Eureqa

User Guide

1. Getting Help

If you have questions, comments, or suggestions, please use the Eureqa Google Group at:

http://groups.google.com/group/eureqa-group

This guide describes the basic usage of Eureqa. For tutorials and information on more advanced topics and techniques, visit the Eureqa blog at:

http://blog.eureqa.com

2. Entering Data

You can paste, view, and edit your data in the far-left tab. Entering data is very similar to using an ordinary spreadsheet application.

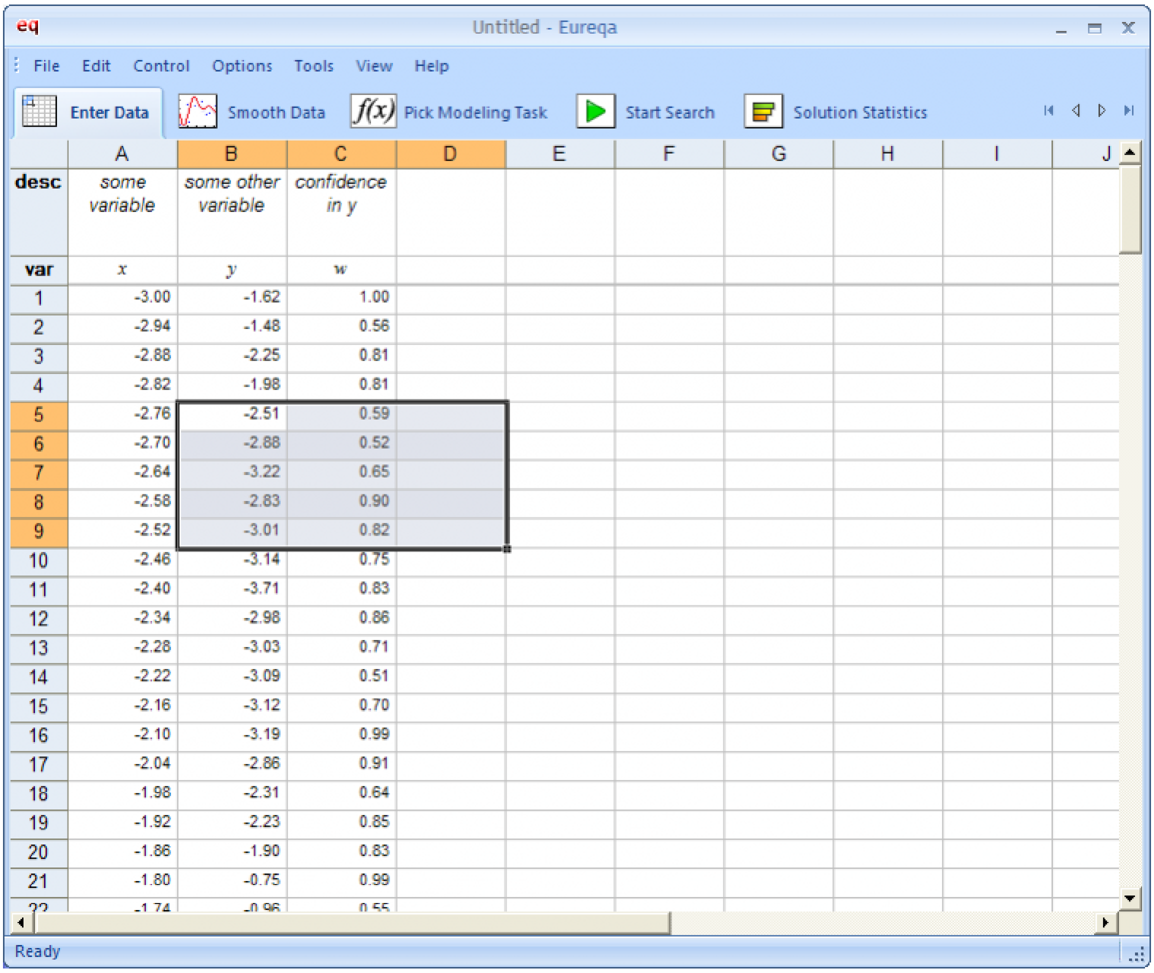
The Eureqa application uses a special layout:

- Each column corresponds to a single variable of your data (e.g. "time", or "oxygen concentration")
- The first row, labeled "**desc**" is for commenting and describing what the variable means or measured
- The second row, labeled "**var**" is the symbol given to each variable (e.g. "x" or "CO2")
- All remaining rows correspond to simultaneous data measurements and values

You can **paste** data into cells from many applications that contain spreadsheets such as Microsoft Excel, Matlab's array editor, or any tab-separated-value text file.

Double click on a cell to edit its value. You may also enter simple expressions into a cell to generate additional variables (for example, "= x + sin(y)" will fill entire column with the numerical result of this expression on each row using the current variable symbols and values.

If your data is partitioned into **discontinuous parts** (e.g. two or more independent time-series or experiments), they should be separated by a blank row. This tells the program not to smooth or differentiate across discontinuous data points.



	A	B	C	D	E	F	G	H	I	J
desc	*some variable*	*some other variable*	*confidence in y*							
var	x	y	w							
1	-3.00	-1.62	1.00							
2	-2.94	-1.48	0.56							
3	-2.88	-2.25	0.81							
4	-2.82	-1.98	0.81							
5	-2.76	-2.51	0.59							
6	-2.70	-2.88	0.52							
7	-2.64	-3.22	0.65							
8	-2.58	-2.83	0.90							
9	-2.52	-3.01	0.82							
10	-2.46	-3.14	0.75							
11	-2.40	-3.71	0.83							
12	-2.34	-2.98	0.86							
13	-2.28	-3.03	0.71							
14	-2.22	-3.09	0.51							
15	-2.16	-3.12	0.70							
16	-2.10	-3.19	0.99							
17	-2.04	-2.86	0.91							
18	-1.98	-2.31	0.64							
19	-1.92	-2.23	0.85							
20	-1.86	-1.90	0.83							
21	-1.80	-0.75	0.99							
22	-1.74	-0.96	0.55							

3. Smoothing Data

This is an **optional step** where the Eureqa application can automatically smooth your data.

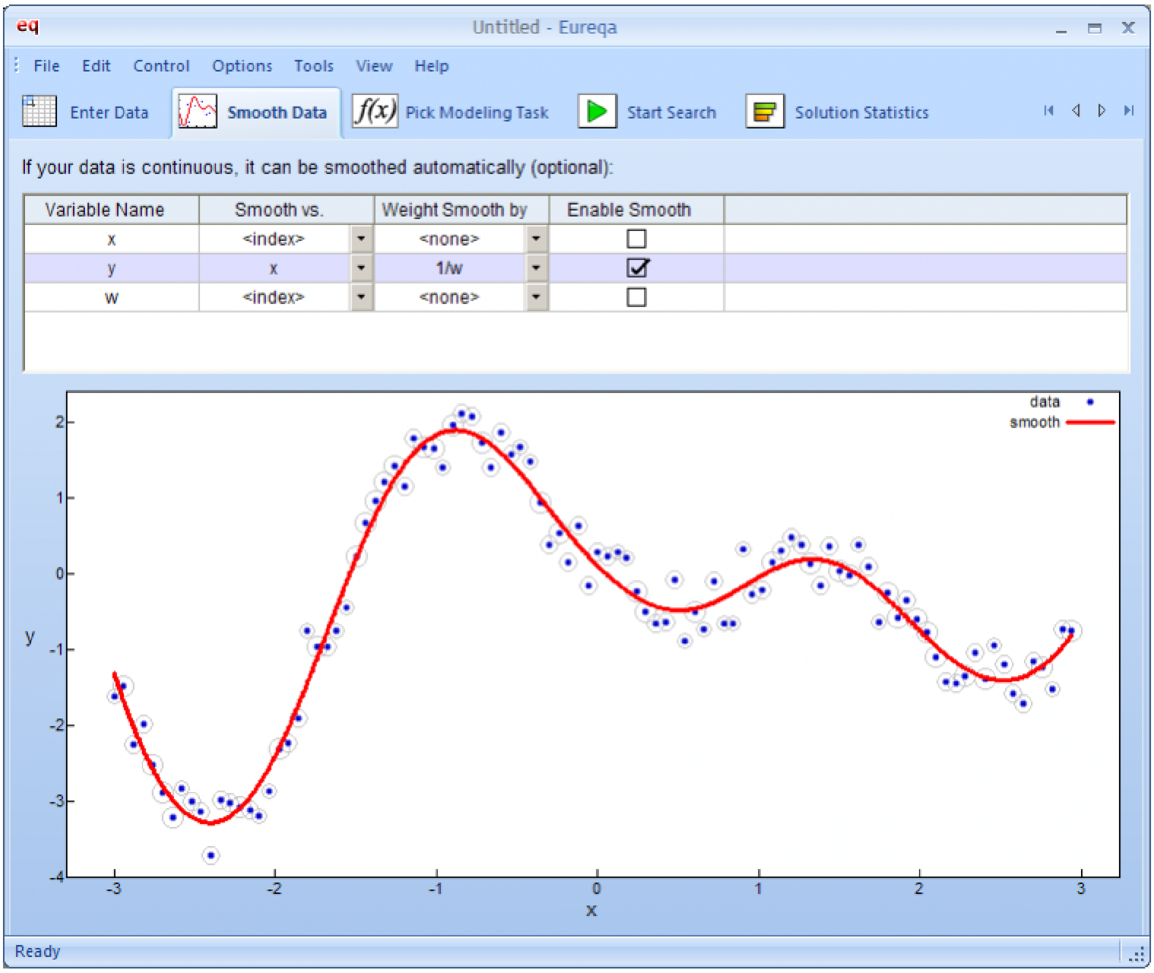
Smoothing data can greatly improve both the speed and the likelihood of finding accurate solutions with the Eureqa formula search. Deciding whether or not to smooth implies some expert knowledge from the user that variables in the data are in fact smooth signals combined with noise.

To smooth a variable in Eureqa:

- Select the row corresponding to the variable you'd like to smooth. This will show the data in the bottom plot.
- Select an **independent variable** in the "Smooth vs." column to smooth against (e.g. "time").
- Select a **confidence weight** in the "Weight Smooth by" column if you have a confidence strength variable entered.

- Checkmark the "Enable Smooth" option to smooth.

Eureqa picks the best smooth using generalized cross-validation among cubic b-splines. If you data can benefit from more sophisticated pre-processing, you are encouraged to do so in another application of your choice and copy the result into Eureqa by hand.



If your data is continuous, it can be smoothed automatically (optional):

Variable Name	Smooth vs.	Weight Smooth by	Enable Smooth
x	<index>	<none>	□
y	x	1/w	☑
w	<index>	<none>	□

4. Pick Modeling Task

Here you control what type of formula to search for, and how to search for it.

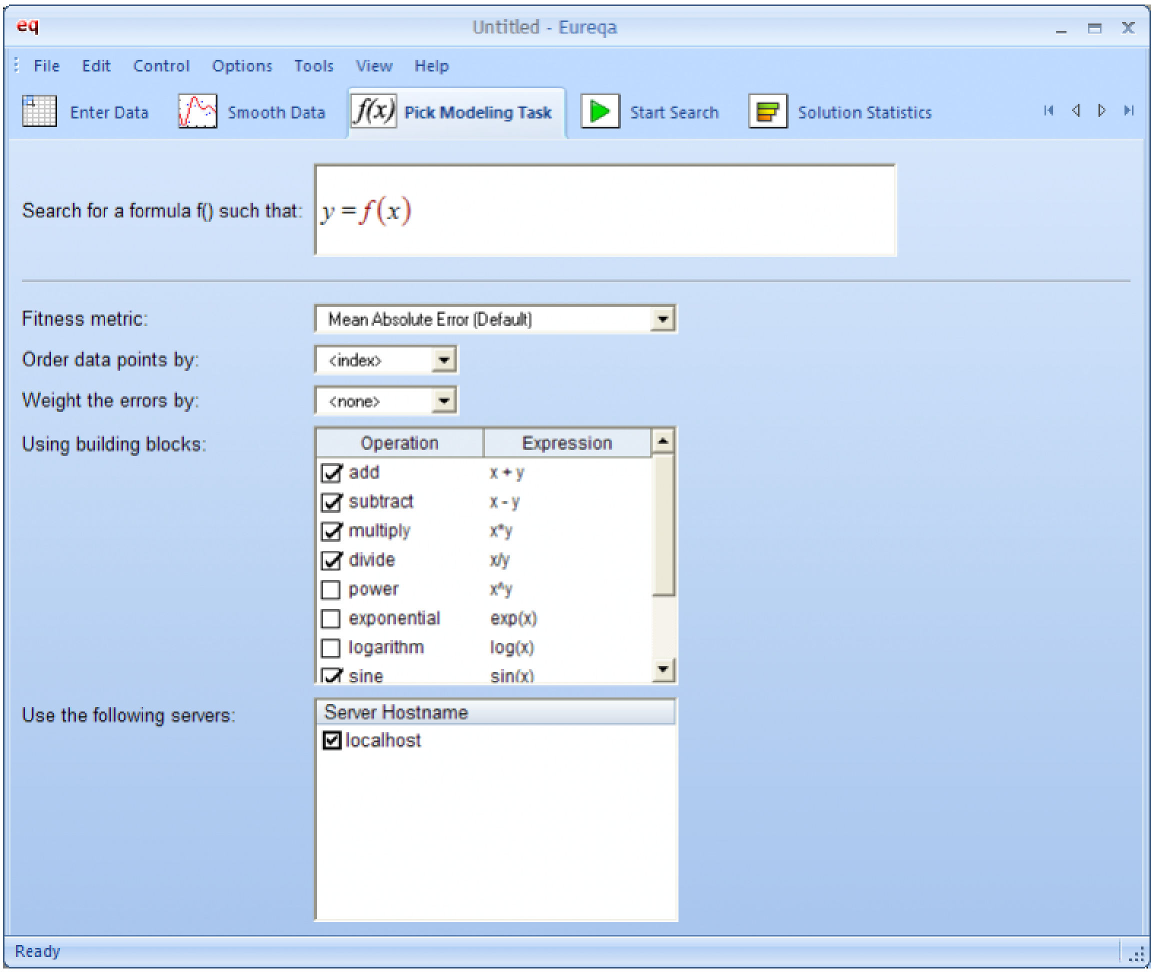
Search for a formula *f()* such that:

Edit this expression to specify the type of relationship you want to model. For example, if you want to model the variable *"z"* as a function of "*x*" and "*y*", you would enter "*z* = *f*(*x*, *y*)".

To search for a differential equation, you can use the *D*(x,y) command. For example, to find an ordinary differential equation for "*y*" as a function of "*y*", you could enter "*D*(*y*,*t*) = *f*(*y*)".

You may also enter more complex target forms. For example, entering "*z* = *f*(*x*) + *f*(*y*)" indicates that you would like to find a function *f* that is evaluated on both "*x*", and "*y*", then added together to model *"z"*.

Remaining options are described below...



Search for a formula f() such that: $y = f(x)$

Fitness metric: Mean Absolute Error (Default)

Order data points by: <index>

Weight the errors by: <none>

Using building blocks:

Operation	Expression
☑ add	x + y
☑ subtract	x - y
☑ multiply	x*y
☑ divide	x/y
□ power	x^y
□ exponential	exp(x)
□ logarithm	log(x)
☑ sine	sin(x)

Use the following servers:

Server Hostname
☑ localhost

Fitness metric:

This specifies what type of error to measure when comparing and optimizing solutions. For example, you may wish to minimize "Squared Error" if your data has normally distributed noise, or "Logarithmic Error" if it contains many outliers.

The list below describes some of the fitness metrics available in Eureqa. All fitness metrics are normalized based on the target values in the data set.

- **Mean Absolute Error (MAE):** minimizes the mean of the absolute value of residual errors, mean(abs(error)). Assumes noise follows a double exponential distribution.

- **Mean Squared Error (MSE):** minimizes the mean of the squared residual errors, mean(error^2). Assumes noise follows a normal distribution.

- **Root Mean Squared Error (RMSE):** minimizes the square root of the MSE, sqrt(mean(error^2)). Assumes noise follows normal distribution.

- **Mean Logarithmic Error (MLE):** minimizes the mean of the natural log of the residual errors, mean(log(1 + error)). Assumes noise follows a heavy-tailed distribution with large outliers.

- **Exponential Mean Logarithmic Error (EMLE):** minimizes the exponential of the MLE, exp(mean(log(1 + error))). Assumes noise follows a heavy-tailed distribution with large outliers.

- **Correlation Coefficient (R):** maximizes the correlation coefficient, normalized covariance. Scale and offset invariant, models the "shape" of the data.

- **Minimize the Difference:** minimizes the signed difference between left and right hand sides of the target formula. Use to create custom fitness functions, for example "(y - f(x))^4 = 0", would minimize the 4th-power error.

- **Akaike Information Criterion (AIC):** minimizes natural log of the MSE and number of parameters (see wikipedia). Entropy measure, use to explicitly minimize the number of free parameters of the model.

- **Bayesian Information Criterion (BIC):** minimizes natural log of the MSE and number of parameters (see wikipedia). Entropy measure, use to explicitly minimize the number of free parameters of the model.

- **Maximum Error (Maximum):** minimizes the single highest error of the residuals. Use to minimize the worst case error or to force algorithm to model a small residual feature.

- **Median Error (Median):** minimizes the single median error of the residuals. Invariant to outliers, use to minimize the "typical case" error.

- **Implicit Derivative Error (Implicit):** minimizes the difference between implicit derivatives derived from a model and estimated from the data set. Use to search for invariant relationships, e.g. "f(x, y) = 0", where "x" and "y" are continuous and in ordered by an independent variable such as time.

Order data points by:

This is the variable Eureqa will use to plotting your data against by default, and the order used for calculating derivatives if any are used.

Weight errors by:

This is the variable to weight each data point by in the fitness metric.

Using building-blocks:

Eureqa searches for formula by combining mathematical these building-blocks (e.g. add, subtract, multiply, divide). You can limit set of building-blocks that the algorithm uses by checking and un-checking the various built-in operations.

The list below describes several of the Eureqa formula building blocks:

Name	Usage	Comments
constant	1.234	Allows solutions to use numeric constants
add	x + y or add(x,y)	
subtract	x - y or sub(x,y)	
multiply	x * y or mul(x,y)	
divide	x / y or div(x,y)	y must be non-zero
square root	sqrt(x)	Returns x^0.5. x must be positive
exponential	exp(x)	Returns *e*^x
logarithm	log(x)	This is the natural logaritm (base *e*)
sine	sin(x)	The angle is in radians
cosine	cos(x)	The angle is in radians
tangent	tan(x)	The angle is in radians
absolute value	abs(x)	Returns the positive value of x
power	x ^ y or pow(x,y)	x and y could be any expression
power to constant	powc(x,c)	Provides a restricted form of the power building-block. x can be any expression, c must be a constant
time delay	delay(x,c)	Returns the value of expression x at c time units in the past. x can be any expression, c must be a positive constant
time delay of variable	delay_var(v,c)	Provides a restricted form of the delay building-block. Returns the value of variable v at c time units in the past. v must be a variable, c must be a positive constant
simple moving average	sma(v,c) or sma_var(v,c)	Returns the average of the data points within the past c time units. v must be a variable, c must be a postiive constant
time integral	integral(x)	Returns the trapezoidal sum of the expression x, starting at 0 up to the current data point
step function	step(x)	Returns 1 if x is positive, zero otherwise
sign function	sign(x)	Returns -1 if x is negative, +1 if x is positive, and 0 if x is zero
logistic function	logistic(x)	This is a common sigmoid squashing function. Returns 1/(1+ exp(-x))

hill function	hill2(x)	This is a common saturation function. Returns x^2/(1 + x^2). x must be non-zero
gamma function	gamma(x)	This is a continuous version of the factorial. It returns the fast approximation pow((x/*e*)*sqrt(x*sinh(1/x)),x)*sqrt(2**pi*/x). x must be non-zero
gaussian function	gauss(x)	This is a bell-shaped squashing function. Returns exp(-x^2)
minimum	min(x,y)	Returns the minimum (signed) result of x and y for the data point
maximum	max(x,y)	Returns the maximum (signed) result of x and y for the data point
modulo	mod(x,y)	Returns the remainder of x/y
floor	floor(x)	Returns an integer of x rounded down toward -infinity
ceiling	ceil(x)	Returns an integer of x rounded up toward +infinite
less than	less(x,y)	Returns 1 if x < y, 0 otherwise
equal to	equal(x,y)	Returns 1 if x equals y numerically, 0 otherwise
boolean and	and(x,y)	Returns 1 if both x and y are greater than 0, 0 otherwise
boolean or	or(x,y)	Returns 1 if either x or y are greater than 0, 0 otherwise
boolean xor	xor(x,y)	Returns 1 if (x <= 0 and y > 0) or (x > 0 and y <= 0), 0 otherwise
boolean not	not(x)	Returns 0 if x is greater than 0, 1 otherwise
inverse sine	asin(x)	x must be between -1 and +1
inverse cosine	acos(x)	x must be between -1 and +1
inverse tangent	atan(x)	
inverse tangent (2-argument)	atan2(y,x)	Returns atan(y/x), respecting the quadrant and sign of the vector. x and y cannot both be zero
hyperbolic sine	sinh(x)	
hyperbolic cosine	cosh(x)	
hyperbolic tangent	tanh(x)	This is a common squashing function. Returns a value between -1 and +1
inverse hyperbolic sine	asinh(x)	
inverse hyperbolic cosine	acosh(x)	
inverse hyperbolic tangent	atanh(x)	

Limiting the building-blocks implies some expert knowledge from the user. For example, the user may know that a chemical reaction is unlikely to use trigonometric terms.

Limiting the set of building blocks can greatly improve the speed and likelihood that Eureqa finds an exact solution. However, disabling too many building-blocks could preclude the search from finding the exact solution if a necessary operation is disabled.

Using servers:

Eureqa is designed to use multiple computers when searching for solutions. This option allows you to specify additional computers that are running the Eureqa Stand-alone Server.

Eureqa servers running on the local network will be listed automatically. You may need to enter other servers manually by right-clicking on the server list and clicking the "Add hostname..." menu item.

5. Start and Monitor the Search

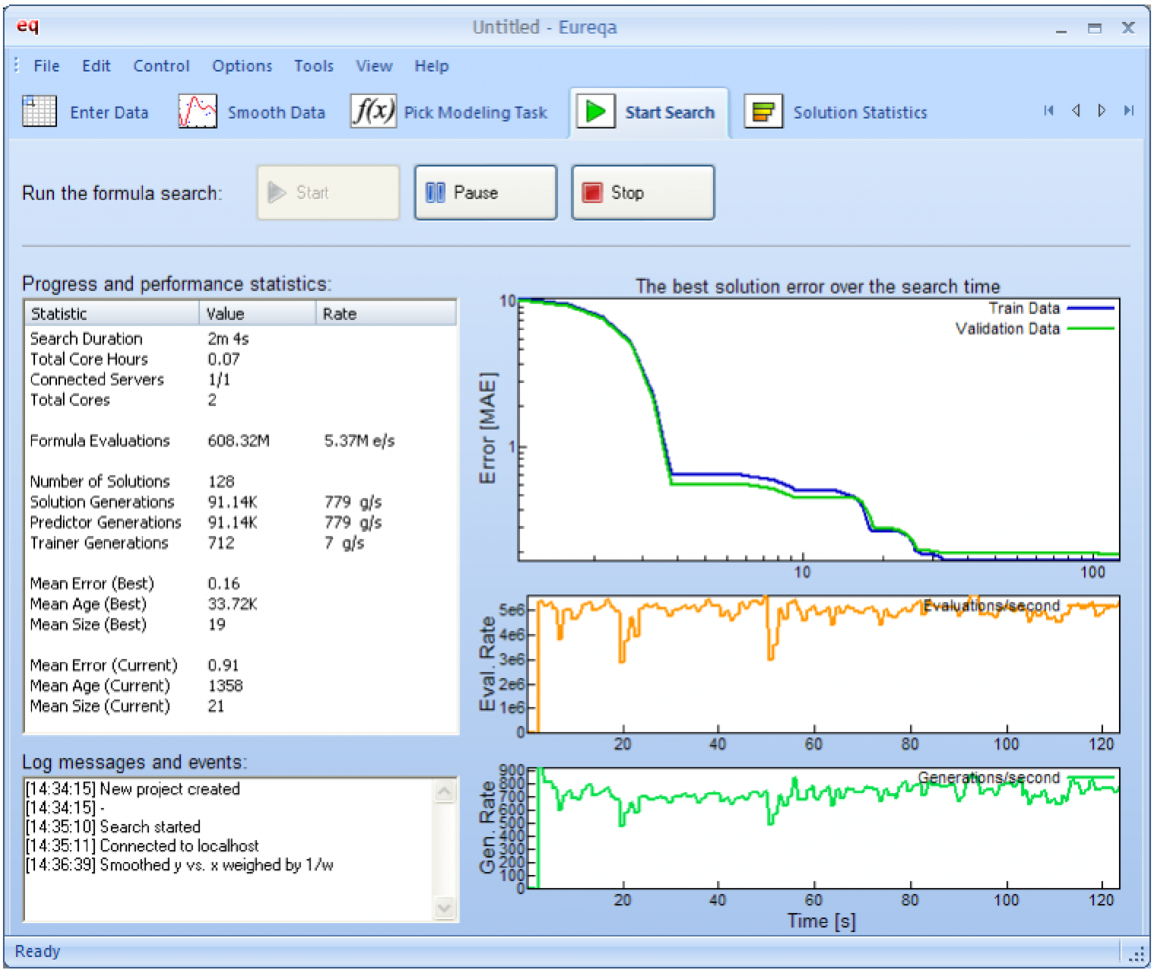
This view allows you to **start**, **pause**, and **stop** the formula search, as well as monitor various performance and progress statistics of the search.

Click on the "Start" button to begin the search.

After starting, Eureqa will attempt to connect to and initialize the computers selected in the settings view. Important messages will be displayed in the "Log messages and events" text window.

The "Progress and performance statistics" window shows several important statistics such as the **time duration of the search**, the **number of servers** connected to, and the **performance speed** of the search.

The main plot to the right shows the fitness metric of the best solution Eureqa was able to find since the search began. Long plateaus in progress may indicate that the search has exhausted searching for the simple explanations of the data.



Statistic	Value	Rate
Search Duration	2m 4s	
Total Core Hours	0.07	
Connected Servers	1/1	
Total Cores	2	
Formula Evaluations	608.32M	5.37M e/s
Number of Solutions	128	
Solution Generations	91.14K	779 g/s
Predictor Generations	91.14K	779 g/s
Trainer Generations	712	7 g/s
Mean Error (Best)	0.16	
Mean Age (Best)	33.72K	
Mean Size (Best)	19	
Mean Error (Current)	0.91	
Mean Age (Current)	1358	
Mean Size (Current)	21	

```
[14:34:15] New project created
[14:34:15] -
[14:35:10] Search started
[14:35:11] Connected to localhost
[14:36:39] Smoothed y vs. x weighed by 1/w
```

6. View and Analyze Results

This view shows the **best solutions** Eureqa finds in real time.

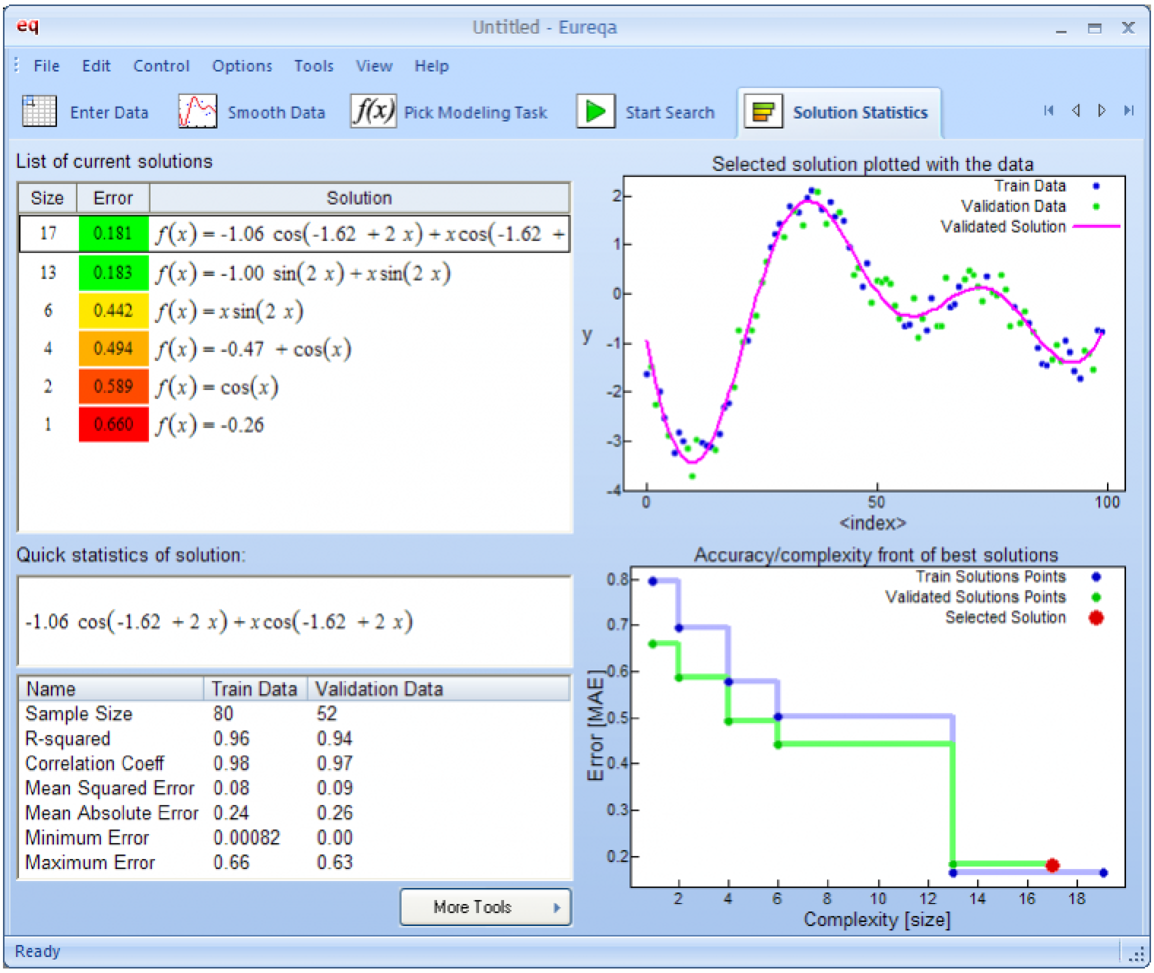
The best solutions are determined by two factors: their **complexity** ("Size") and their **accuracy** ("Error") on the validation data. Those listed in the "List of current best solutions" window have the highest accuracy for various complexities/sizes of solutions.

The **Training Data** is a subset of your data that is being used by the Eureqa algorithm to search for solutions. The **Validation Data** is a second subset that is only used by this window in order to select the best solutions to display/report to the user.

Double-click on a formula in the list to select it in plain text for copying to the clipboard.

Single-click to **select** a formula, which plots it against the data points, and calculates some quick statistics of the fit.

Several tools are also available for further analysis of the selected solution, such as finding the global maximum, generating a report of solutions, and suggesting new experiments and data to collect.



Size	Error	Solution
17	0.181	$f(x) = -1.06\ \cos(-1.62\ +2\ x) + x\cos(-1.62\ +$
13	0.183	$f(x) = -1.00\ \sin(2\ x) + x\sin(2\ x)$
6	0.442	$f(x) = x\sin(2\ x)$
4	0.494	$f(x) = -0.47\ + \cos(x)$
2	0.589	$f(x) = \cos(x)$
1	0.660	$f(x) = -0.26$

$-1.06\ \cos(-1.62\ +2\ x) + x\cos(-1.62\ +2\ x)$

Name	Train Data	Validation Data
Sample Size	80	52
R-squared	0.96	0.94
Correlation Coeff	0.98	0.97
Mean Squared Error	0.08	0.09
Mean Absolute Error	0.24	0.26
Minimum Error	0.00082	0.00
Maximum Error	0.66	0.63

7. More Information

The most up-to-date user guide information is the HTML pages at the Eureqa website:

http://www.eureqa.com

For help, questions, comments, and suggestions, please post to the Eureqa Google Group at:

http://groups.google.com/group/eureqa-group

Finally, visit the Eureqa blog to learn more advanced techniques and tips for modeling using Eureqa at:

http://blog.eureqa.com