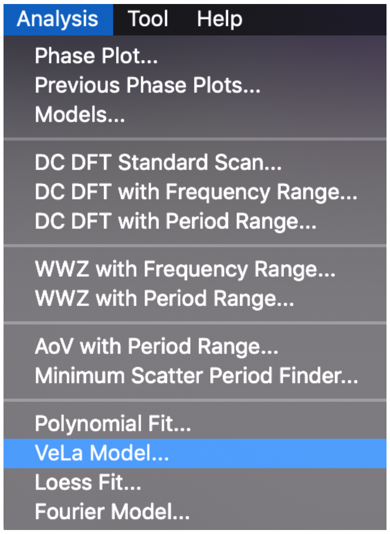
VeLa Model Creator Plug-In

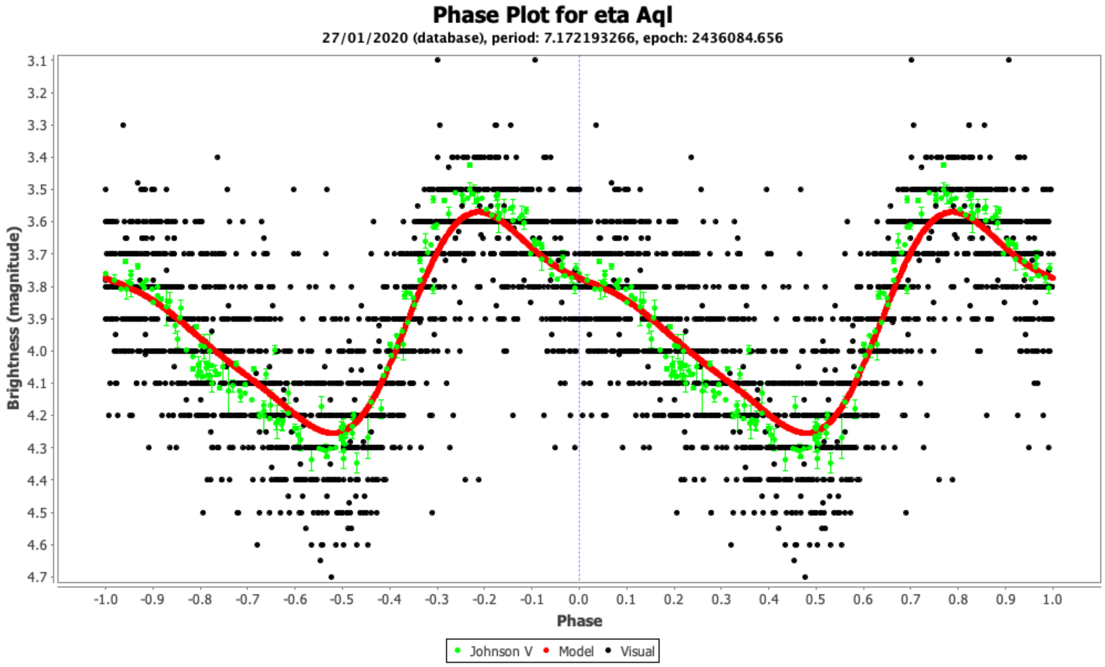
The model creator plug-in is accessible from the `Tool` menu and allows a VeLa model to be created and applied to observations in a selected series.



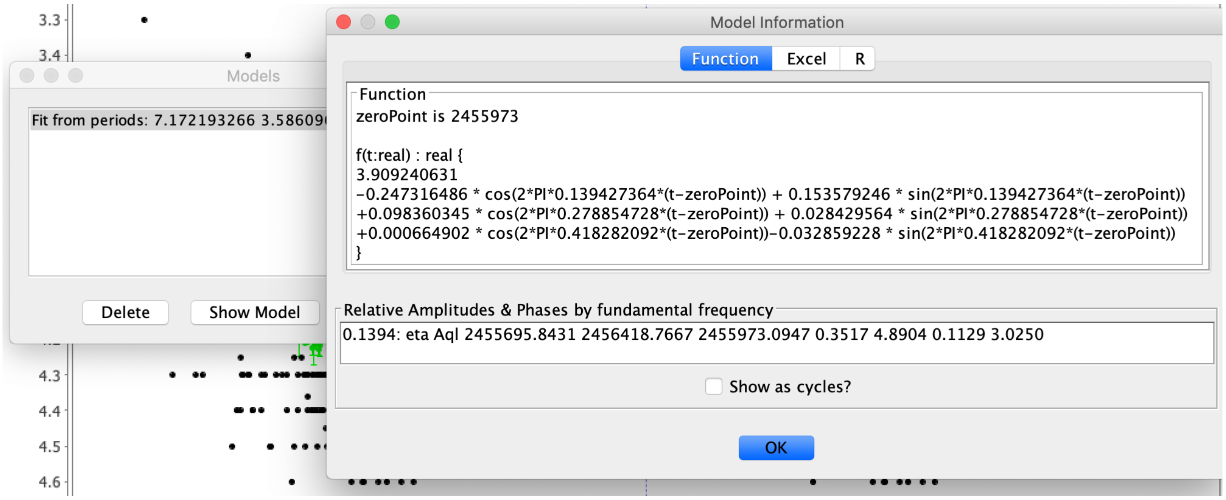
A VeLa function called `f` must be defined that takes a single `real` parameter (of arbitrary name) representing time and returns a `real` result corresponding to evaluation of the function for a given time value.

The function is applied to each time corresponding to observations in the selected series, returning the magnitude value.

Suppose that visual and Johnson V observations for eta Aql have been loaded for the date range 2455695 to 2456419, a 3 harmonic Fourier model from the top-hit period has been created via a `DCDFT Standard Scan` of the Visual series, and a phase plot created:



ng it)



Fit from periods: 7.172193266 3.58609

Function

```
zeroPoint is 2455973

f(t:real) : real {
3.909240631
-0.247316486 * cos(2*PI*0.139427364*(t-zeroPoint)) + 0.153579246 * sin(2*PI*0.139427364*(t-zeroPoint))
+0.098360345 * cos(2*PI*0.278854728*(t-zeroPoint)) + 0.028429564 * sin(2*PI*0.278854728*(t-zeroPoint))
+0.000664902 * cos(2*PI*0.418282092*(t-zeroPoint))-0.032859228 * sin(2*PI*0.418282092*(t-zeroPoint))
}
```

Relative Amplitudes & Phases by fundamental frequency

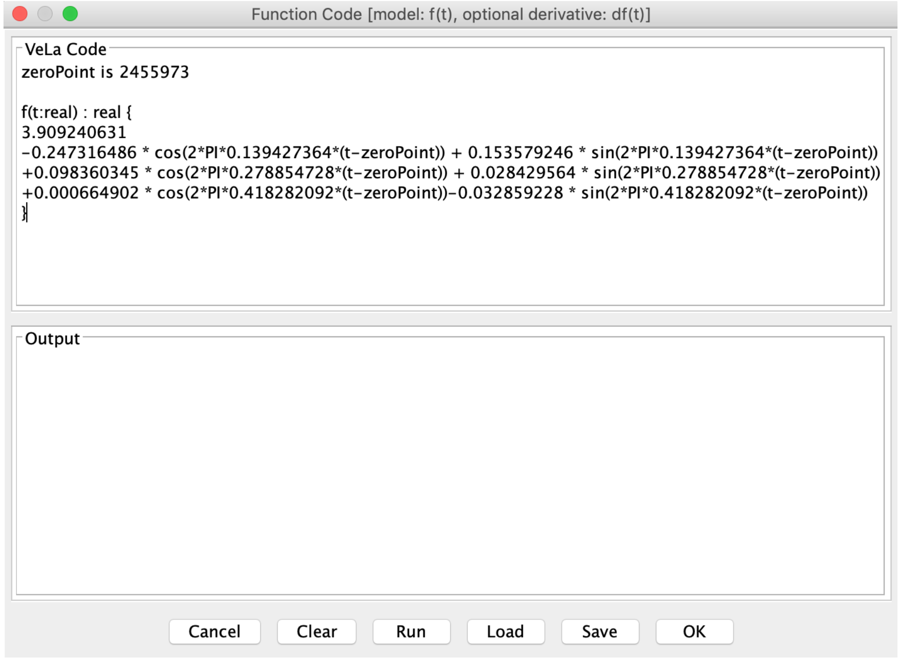
0.1394: eta Aql 2455695.8431 2456418.7667 2455973.0947 0.3517 4.8904 0.1129 3.0250

☐ Show as cycles?

The function text from the models dialog (`Analysis` → `Models...`) can be selected with the mouse and copied.

Suppose that in a future VStar session, the same eta Aql observations are loaded, a phase plot is created (at the same period as before) and `VeLa Model...` is then selected from the `Analysis` menu.

This will open a series selection dialog. Clicking the `OK` button after selecting the series yields a VeLa dialog, in which code can be loaded or pasted and tested before use via the `Run` button:



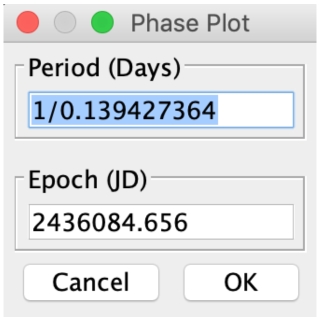
VeLa Code

```
zeroPoint is 2455973

f(t:real) : real {
3.909240631
-0.247316486 * cos(2*PI*0.139427364*(t-zeroPoint)) + 0.153579246 * sin(2*PI*0.139427364*(t-zeroPoint))
+0.098360345 * cos(2*PI*0.278854728*(t-zeroPoint)) + 0.028429564 * sin(2*PI*0.278854728*(t-zeroPoint))
+0.000664902 * cos(2*PI*0.418282092*(t-zeroPoint))-0.032859228 * sin(2*PI*0.418282092*(t-zeroPoint))
}
```

Clicking the `OK` button applies the model function to the selected series, yielding the same Fourier model shown earlier.

As an aside, notice that the primary frequency (or the first harmonic) here is 0.139427364. The phase plot for the loaded observations could be created like so:



Period (Days): 1/0.139427364

Epoch (JD): 2436084.656

Therefore, one use for this plugin is so that a model can be created, the corresponding VeLa function later re-applied when the same observations are loaded at some point in the future.

The example above showed a Fourier model, but any model for which a VeLa model exists can be used, e.g. a polynomial fit.

Optionally, a function called `df` can also be created, with the same parameter and return type as the model function `f`. This function is intended to be the derivative of `f` which will be used to find the model's extrema.

Suppose you have loaded Johnson V data for the eclipsing binary system BL Tel for the date range 2458789 to 2458830 and that you have then created a second degree polynomial model for this and copied the function from the Models dialog:

```
zeroPoint is 2458803.3611

f(t:real) : real {

    -0.0014679078518897227*(t-zeroPoint)^2 +

    0.020030067958860795*(t-zeroPoint)^1 +

    8.762759122195401

}
```

A function corresponding to the first derivative of `f` can be added, as follows, to find `f`'s minima or maxima:

```
df(t:real) : real {

    2*-0.0014679078518897227*(t-zeroPoint) +

    0.020030067958860795

}
```

Entering and applying these two functions via `VeLa Model...` will yield a new model and an entry in the models dialog that includes extrema information that should correspond to those of the original model.

If a VeLa binding called `resolution` is supplied in the model code, it will be used instead of the default of 0.1 when attempting to find extrema if the derivative is defined.

An existing model can also be modified to incorporate additional information, since the VeLa code is available. This allows for a model to take into account other aspects of observations such as observer code. To take full advantage of this, in future, a linear regression capability would need to be available for VeLa models.

Moreover, a model function can be written from scratch, such as this one which is intended to be a piece-wise linear function applied to a Mean series:

https://sourceforge.net/p/vstar/code/HEAD/tree/trunk/script/VeLa/piecewise_linear_model.vl