Overview

This is a set of scripts for calculating EC50 from dose-response curves generated from deep sequencing data of a large number of protein variants placed under antibiotic selection. The script is designed to start from sequencing counts of all variants, which are converted into variant frequencies. The frequencies are multiplied with the cell concentration of the sample in which the variants were selected to estimate the population size. The population size is then plotted as a function of the antibiotic concentration for a dose-response curve.

Installation

The main script, 'dms_ec50_fit.py', requires Python 3.6 or higher, and the 'pandas', 'matplotlib', 'numpy', 'seaborn' and 'scipy' python packages. There are also two helper scripts 'helper.py' and 'plot.py' which are included and need to be placed in the same directory as the main script.

Input data

The input deep sequencing count data can be found in the 'data' folder.

This script takes deep sequencing counts generated from the 'DMS-FastQ-processing' script (https://github.com/johnchen93/DMS-FastQ-processing).

To generate the data, variant count data from selected conditions (containing 'AMP' in the filename) can be taken directly from **step 2a** in DMS-FastQ-processing (Identify variants). The sequencing data from the non-selected library (without 'AMP' in the filename) are also composed of the the variant count data from **step 2a**, which is then combined with the expected variant frequencies calculated in **step 2b** (Calculate mutations expected from sequencing errors).

The OD600 data is contained in the 'VIM_library_OD600_post_6hr_sel.xlsx' Excel file. There is a separate worksheet for the OD600 of all samples (OD growth) and just the non-selected samples (no sel OD).

Usage

The script can be separated into several major functions. These functions are in the 'main()' function which will be run when the script is run from command line as follows:

```
python    dms_ec50_fit.py
```

There is an option call 'version' above the 'main()' function that is used to determine the filenames in fitting and plotting output.

For the core EC50 fitting, the following functions should be executed in the indicated order:

1. mergeGrowthPercentage()
2. fitEC50()
3. cleanUpFits()

For a more detailed explanation of all functions, continue reading below.

The 3 main functions for the EC50 calculation are as follows:

1. mergeGrowthPercentage() – Takes input count files and combines them into a single file while converting the counts into frequencies. It also performs noise filtering to remove variants in non-selected conditions that have frequencies lower than 2 × of the expected frequency from sequencing errors alone.

   This function produces 2 output files:

   a. 'variants_above_noise.txt' – A file containing the variants that are above sequencing errors.
   b. 'dms_survival.txt' – A file containing the variant frequencies of the selected conditions.

2. fitEC50(sample_step=None) – Takes the output files from 'mergeGrowthPercentage()' and performs curve fitting based on a 4 parameter sigmoidal equation, defined by the 'ECcurve' function. If a file of initial estimates is not already available, it will also make initial estimates for the fit using the 'fitGuess()'.

   If the argument 'sample_step' is set to any number greater than 0, the function will also plot samples of variant curve fits that failed for trouble shooting purposes. Every n curves (n = 'sample_step') will be plotted. Set 'sample_step' to 1 to plot every curve. Setting 'sample_step' to 'None' (default) will skip the plotting.

   This function produces up to 3 files:

   a. 'initial_guesses_v{version}.txt' – A file that contains the initial estimates for each parameter in the fit. Once generated, the script will re-use this file for future fitting attempts. The user can manually adjust individual initial values in this file to adjust the fits of individual variants.
   b. 'fitted_values_v{version}.txt' – A file that contains the final fitted values for each variant.

      This file also has some automatically generated warning about the data by 'figGuess()', which are only for the user's information and do not affect the fit. Though they are programmed in, some of these errors never occur. The warnings are as follows:

      i. 'rising pop size' – Indicates the highest concentration has a higher population size than the lowest concentration.
      ii. 'low pop difference' – The population sizes at the lowest and highest concentrations are within a 10 fold range, which is a fairly small difference.
      iii. 'immediate pop drop' – The lowest concentration has a population size that is less than 20% of the non-selected condition, possibly indicating the variant is already dead.
      iv. 'low naive pop' – The population size of the non-selected condition is less than 100,000, which may be a bit low.
      v. 'very low naive pop' – The population size of the non-selected condition is less than 1,000, which is very low.

vi. 'high end survival' – The population size of the highest concentration is greater than 25% of the non-selected concentration.

c. 'failed_fit_curves_v{version}.pdf' – File containing plots of data from variants with failed curve fits, sampled according to the 'sample_step' option. Only generated if the 'sample_step' is not 'None'.

3. cleanUpFits() – A function that takes user generated excel sheets and makes final adjustments to the EC50 values produced by 'fitEC50()'. To make use of this, see the example Excel file 'manual_fit_QC.xlsx', which is used to determine if a specific variant should be ignored, or set to the minimum/maximum value of the experimental range. The user should generate the Excel sheet based on visual inspection of all fitted curves.

   The Excel file should have two worksheets:

   a. 'failed' – Notes on particular failed fits. Does not need to include all failed variants. In the 'note' column, set the value to 'HIGH RESIST' if the fit failed because the dose response transition was above the experimental range. Any other value placed in this column will cause the particular variant to be ignored. Leaving the 'note' blank has no effect on the data.
   b. 'working fits' – Notes on what to do for specific fits. Does not need to include all working fits. To adjust the fits place a 'T' in the note columns 'missing_range', 'high_resist', 'dead', or type a comment into 'exclude_other_note'. If 'missing_range' or 'exclude_other_note' is marked, this variant's fit values will be discarded. Marking 'high_resist' will cause the variant EC50 to be set to the maximum of the experimental range, while 'dead' causes the variant EC50 to be set to the minimum of the experimental range. The 'user_note' column has no effect on curation, and is simply for the user's own record in case there are specific curves they would like to remember for later.

   For all other failed variants not included in the Excel sheet (or included but without a special note), they are assumed to have failed the fit due to dying at low selection concentrations and their EC50 are set to the lowest in the experimental range.

   The final filtered set of EC50s are saved to 'ec50_fitted_filled.txt'.

There are 2 functions that can be used to visualize the fits. The outputs can be seen in the 'example plots' folder.

1. plotSample() – Samples 50 variants from the data and plots their dose-response curves as scatter plots, using values in the 'dms_survival.txt' file generated by 'mergeGrowthPercentage()'. Intended for the user to see what the raw dose-response data is like without performing any fitting.

   Produces a single PDF with all plots named 'pop_size_plot_sample_{i}.pdf', where 'i' is simply a number that is automatically incremented if a previous file with the same name exists.

2. plotWorkingFit(sample_step=50, just_wt=False, warning='') – Plots the successful fits from the 'ec50_fitted_filled.txt' file generated by 'cleanUpFits()', including the sigmoidal curve from the final fitted parameters.

   The 'sample_step' argument indicates intervals at which to sample the data, where the default of 50 means to sample every 50th variant; set 'sample_step' to 1 to plot every variant.

   If the 'just_wt' argument is set to true, only the wt fits will be plotted. In this case 'sample_step' is ignored and all wt fits are plotted.

   The 'warning' option takes a string with the exact warning generated by the 'fitEC50()' function, and plots only variants with the given warning. Intended to isolate variants with specific warnings during initial fitting estimation. In this case, 'sample_step' works as usual.

   The function produces a single PDF with all plots named 'working_fit_curves_v{version}{wtonly}{warn_label}.pdf'. If a version is given, it will be included in the file name. Additionally, the name also indicates if the plot contains only wt fits or is filtered by a specific error.