SLiM Workshop Series #8: SLiMgui Graphing

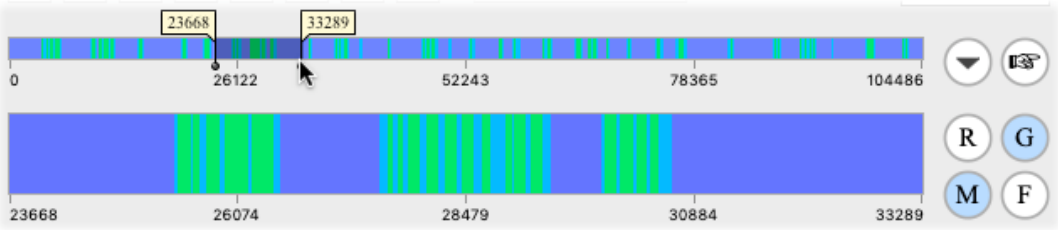
1. Start with recipe 7.3 in SLiMgui

Run SLiMgui, and close any existing model windows. Go to the **Open Recipe** subfolder of the **File** menu, and select recipe 7.3. We will use this recipe as a testbed for exploring various graphing features in SLiMgui.

The model is a bit long to reproduce here, but it is reasonably straightforward: four mutation types (two neutral types, one beneficial, and one deleterious), three genomic element types (representing exons, introns, and non-coding regions), and a randomly generated genomic structure (this is the majority of the code in the whole model). Take a minute to review the model, and look up any functions or methods that are unfamiliar in the online help by option-clicking them.

Step once, to see the genomic structure generated by the model. Since it is random, it's different every run, but it should look something like this:

This is shown in the top chromosome view, which always shows the genomic structure of the model. Click on the **G** button to the right to make the genomic structure display in the bottom chromosome view too. Then zoom in on one region of interest, by dragging out a selection in the top chromosome view. You should see something like this, as you are dragging:



In this zoomed view, we can see that there is some fine structure to the genetic model that was harder to see when zoomed out. But what do the colors represent? Click the tables drawer button to open SLiMgui's drawer:

You should see, among other things, a table showing the genomic element types that are defined in the mode, with color swatches to show how SLiMgui is displaying them:

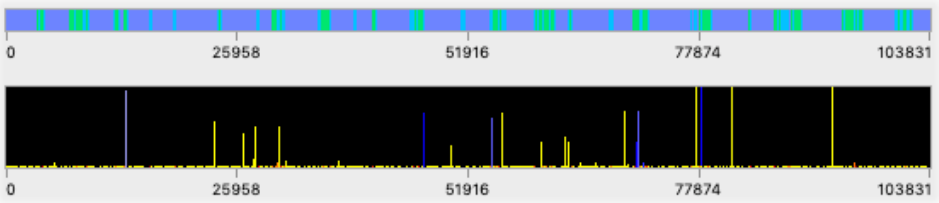
Genomic Element Types:

ID	Color	Mutation types
g1		m2=2.000, m3=8.000,...
g2		m1=9.000, m3=1.000
g3		m1=1.000

From this, we can see that the long purple stretches are type `g3`, declared in the script as "non-coding", whereas the intervening stretches are genes, comprised of alternating exons (`g1`, shown in

turquoise) and introns (`g2`, shown in green). So far so good. Click the tables drawer button again to close the drawer, click once in the top chromosome view to clear the zoomed selection, and click the **G** button again to turn off display of the genomic structure in the bottom chromosome view.

Now click Play to watch the model run. You will see that most mutations are at very low frequency, but that beneficial mutations (green or blue) occasionally sweep, and neutral mutations (yellow) sometimes hitchhike along with those sweeps. Sometimes even a deleterious mutation (orange or red) might get carried along by a sweep, but this is less common; recombination is pretty good at breaking up such associations. A typical snapshot of the running model:



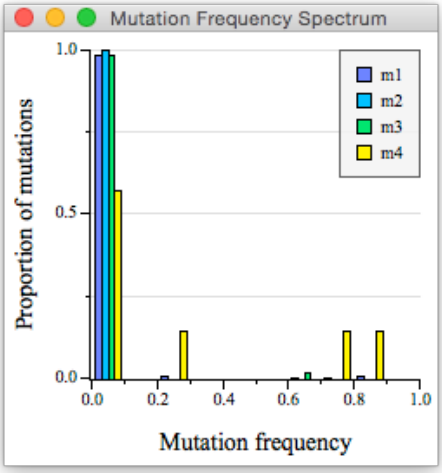
If you look closely, you'll notice that the beneficial mutations are always in exons; the genomic element types are configured such that exons are the only place where beneficial mutations can occur.

That's enough review of the model itself. Now we're going to look at some of the graphing tools SLiMgui provides to help visualize and understand complex models like this.

2. Mutation frequency spectra

Click the graphing button in SLiMgui:

A menu will appear; choose Graph 1D Population SFS. A graph window should appear; depending upon the state of the model when your run was paused, it may or may not look like this:



The *x*-axis is frequency, and the *y*-axis is the proportion of mutations of a given type that are at a particular frequency. We can see that mutations of type `m1`, `m2`, and `m3` are almost all at low frequency; these are the neutral and deleterious mutations. Mutations of type `m4` are often at high frequency, however; these are beneficial mutations, sweeping and fixing.

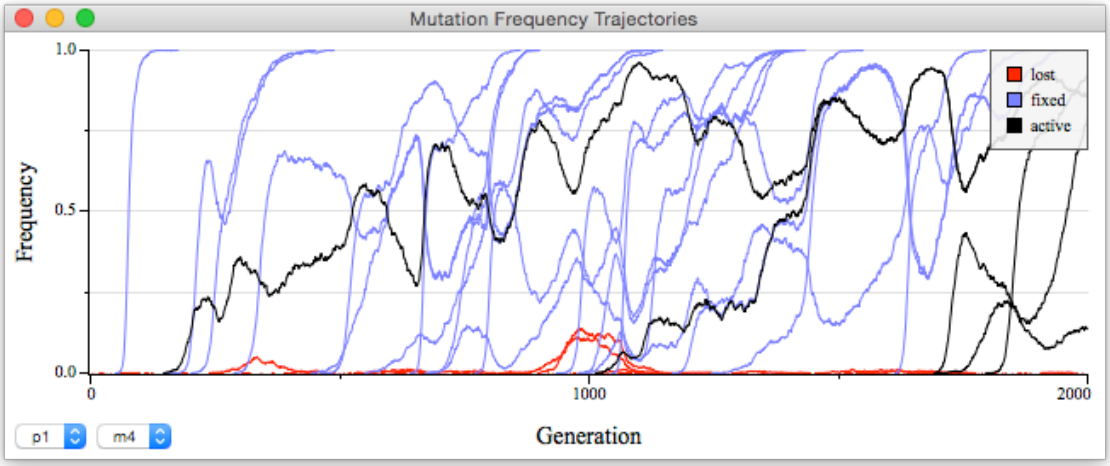
Recycle and run the model now, and you will see that the graph is live: it updates continuously to show the dynamics of the mutation frequency spectrum over time. You can sometimes see the sweep of beneficial mutations, as yellow bars march upward on the graph from low frequency to high.

Right-click on the graph window (i.e., click the right mouse button, or hold down the control key and click), and you will get a pop-up "context menu" showing various options for customizing the graph. We're not going to go into detail on all of these customizations here, but you should play around with them for a couple of minutes before moving on: try hiding and showing the legend and the grid lines, and try changing the binning for the plot by selecting **Change X Bar Count** and setting it to 20. With that many bins, you might then want to resize the window horizontally to give the plot more room. Note that you can also copy the plot to the clipboard or export it as a PDF file, and you can copy the underlying data or save that to a CSV file. Try any of these features you're interested in; you won't do any harm.

When you're ready to move on, close the mutation frequency spectrum graph window.

3. Mutation frequency trajectories

Now click the graphing button again, and select **Graph Mutation Frequency Trajectories**. In this case, the initial graph window you see will probably be empty – just axes and a legend and a couple of pop-up menu buttons at the bottom. This is because the data needed for this graph is not collected by default; it takes up a lot of memory, and is slow to collect. SLiMgui therefore has nothing to plot at the moment. We'd like to see the frequency trajectories for `m4` mutations – the beneficial ones – so select `m4` from the appropriate pop-up, resize the window to be much wider (there will be a lot of information!), and then recycle and run. As the model runs, frequency trajectories will be plotted, and when the model reaches the end you should see something like this:



Each curve is the frequency trajectory for one beneficial mutation. As the legend indicates, those that were ultimately lost are colored red, those that ultimately fixed are blue, and those that are still segregating are black; as time passes, black lines will change to red or blue as their mutation is lost or fixes. You can see that some beneficial mutations exhibit a very clean, classical sweep trajectory, while others display complex dynamics over time – perhaps because their selection coefficient is very small (nearly neutral), or perhaps because of competition among multiple beneficial mutations segregating simultaneously in the population.

Select `m3` from the graph window's popup, then recycle and run again. You will see very different dynamics, with the `m3` (deleterious) mutations mostly staying at very low frequency and being lost. Perhaps there will be an occasional `m3` mutation that rises to high frequency, or even fixes, due to

linkage with a sweeping `m4` mutation. Unless those `m3` mutations fix quickly, though, they will probably be separated from the sweep mutation by recombination and fall back to low frequency.

You might also want to look at how neutral (`m1` or `m2`) mutations look. Despite both being neutral, they display fairly different patterns because `m2` mutations tend to be tightly linked with deleterious mutations within genes, whereas `m1` mutations are often distant from any selected region.

Before moving on, you might again try right-clicking on the graph window to explore some of the visual configuration options offered; they are somewhat different for this graph than for the previous one. When you're done, close the graph window.

4. Loss time and fixation time

Now click the graphing button and open two graphs: **Graph Mutation Loss Time Histogram** and **Graph Mutation Fixation Time Histogram**. Side by side, they might look like this after the model has run to the end:

These plots are based upon data gathered through the entire model run, so if you recycle and run you will see that they start out looking rather different from the above, but converge to the above appearance over time. Try that now, and let the model run all the way to the end.

These graphs are showing the distribution of times observed for the loss or fixation of mutations of each mutation type. Consider `m4` (beneficial) mutations. The graph on the left says that *when* `m4` mutations are lost, they tend to be lost quite quickly (within the first ten generations), much like the other mutation types. It says nothing about how likely `m4` mutations are to be lost, however; it is showing only the distribution of times for the cases in which they *are* lost. (One can say that the plot is "conditional on loss".) The graph on the right says, similarly, that when `m4` mutations fix, they tend to fix more quickly than the other mutation types – within the early bins the `m4` bar is highest. Again, this says nothing about how likely `m4` mutations are to fix, but only about the distribution of times when they *do* fix; it is "conditional on fixation".

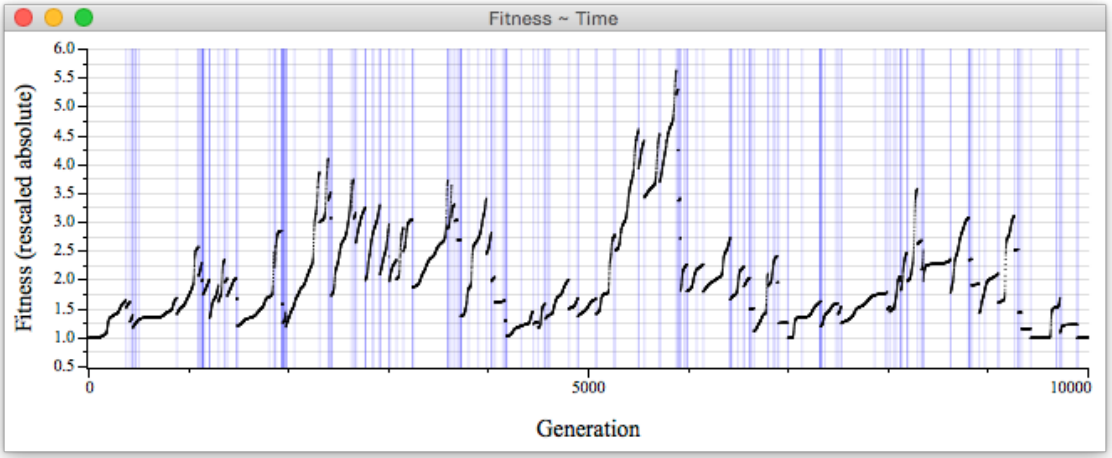
These patterns make sense; beneficial mutations ought to fix quickly, since they're selected for, whereas the other mutation types fix mostly when they hitchhike along with a beneficial mutation – an event that they might have to wait a little while for.

Play around with these graphs a bit if you wish, and then close their windows.

5. Fitness as a function of time

Recycle the model; then click on the graphing button and select Graph Fitness ~ Time. Resize this window to be quite wide; again, there will be lots of information. Then click Play.

The graph will build over time, and the *y*-axis scale will resize to accommodate the data dynamically. At the end of the model run, you might see something like this:



This graph shows the fitness of the population over time. The fitness graphed is absolute fitness, as calculated by SLiM: the multiplicative effect of all segregating mutations and other fitness effects in the model. Since this is a Wright–Fisher model, these absolute fitness values get rescaled to relative fitness internally; a high absolute fitness does not mean that the population is expanding, and a low absolute fitness does not mean that the population is shrinking (in a non-Wright–Fisher model that would be the case, though).

The axis label says the absolute fitness is "rescaled"; what that means is that mutations that have fixed, and been removed from the model, are not included in the fitness values plotted. If they were, the plot would be an almost monotonically rising curve, as more and more beneficial mutations sweep and fix; but here, each time a beneficial mutations fixes it no longer influences SLiM's calculated absolute fitnesses, and so the graph line drops. Each such point (where a mutation fixes) is shown with a blue vertical line; as you can see, those are the points at which the absolute fitness drops.

It is visually apparent that multiple beneficial mutations are often segregating simultaneously; when a blue line indicates a fixation, the fitness usually doesn't drop all the way back down to 1.0 because other beneficial mutations are still segregating.

EXERCISE: Change the mutation rate to `1e-8`, then recycle and run with the graph window still open. How do the patterns change? Why? Open the other graph windows we've looked at and see whether they look different with this lower mutation rate. You should be able to observe some interesting effects – for example, the frequency trajectories of `m1` mutations over time should clearly show the loss of genetic diversity that occurs when a sweep occurs elsewhere in the genome. Now set the recombination rate to zero with the original mutation rate of `1e-7`; this is interesting, because dominance effects lead to balancing selection among competing beneficial mutations, as can be seen clearly in the frequency trajectory plot for `m4` mutations. The model will start to run very slowly because so many mutations are segregating at the same time! Once you've observed that effect, try changing the dominance coefficient for `m4` mutations from `0.8` to `0.2` (leaving the mutation rate at `1e-7` and the recombination rate at zero). The pattern shown in the graphs will be quite different; the balancing selection is gone. Dominance effects can be very important, and can lead to very weird dynamics! Play around more if you wish; this model is an interesting testbed for a variety of evolutionary dynamics. When you're done, revert your changes (perhaps just by closing the main SLiMgui window and then re-opening recipe 7.3).

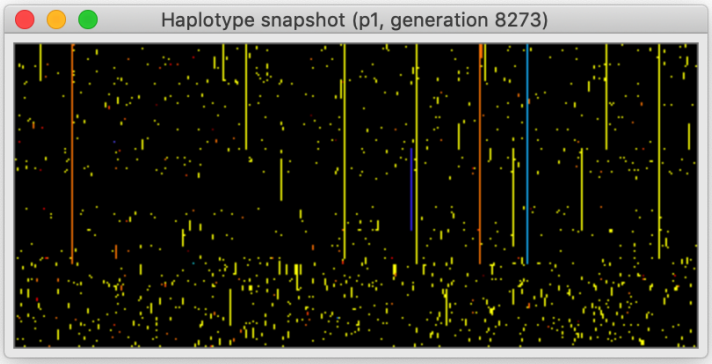
6. Haplotype plots

You should have the original recipe 7.3 model now (if not, close the window and re-open the recipe). Close all graph windows, and then recycle and run the model forward to a point when something interesting is happening, such as a beneficial mutation in mid-sweep. This might require you to play and pause the model a number of times, to catch the model at a good moment.

Once you've got the simulation paused at a moment you like, click the graphing button again and choose **Create Haplotype Plot**.

A sheet will appear providing a couple of configuration choices. The defaults are fine: "all genomes" and "greedy". (For very large models, you might want to change these settings to make the plot generation faster, but this model is not very large.) Click OK.

After a few seconds of work, with a progress panel showing how it's coming along, a plot window will appear, something like this:



We saw in the "chromosome hierarchy" worksheet that it's possible to switch the chromosome view to a haplotype-based display mode. Create Haplotype Plot does much the same thing, but produces much more detailed, high-quality plots; haplotype display in the chromosome view is based upon a small random sample of the population, and so the results can be inaccurate or just lacking in detail. Here we see the haplotype structure generated by the sweep of two beneficial mutations, one (the dark blue line near the center) on the background of the other (the lighter blue line two-thirds of the way to the right). To remind you: each horizontal row in this image represents one genome in the model, showing with colored pixels which mutations that genome possesses at positions along the chromosome from left to right. When several genomes contain the same mutation, a vertical line results, which is really just the amalgamation of the colored pixels across multiple genomes.

Lots of interesting detail can be seen here. There are neutral and deleterious mutations that have hitchhiked along with the first sweep (and a few that probably arose on the background of the first sweep, and then got caught up in the second sweep). The loss of genetic diversity due to the sweeps is apparent; there is much more genetic diversity outside both sweeps than there is within the sweeps, and the genomes in the second sweep display the least genetic diversity.

EXERCISE: Try changing the model in a couple of ways, as you did before, and then recycling, running to an interesting-looking point, and generating a haplotype plot. You should see very different patterns of haplotype structure depending upon the model parameters. Try right-clicking on the haplotype plot window to explore the configuration options available there.

7. BEEP! We're about to get more complicated, so you might review what we've already covered in previous worksheets to make sure you have a solid foundation upon which to build.