LMU | LA
McNair Scholars Program

Test-Driven Development and Functionality Improvements to GRNmap, a Gene Regulatory Network Modeling Application

Trixie Anne Roque[1], Kam D. Dahlquist[2], Ben G. Fitzpatrick[3], and Dr. John David N. Dionisio[1]

[1]Department of Electrical Engineering and Computer Science, [2]Department of Biology, [3]Department of Mathematics,
Loyola Marymount University, 1 LMU Drive, Los Angeles, CA 90045 USA

GRNmap Uses ODE to Model Networks

GRNmap is a MATLAB software that models the gene regulatory network (GRN) dynamics of *Saccharomyces cerevisiae*, budding yeast, in response to the environmental stress of cold shock. Figure 1 shows the hypothesis network developed by our group throughout the years.

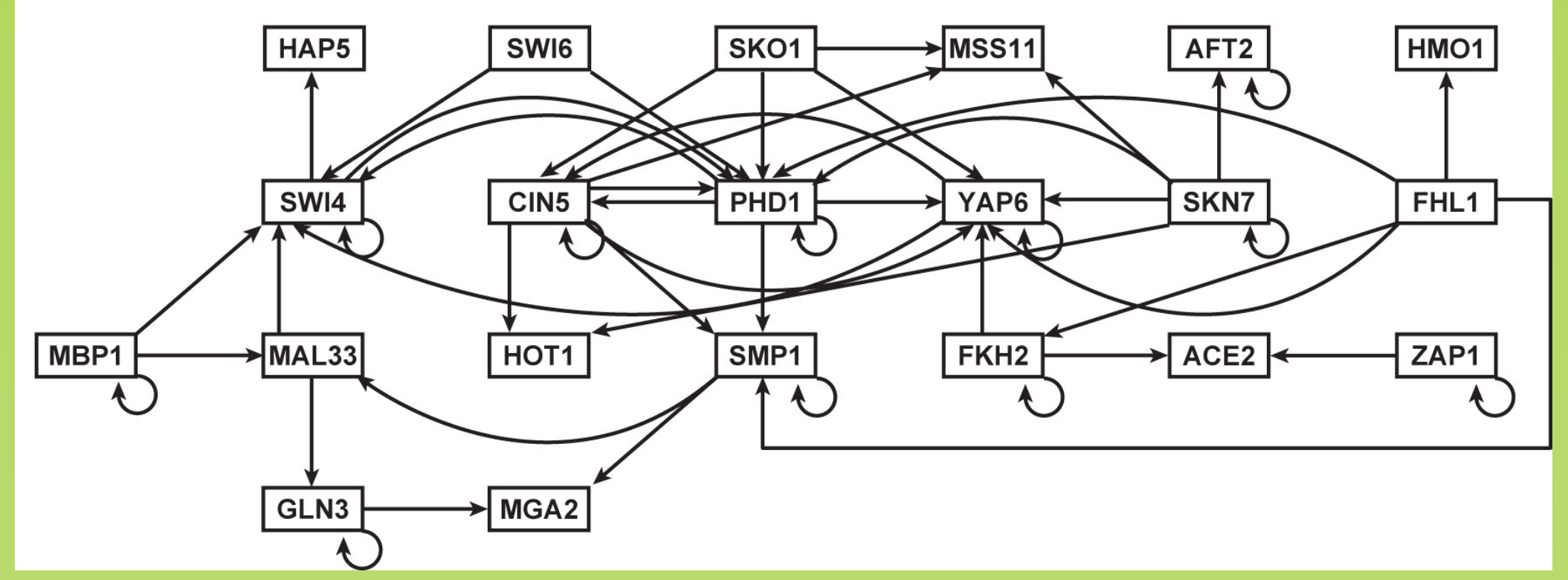

Figure 1: A 21-gene 50-edge gene regulatory network

The rate of change in expression of each gene (x_i) in the network is modeled by a differential equation (Equation 1) where *p(x)* is the production rate of the gene, λ_i is the degradation rate constant and x_i is the expression profile of the gene. We model the production term, *p(x)*, using two different models, the sigmoidal model (Equation 2) and the Michaelis-Menten model (not shown)

$$\frac{dx_i}{dt} = p_i(\vec{x}) - \lambda_i x_i$$

Equation 1

$$\frac{dx_i(t)}{dt} = \frac{P_i}{1+\exp(-\sum_j(w_{ij}x_j(t))+b_i)} - d_i x_i(t)$$

Equation 2: Sigmoidal model used by GRNmap

In the Sigmoidal model (Dahlquist et al., in press), P_i, is the production rate constant of a particular gene *i*, w_{ij} is the production weight of transcription factor *j*, and b_i is the expression threshold.

Written in MATLAB, GRNmap loads an Excel spreadsheet containing DNA microarray data provided as $\log_2$ ratios of expression for each gene in the network as inputs. It outputs another Excel sheet containing the estimated network weights, expression thresholds, and production rates.

The software makes heavy use of two MATLAB functions: ODE45 and FMINCON. We use ODE45 to solve the model's differential equation and we use FMINCON to estimate the parameters of the model using a penalized least squares fit criterion.

GRNmap Manipulates Data Provided by Our Group

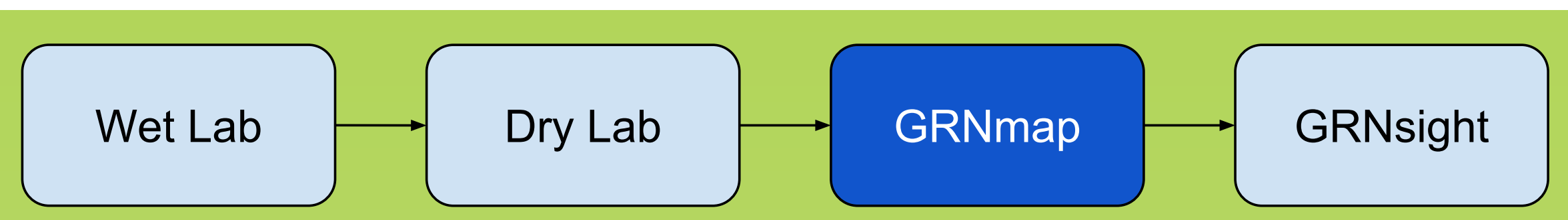

Figure 2: DNA microarray data flow

Our group consists of 4 teams in charge of generating and manipulating data as shown in Figure 2.

- Wet Lab: Generates gene expression data from DNA microarrays. Individual gene deletion experiments are conducted at this stage in order to observe how each transcription factor affects the network.

- Dry Lab: Transforms the wet lab data into $\log_2$ expression profiles for each deletion strain and works on data normalization and statistical analysis in order to generate the GRN. They compile the data into an Excel workbook.

- GRNmap takes in the Excel file. It gives the user the option to estimate parameters or just run the forward simulation. The parameters that the user can manipulate are the production rates, threshold, plots generated, and model used to solve the differential equation.
  - Over the course of its development, we have improved on the functionalities of the software package. Since v1.0.8, changes we have made to the source code include changing the names of the worksheets in the input and output workbooks, computing standard deviations, optimization diagnostics outputted for each run (contains LSE, penalty term, and iteration count), computing minimum LSE and sum of squares error of individual genes, and saving the graphs according to their gene names.
  - Routine bug fixes were also performed on the packages. They include corrected estimated production rates and threshold parameters output worksheets, corrected computation for threshold for genes with no inputs, and corrected penalty computation for production rates.
  - These changes have been documented in our external GRNmap website (http://kdahlquist.github.io/GRNmap/index.html) and GRNmap developer wiki (https://github.com/kdahlquist/GRNmap/wiki)

- The data outputted by GRNmap is used by GRNsight to visualize the network. A visual depiction of the hypothesis network developed by our group is shown in Figure 1. The network is represented as an adjacency matrix where each node represents a gene, each edge represents a regulatory relationship between the genes (activation/repression depending on the sign of the weight in the model).

Test-Driven Development Process

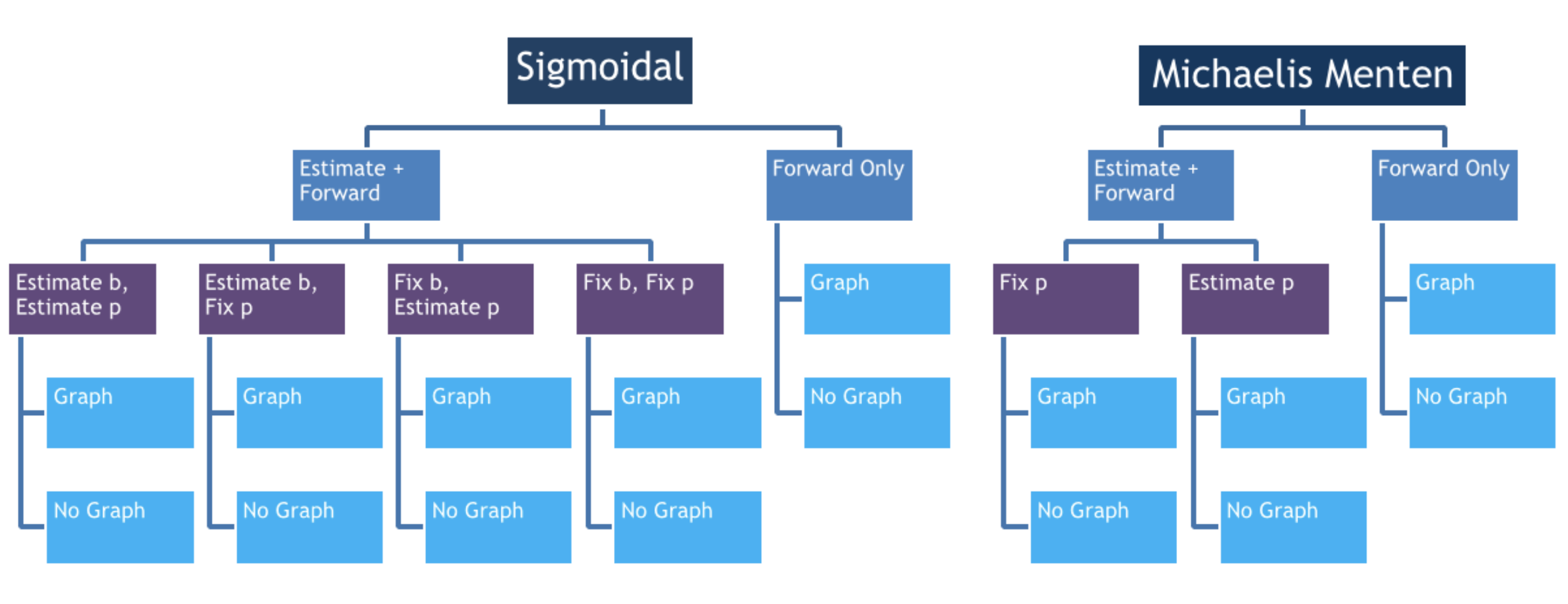

Figure 4: 16 manual test input sheets

```
DEFINE main function
    CALL functiontests (localfunctions) to make
a tests array
END

DEFINE function firstTest (testCase)
    actualOutput = evaluate function by using
known inputs
    expectedOutput = assign expected results
    VERIFY actualOutput equals expectedOutput
END
```

Figure 5: Pseudocode for unit tests

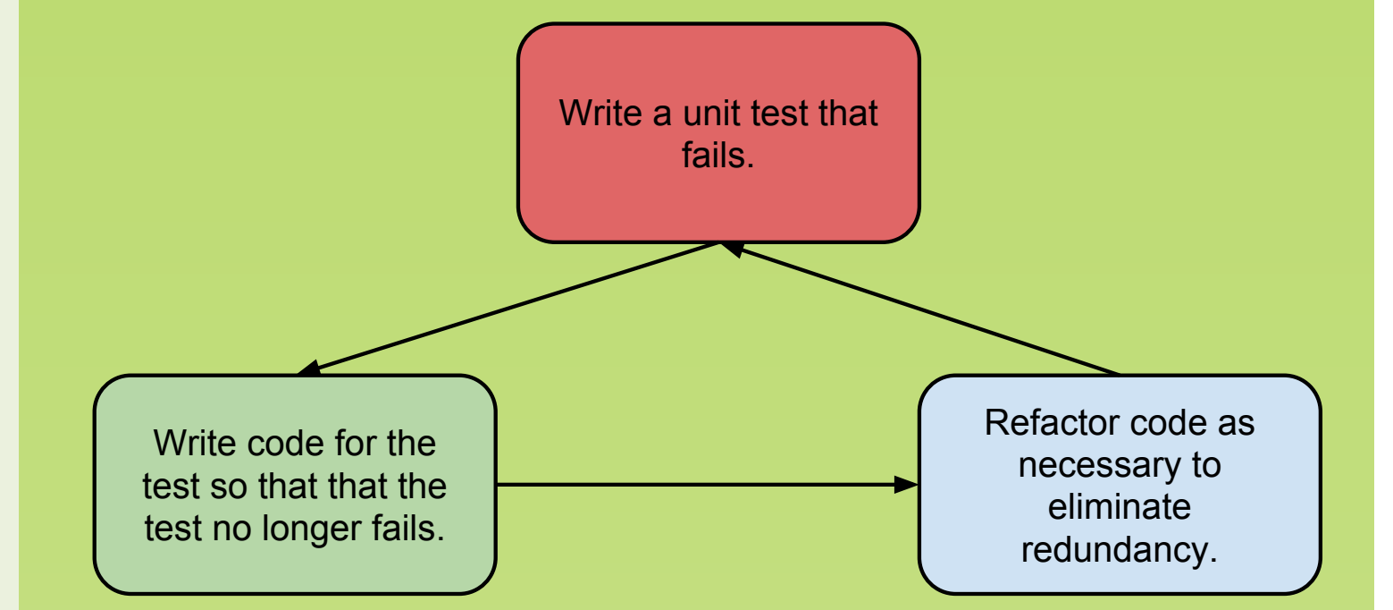
Write a unit test that fails.
Write code for the test so that that the test no longer fails.
Refactor code as necessary to eliminate redundancy.

Figure 6: Test-driven development process

The GRNmap software has been in development for over 7 years, but has only recently been refactored to a more modular function-based package and moved to GitHub where we employ its version control system. Because of GRNmap's complexity and history of switching developers, we are trying to impose test-driven development post-hoc to make debugging easier. Our approach first involved creating manual tests for every combination of optimization parameters that the user enters (as shown in Figure 4). Next, we proceed to write automated tests for these 16 different inputs. The pseudocode in Figure 5 shows what a unit test would look like when coded. Once we've finished automating these tests, we will switch over to writing the failing tests first and follow the standard procedure for test-driven development (as shown in Figure 6).

Activity Diagram Shows How Data is Processed by GRNmap

Dialog box prompts user to choose Excel input sheet
Invalid file?
[Yes]
Program gives error message and terminates
GRNstruct structure with no data inputs
[No]
readInputSheet
Read optimization parameters sheet and count the number of strains
Collect microarray data for each strain
Add simulation parameters to GRNStruct
Add control parameters to GRNstruct
Model?
[MM]
Define zero vector for thresholds as a global variable
[sigmoid]
Define initial thresholds vector as a global variable
Calculate average and standard dev for all microarray data.
Initiate additional global variables
GRNstruct populated with data from Excel sheet
lse
Unpackage control parameters from GRNstruct
Initialize vector for containing weights
Fixed pro rates?
[Yes]
[No]
Set weights in vector to given production rates for active genes
Create upper and lower bounds.
Fixed thresholds?
[No]
Set weights in vector to 0 for all forced indices
[Yes]
Calculate the least squares error of the initial weight vector
Estimate weights?
[Yes]
Calculate new optimized weights using fmincon
[No]
Store the solution arrays to log2FC. model and the times to log2FC. simtime for each strain.
Store the initial and optimized weights to GRNstruct
GRNstruct with data calculated by the lse routine
output
Include graphs?
[Yes]
Generate plots for each transcription factor
[No]
Unpack data that will be written in output sheet
Output simulation information to Excel sheets
Unpack data that will go into the . mat file
Save .mat file
Program Terminates
lse
general_least_squares_error
initial guesses (theta) vector containing weights, thresholds, and prorates
Store weights from initial guesses vector
Fixed thresholds?
[Yes]
[No]
Assign b with some of the values in theta
Fixed pro rates?
[Yes]
[No]
Assign P with some of the values in theta
Store initial concentrations to x0 vector
Store time to tspan1. If there is no 0 time, add it to tspan1
Initialize error matrix and SSE
Sigmoid model?
[Yes]
Call ode45 to solve the sigmoidal model for each strain
[No]
Call ode45 to solve the michaelis-menten model for each strain
Calculate error matrix and SSE
Output graphs every 100 iterations
Calculate sum of errors vector and assign to lse output
Initialize variable for containing thresholds
Fixed thresholds?
Use thresholds from input sheet
[No]
Fixed pro rates?
[No]
[Yes]
Zero out prorate vector
Calculate penalty using weights, thresholds, and prorate
Calculate L using alpha, penalty and previous sum of errors vector
penalized least squares fit criterion L

Future Work

- We plan to complete the testing framework for all current functionality of the code, fixing bugs and refactoring code as needed.
- We will then revise the variable names and worksheet formats so that they follow a consistent style and make intuitive sense to the user. In the future, GRNmap will automatically detect when data are present for different strains, based on the worksheets present.
- With these changes, the documentation will then be ready to be moved from our developer wiki to our front-end production web site for the users.
- We will add new functionality so that GRNmap computes the within- and between-strain ANOVA p values for the expression data, not just the standard deviations. This will allow the user to judge the quality of the expression data upon which the model parameters are based.

Acknowledgments

We would like to thank Juan S. Carrillo, Nicholas A. Rohacz, Katrina Sherbina, and Alondra J. Vega, for previous work on the GRNmap program. This project was supported by the Summer Undergraduate Research Program at Loyola Marymount University (T.A.R.), a Kadner-Pitts Research Grant (K.D.D.), and the Clarence Wallen, S.J. Chair in Mathematics (B.G.F.).

References

Dahlquist, K.D., Fitzpatrick, B.G., Camacho, E.T., Entzminger, S.D., and Wanner, N.C. (2015) Parameter Estimation for Gene Regulatory Networks from Microarray Data: Cold Shock Response in Saccharomyces cerevisiae. Bulletin of Mathematical Biology, in press.