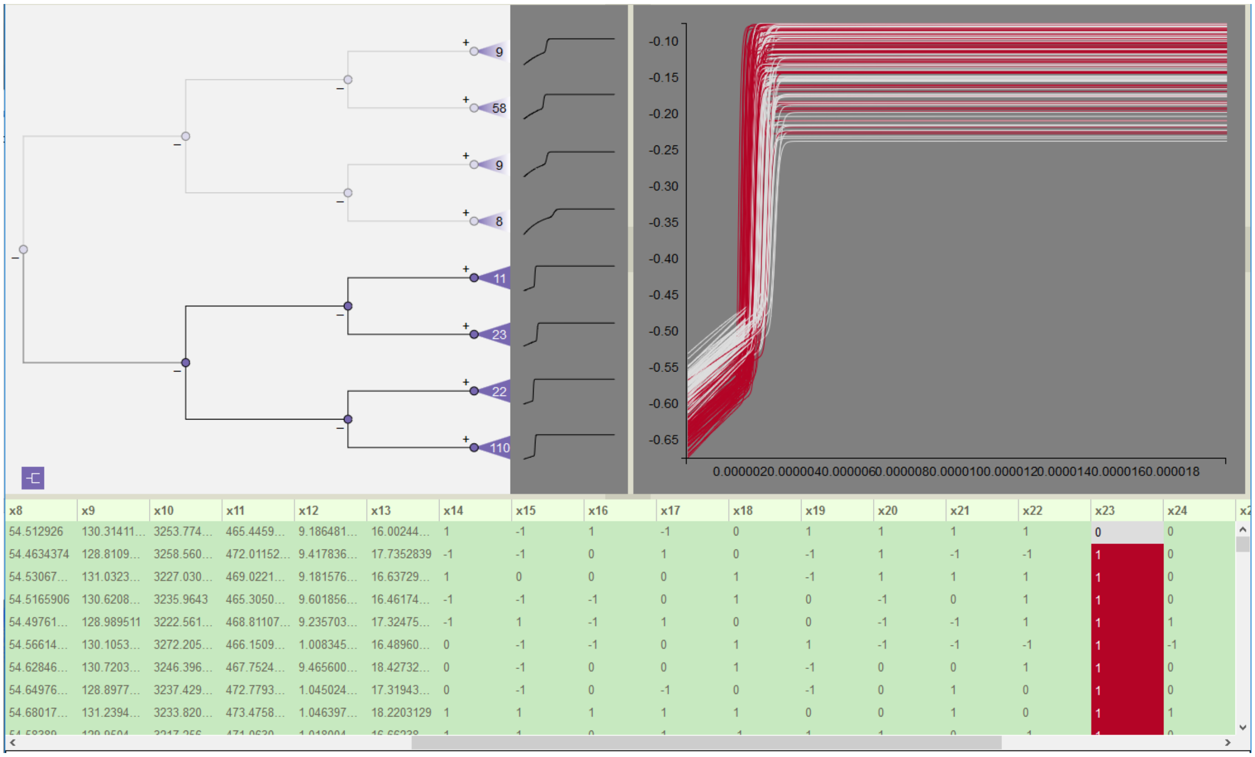
Figure 60: Collapsed subtree that combines the 4 subtrees from the lower half of the dendrogram in Figure 57.

The dendrogram also acts as a visibility filter to select which lines are shown in the *Simulation View* and which rows appear in the *Variable Table*. Click on the purple dot representing any node (or the dot at the tip of a subtree triangle) to restrict visibility to the leaf nodes associated with that subset of the tree. Non-visible nodes are grayed out. Figure 61 and Figure 62 show the results of limiting visibility to the subtrees of the upper and lower halves of the dendrogram, respectively. These examples clearly demonstrate that the upper and lower subtrees are grouping the results generated by input differences in the variable *x23* into two categories. Inputs of -1 (color-coded blue) start higher on the y-axis and escalate slowly over a longer time period before peaking, while 0 and 1 (white and red, respectively) start lower on the y-axis and peak more rapidly. These two groups have distinctly different characteristics, which the analysis has captured.



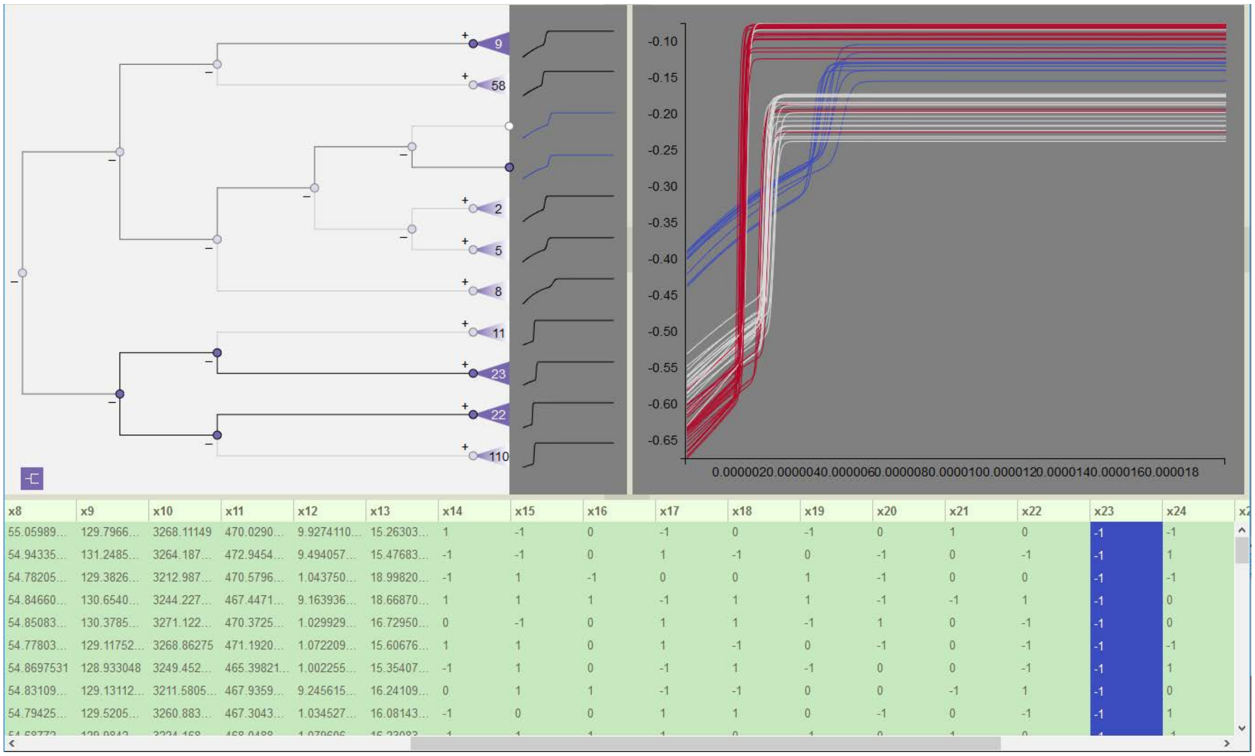
x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	x21	x22	x23 ▲	x24
55.43448...	131.1507...	3216.930...	470.7983...	1.044376...	18.58429...	-1	-1	-1	1	-1	1	-1	-1	-1	-1	0
55.42288...	128.8874...	3268.398...	472.5309...	9.635910...	15.76290...	1	-1	0	-1	0	1	0	0	1	-1	0
55.4087773	130.9942...	3227.045...	470.1421...	9.040474...	17.98799...	1	-1	-1	-1	-1	0	-1	1	0	-1	-1
54.92484...	129.5342...	3252.627...	467.976613	1.071251...	18.96834...	0	-1	-1	-1	-1	1	1	1	1	-1	0
55.02405...	129.0687...	3236.479...	465.4545...	9.510339...	18.14525...	1	1	0	0	0	0	-1	1	1	-1	-1
54.61111086	129.554161	3245.103...	466.4386...	9.853225...	18.09355...	1	-1	0	0	-1	0	1	0	0	-1	0
54.84660...	130.6540...	3244.227...	467.4471...	9.163936...	18.66870...	1	1	1	-1	1	1	-1	-1	1	-1	0
55.30089...	129.1968...	3218.685...	470.4039...	9.291855...	15.37277...	0	-1	-1	0	-1	-1	1	-1	0	-1	0
54.6681377	130.0522...	3214.315...	469.3074...	9.667612...	18.78279...	1	0	-1	1	-1	1	-1	1	0	-1	1

Figure 61: Visibility is limited to nodes in top half of dendrogram by clicking the upper second-level node. Table order is sorted by the values of variable x23 (not in dendrogram leaf order, shown by lavender graph icon in dendrogram lower left).



x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	x21	x22	x23	x24
54.512926	130.31411...	3253.774...	465.4459...	9.186481...	16.00244...	1	-1	1	-1	0	1	1	1	1	0	0
54.4634374	128.8109...	3258.560...	472.01152...	9.417836...	17.7352839	-1	-1	0	1	0	-1	1	-1	-1	1	0
54.53067...	131.0323...	3227.030...	469.0221...	9.181576...	16.63729...	1	0	0	0	1	-1	1	1	1	1	0
54.5165906	130.6208...	3235.9643	465.3050...	9.601856...	16.46174...	-1	-1	-1	0	1	0	-1	0	1	1	0
54.49761...	128.989511	3222.561...	468.81107...	9.235703...	17.32475...	-1	1	-1	1	0	0	-1	-1	1	1	1
54.56614...	130.1053...	3272.205...	466.1509...	1.008345...	16.48960...	0	-1	-1	0	1	1	-1	-1	-1	1	-1
54.62846...	130.7203...	3246.396...	467.7524...	9.465600...	18.42732...	0	-1	0	0	1	-1	0	0	1	1	0
54.64976...	128.8977...	3237.429...	472.7793...	1.045024...	17.31943...	0	-1	0	-1	0	-1	0	1	0	1	0
54.68017...	131.2394...	3233.820...	473.4758...	1.046397...	18.2203129	1	1	1	1	1	0	0	1	0	1	1

Figure 62: Visibility is limited to nodes in bottom half of dendrogram by clicking the lower second-level node. Table order is in default dendrogram leaf order (shown by purple graph icon in lower left of dendrogram).



x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	x21	x22	x23	x24
55.05989...	129.7966...	3268.11149	470.0290...	9.9274110...	15.26303...	1	-1	0	-1	0	-1	0	1	0	-1	-1
54.94335...	131.2485...	3264.187...	472.9454...	9.494057...	15.47683...	-1	-1	0	1	-1	0	-1	0	-1	-1	1
54.78205...	129.3826...	3212.987...	470.5796...	1.043750...	18.99820...	-1	1	-1	0	0	1	-1	0	0	-1	-1
54.84660...	130.6540...	3244.227...	467.4471...	9.163936...	18.66870...	1	1	1	-1	1	1	-1	-1	1	-1	0
54.85083...	130.3785...	3271.122...	470.3725...	1.029929...	16.72950...	0	-1	0	1	1	-1	1	0	-1	-1	0
54.77803...	129.11752...	3268.86275	471.1920...	1.072209...	15.60676...	1	1	0	1	-1	0	-1	0	-1	-1	-1
54.8697531	128.933048	3249.452...	465.39821...	1.002255...	15.35407...	-1	1	0	-1	1	-1	0	0	-1	-1	1
54.83109...	129.13112...	3211.5805...	467.9359...	9.245615...	16.24109...	0	1	1	-1	-1	0	0	-1	1	-1	0
54.79425...	129.5205...	3260.883...	467.3043...	1.034527...	16.08143...	-1	0	0	1	1	0	-1	0	-1	-1	1

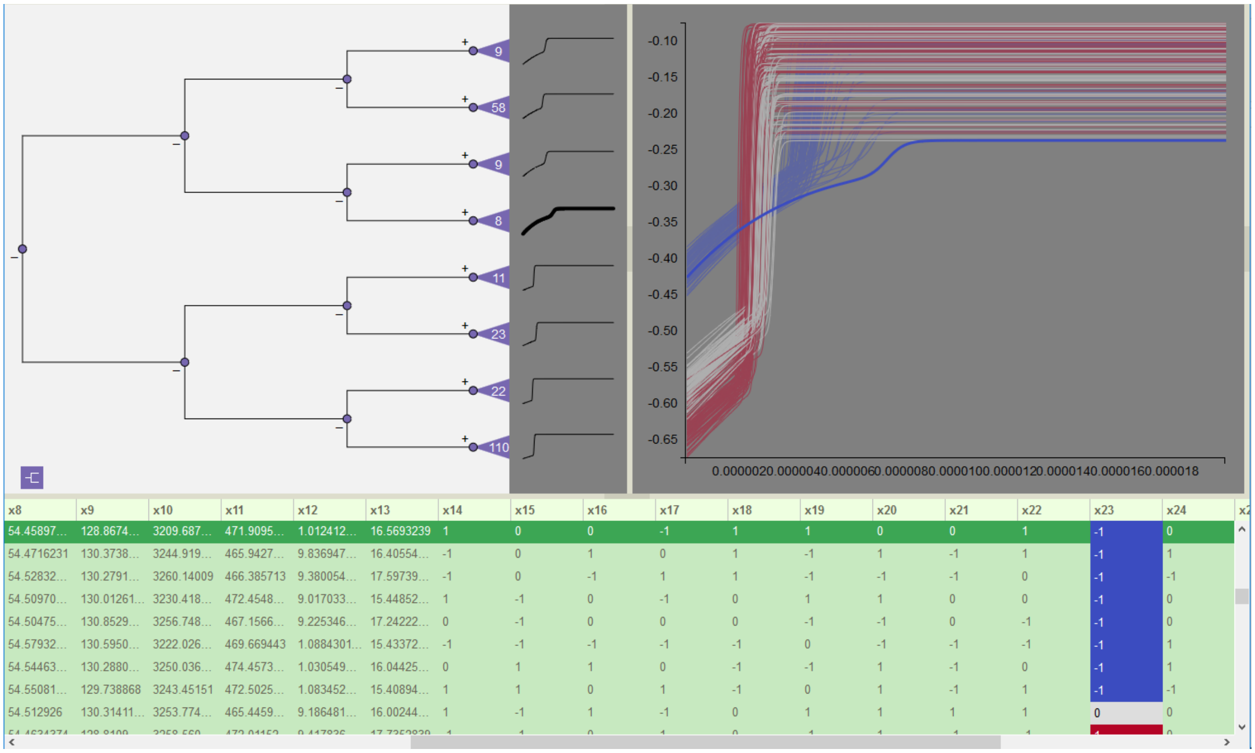
Figure 63: Complex visibility selection made through a combination of dendrogram expansion and using control-click to add nodes to the visible set. Paths from the root to the visible nodes are darkened, while the other paths are grayed out.

As shown in Figure 63, more complex visibility selections can be constructed through a combination of dendrogram expansion operations and using control-click to add individual nodes to the visible set. Control-click functions as a toggle, so it flips the visibility state for any node with each click. The paths from the root to all visible nodes are darkly drawn, while the remaining paths, though still visible, are grayed out.

Sparklines

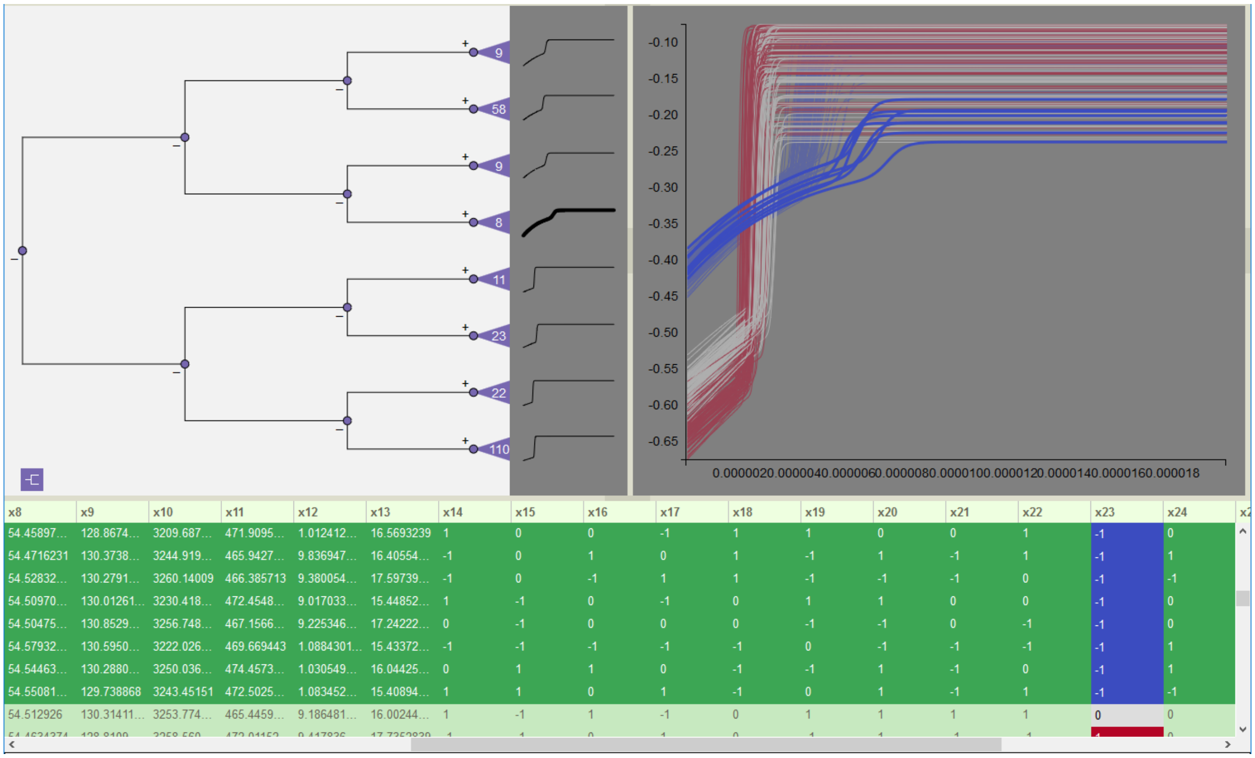
To the right of the subtree icons and leaf nodes are small graphs, called *sparklines* [9], providing a high-level representation of the general shape characteristics of the associated node/subtree. For a node, its *sparkline* is its time series plot rendered into a thumbnail image. For a subtree, its *sparkline* is the *sparkline* of the node closest to the centroid of the group in the subtree.

Sparklines for subtrees are drawn in black. *Sparklines* for leaf nodes are color-coded to match the line color of the corresponding run in the *Simulation View*, and the cell color of the corresponding simulation in the *Variable Table*. Beyond color-coding being linked between all three views, selection is also linked. Selection of a line (or lines) in the *Simulation View* or *Variable View* will highlight (darken) the *sparklines* of the associated subtrees and/or nodes to reveal their location within the hierarchy, as shown in Figure 64. Alternatively, clicking on a *sparkline* performs a group operation that selects the associated node set, highlighting the corresponding lines in the *Simulation View*, and the corresponding rows in the *Variable Table*. Figure 65 shows how clicking the *sparkline* for the 8 node subtree results in highlighting the eight associated lines and rows in the other two views.



x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	x21	x22	x23	x24
54.45897...	128.8674...	3209.687...	471.9095...	1.012412...	16.5693239	1	0	0	-1	1	1	0	0	1	-1	0
54.4716231	130.3738...	3244.919...	465.9427...	9.836947...	16.40554...	-1	0	1	0	1	-1	1	-1	1	-1	1
54.52832...	130.2791...	3260.14009	466.385713	9.380054...	17.59739...	-1	0	-1	1	1	-1	-1	-1	0	-1	-1
54.50970...	130.01261...	3230.418...	472.4548...	9.017033...	15.44852...	1	-1	0	-1	0	1	1	0	0	-1	0
54.50475...	130.8529...	3256.748...	467.1566...	9.225346...	17.24222...	0	-1	0	0	0	-1	-1	0	-1	-1	0
54.57932...	130.5950...	3222.026...	469.669443	1.0884301...	15.43372...	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	1
54.54463...	130.2880...	3250.036...	474.4573...	1.030549...	16.04425...	0	1	1	0	-1	-1	1	-1	0	-1	1
54.55081...	129.738868	3243.45151	472.5025...	1.083452...	15.40894...	1	1	0	1	-1	0	1	-1	1	-1	-1
54.512926	130.31411...	3253.774...	465.4459...	9.186481...	16.00244...	1	-1	1	-1	0	1	1	1	1	0	0

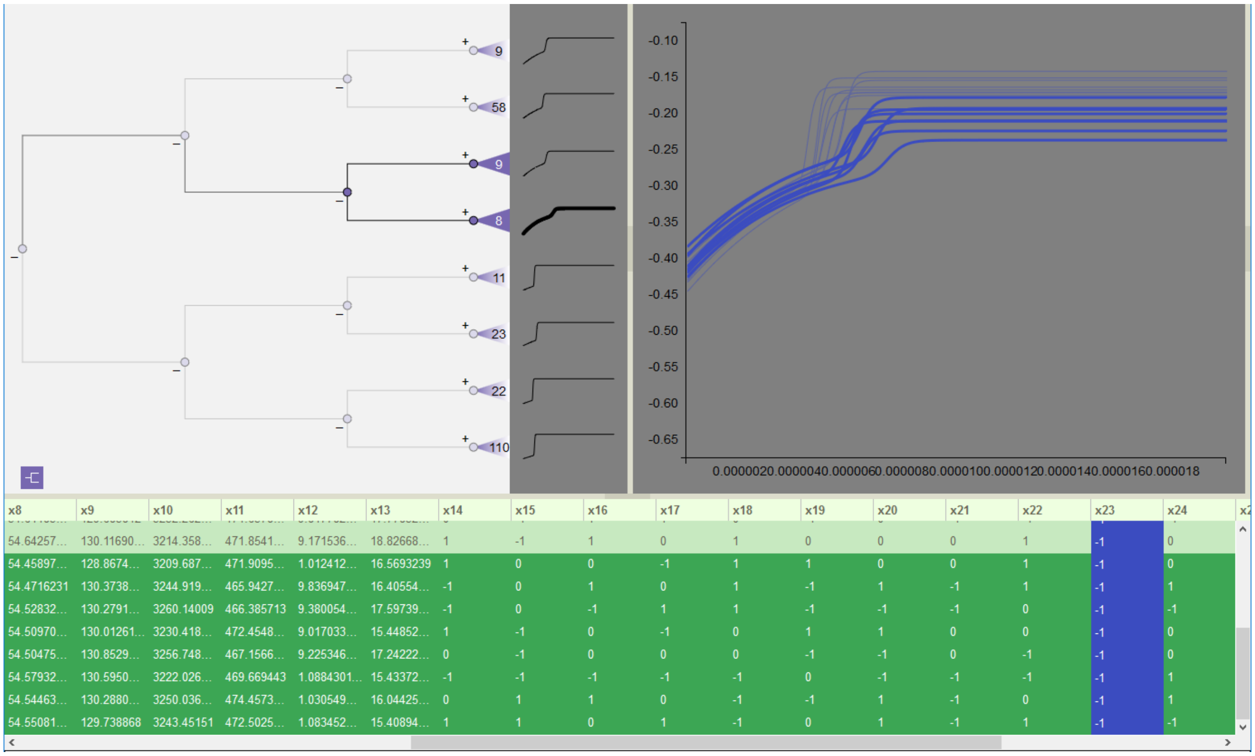
Figure 64: *Sparkline* highlights corresponding subtree for selected line in *Simulation View*,



x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	x21	x22	x23	x24
54.45897...	128.8674...	3209.687...	471.9095...	1.012412...	16.5693239	1	0	0	-1	1	1	0	0	1	-1	0
54.4716231	130.3738...	3244.919...	465.9427...	9.836947...	16.40554...	-1	0	1	0	1	-1	1	-1	1	-1	1
54.52832...	130.2791...	3260.14009	466.385713	9.380054...	17.59739...	-1	0	-1	1	1	-1	-1	-1	0	-1	-1
54.50970...	130.01261...	3230.418...	472.4548...	9.017033...	15.44852...	1	-1	0	-1	0	1	1	0	0	-1	0
54.50475...	130.8529...	3256.748...	467.1566...	9.225346...	17.24222...	0	-1	0	0	0	-1	-1	0	-1	-1	0
54.57932...	130.5950...	3222.026...	469.669443	1.0884301...	15.43372...	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	1
54.54463...	130.2880...	3250.036...	474.4573...	1.030549...	16.04425...	0	1	1	0	-1	-1	1	-1	0	-1	1
54.55081...	129.738868	3243.45151	472.5025...	1.083452...	15.40894...	1	1	0	1	-1	0	1	-1	1	-1	-1
54.512926	130.31411...	3253.774...	465.4459...	9.186481...	16.00244...	1	-1	1	-1	0	1	1	1	1	0	0

Figure 65: Clicking the *sparkline* for the 8-node subtree highlights the corresponding set of lines and table rows.

Highlighting and visibility are independent functions that may be combined. For example, in Figure 66, the set of runs that failed to peak are selected in the dendrogram. Highlighting is used to distinguish the runs in the subtree with 8 nodes from the subtree with 9 nodes. The 8 node subtree consists of runs that took longer to rise and had lower overall values than the 9 node group. Note that these differences are visible even in the *sparklines* for each subtree.



x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20	x21	x22	x23	x24
54.64257...	130.11690...	3214.358...	471.8541...	9.171536...	18.82668...	1	-1	1	0	1	0	0	0	1	-1	0
54.45897...	128.8674...	3209.687...	471.9095...	1.012412...	16.5693239	1	0	0	-1	1	1	0	0	1	-1	0
54.4716231	130.3738...	3244.919...	465.9427...	9.836947...	16.40554...	-1	0	1	0	1	-1	1	-1	1	-1	1
54.52832...	130.2791...	3260.14009	466.385713	9.380054...	17.59739...	-1	0	-1	1	1	-1	-1	-1	0	-1	-1
54.50970...	130.01261...	3230.418...	472.4548...	9.017033...	15.44852...	1	-1	0	-1	0	1	1	0	0	-1	0
54.50475...	130.8529...	3256.748...	467.1566...	9.225346...	17.24222...	0	-1	0	0	0	-1	-1	0	-1	-1	0
54.57932...	130.5950...	3222.026...	469.669443	1.0884301...	15.43372...	-1	-1	-1	-1	-1	0	-1	-1	-1	-1	1
54.54463...	130.2880...	3250.036...	474.4573...	1.030549...	16.04425...	0	1	1	0	-1	-1	1	-1	0	-1	1
54.55081...	129.738868	3243.45151	472.5025...	1.083452...	15.40894...	1	1	0	1	-1	0	1	-1	1	-1	-1

Figure 66: Select visibility combined with highlighting to explore differences in the set of runs that did not peak.

Simulation View

The *Simulation View* is a line plot, where each line represents an ensemble member. The X axis is the shared range of temporal values between simulations, and the Y axis is the time series variable value. The lines are color-coded by using the value of a selected scalar variable (see Color-Coding Lines) to potentially reveal correlations between inputs and groups of similar output plots, as demonstrated by the examples of the previous section.

Line visibility in the *Simulation View* is controlled by selecting nodes and subtrees within the dendrogram (see Dendrogram View). Moving the mouse over the line plots, Slycat™ interactively provides feedback showing which line is being pointed at through a combination of highlighting the focus line and dimming all the other lines. Left-clicking on the line selects the simulation in all views, highlighting the *sparkline* next to the associated subtree in the dendrogram, highlighting the line in the line plot, and highlighting the associated row in the table. Multiple lines can be selected using control-click to toggle the selection state of lines (i.e. holding the control key while clicking adds unselected lines to the selection set, or removes previously selected lines from the set). Control-click selection is available in all three views, operating on *sparklines*, line plots, or table rows. To clear the current set of selections, click in the background of the *Simulation View* in any area away from the lines.

Color-Coding Lines

The first time a model is rendered, the lines are colored by their index number. There are two mechanisms for changing the variable selected for the color-coding: clicking on the column header in the *Variable Table* or selecting a variable from the *Line Color* dropdown list. Irrespective of the interface used, changing the color-mapping variable will lead to changes in all three views and the legend, including: recoloring the line plot using the new variable's values, coloring the cell backgrounds in that variable's table column, returning the cell backgrounds of the previously selected variable's column to the default color (green, lavender, or white depending on whether the variable is an input, output, or neither), recoloring any *sparklines* that are leaf nodes in the dendrogram, and relabeling and redefining the value range in the *Legend* (see below).

Legend

To the right of the line plot is the *Legend*. The *Legend* is in its own view, which can be resized or closed altogether. The *Legend* displays information about the current color-coding variable, including its name, range of values, and the mapping between values and colors. The color palette is defined by the current theme (see Color Themes).

Variable Table

The *Variable Table* is much the same as the table in the CCA model (see Variable Table). However, there is one additional option for row ordering in the table, which we refer to as *dendrogram ordering* (i.e. if the dendrogram were expanded out to the leaf level, the simulations associated with the rows in the table would correspond to those of the dendrogram's leaves). This is the default table order when the model is first visualized. The row ordering choice is explicitly shown by the color of the graph icon in the lower left corner of the *Dendrogram View*. When the graph icon is purple, as in Figure 66, the table is in *dendrogram order*. When the graph icon is lavender, as in Figure 61, the table is in *sorted order*. Clicking on the graph icon restores the table to *dendrogram order*, and returns the icon to purple. Sorting any of the table columns replaces this ordering with the *sorted order*, and changes the icon to lavender.

Acknowledgements

Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This document is approved for release as SAND2018-0620 R.

References

[1] Adams, B.M., Ebeida, M.S., Eldred, M.S., Jakeman, J.D., Swiler, L.P., Bohnhoff, W.J., Dalbey, K.R., Eddy, J.P., Hu, K.T., Vigil, D.M., Bauman, L.E., and Hough, P.D., *Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, uncertainty quantification, and sensitivity analysis: Version 5.3.1 user's manual.* Tech. Rep. SAND2010-2183, Sandia National Laboratories (2013).

[2] Anderson, T. W., *An Introduction to Multivariate Statistical Analysis, 3rd ed.*, Wiley, New York (2003).

[3] Ayachit, U., Bauer, A., Geveci, B., O'Leary, P., Moreland, K., Fabian, N., and Mauldin, J., *ParaView Catalyst: Enabling In Situ Data Analysis and Visualization*, Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization (ISAV2015), pp. 25-29, ACM, New York, NY (2015).

[4] Borga, M., Learning Multidimensional Signal Processing. PhD thesis, Linkoping University, Linkoping (1998).

[5] Hotelling, H., Relations Between Two Sets of Variates. *Biometrika*, 28, 321-377 (1936).

[6] Krzanowski, W. J., *Principles of Multivariate Analysis. A User's Perspective.* Oxford University Press, London (1988).

[7] Moreland, K., Diverging Color Maps for Scientific Visualization. *Advances in Visual Computing, vol. 5876*, pp. 92-103. Springer, Berlin (2009).

[8] S. S. M. Team. Sierra/SolidMechanics 4.22 User's Guide. Technical Report SAND2011-7597, Sandia National Laboratories (2011).

[9] Tufte, E., *Beautiful Evidence*, pp. 46-63, Graphics Press, Cheshire, Connecticut (2006).