

# Software Refactoring and Usability Enhancement for GRNmap, a Gene Regulatory Network Modeling Application

Juan S. Carrillo<sup>1</sup>, Katrina Sherbina<sup>2</sup>, Kam D. Dahlquist<sup>3</sup>, and Ben G. Fitzpatrick<sup>2</sup>.

<sup>1</sup>Department of Electrical Engineering and Computer Science, <sup>2</sup>Department of Mathematics, <sup>3</sup>Department of Biology, Loyola Marymount University, 1 LMU Drive, Los Angeles, CA 90045 USA.

## Abstract

A gene regulatory network (GRN) consists of genes, transcription factors, and the regulatory connections between them that govern the level of expression of mRNA and protein from those genes. The dynamics of a GRN is how gene expression in the network changes over time. Over a period of several years, our group has developed a complex MATLAB software package, called GRNmap, that uses ordinary differential equations to model the dynamics of a 21-transcription factor GRN from budding yeast, *Saccharomyces cerevisiae*. The program estimates production rates, expression thresholds, and regulatory weights for each transcription factor in the network based on DNA microarray data, and then performs a forward simulation of the dynamics of the network. The large number of developers and time span of development led to a code base that is difficult to revise and adjust. We therefore refactored the script-based software with global variables into a function-based package that uses an object to carry relevant information from function to function. This modular approach allows for cleaner, less ambiguous code and increased maintainability. Further revisions to the model will be also be easier to implement. In addition, we have added a simple user interface, removing the need for users to edit MATLAB code. Finally, after the code was refactored and tested, we used the MATLAB compiler to create an executable file that can be run on any Windows machine without the need of a MATLAB license, increasing the accessibility of our program.

## Mathematical Model

- Our group had previously created a deterministic model in MATLAB which modeled the dynamics of how a gene regulatory network (GRN) of *Saccharomyces cerevisiae*, budding yeast, responds to the environmental stress of cold shock.
- The GRN consisted of 21 nodes which represent the genes and the transcription factors they encode. The edges of the network represent the regulatory relationship, either activation or repression, which depends on the sign of the weight term in the model (Figure 1).

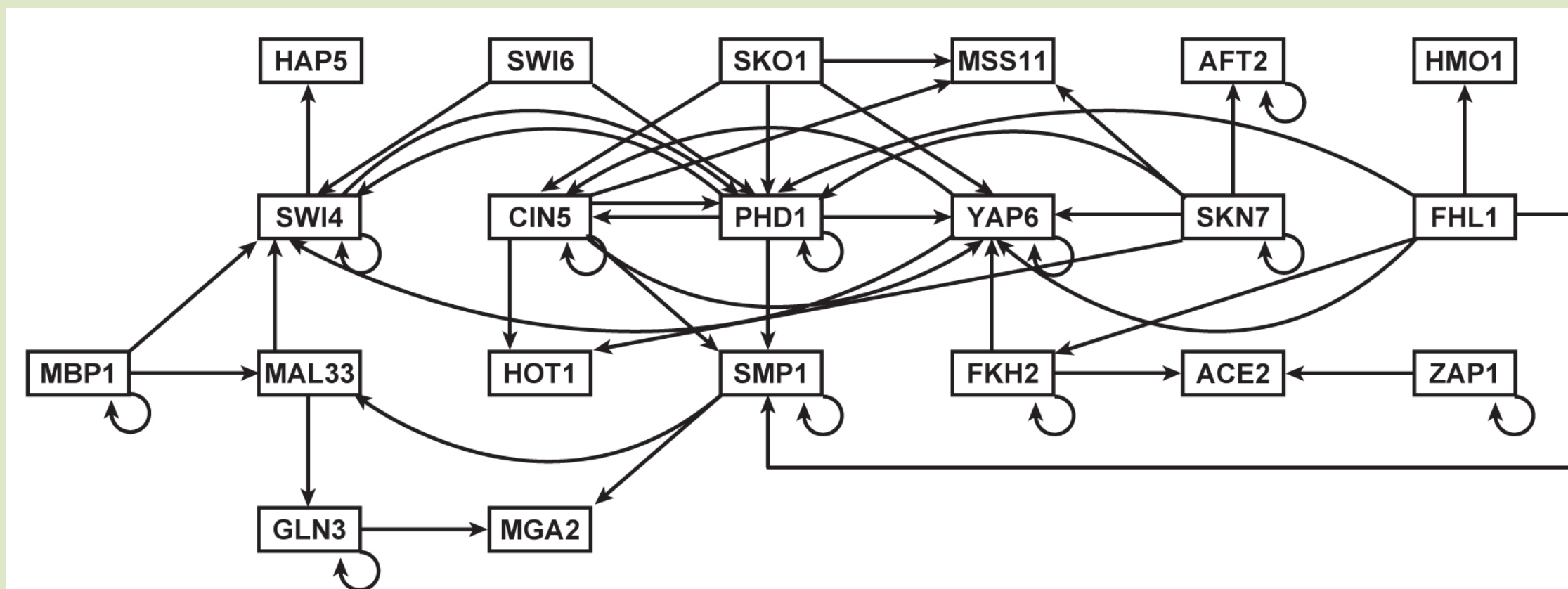


Figure 1. Gene regulatory network

- The rate of change in expression of each gene ( $x_i$ ) in the network is modeled by a differential equation (Equation 1)

$$\frac{dx_i}{dt} = p_i(\vec{x}) - \lambda_i x_i \quad \text{Equation 1.}$$

where  $p(x)$  is the production rate of the gene and  $\lambda_i$  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 sigmoid model (Equation 2) and the Michaelis-Menten model (Equation 3).

$$p_i(\vec{x}) = \frac{P_i}{1 + \exp\left(\sum_j -w_{ij}x_j + b_i\right)}$$

Equation 2.

$$p_i(\vec{x}) = P_i \cdot \left( \sum_j \left( \left[ \frac{|w_{ij}x_j|}{\sum_k w_{ik}x_k} \right] \cdot \left[ \frac{w_{ij}x_j}{1 + w_{ij}x_j} \right] I(w_{ij} > 0) \right) \right)$$

Equation 3.

In the Sigmoid model,  $P_i$  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. For the Michaelis-Menten model,  $P_i$  is the production rate of the gene, the first bracketed term is the relative weight of a gene  $j$ , the second bracketed term represents the Michaelis-Menten reaction rate, and the third term models the effects of repression.

- GRNmap takes as input DNA microarray data which is provided as  $\log_2$  ratios of expression for each gene in the network. Written in MATLAB, the GRNmap software loads an Excel spreadsheet as input. The software makes heavy use of two MATLAB functions: ODE45 and FMINCON. We used ODE45 to solve the model's differential equation (Equation 1) and we used FMINCON to estimate the parameters of the model using a penalized least squares fit criterion.

- The model estimates the production rates, weights, and expression thresholds. The model then performs a forward simulation using those parameters so that model-generated expression data can be compared to the experimental data input to the model.

- GRNmap outputs an Excel spreadsheet with the optimized parameters and resulting simulated gene expression profiles, a MAT file containing the calculated values, and plots corresponding to each gene in the network showing gene expression over time.

## Refactoring Code

- Although the original code was fully functional, it needed refactored to be more readable, modular, and maintainable.
- In order to refactor GRNmap, the program was broken down into three stages:
  - Loading the data from the spreadsheet
  - Running the model
  - Outputting the results
- Scripts and global variables were replaced with functions and a large data structure. The functions would take in the data structure, unpack any necessary data, process that data, possibly append resulting output data, and then return the modified data structure.

### GRNmap Before Refactoring

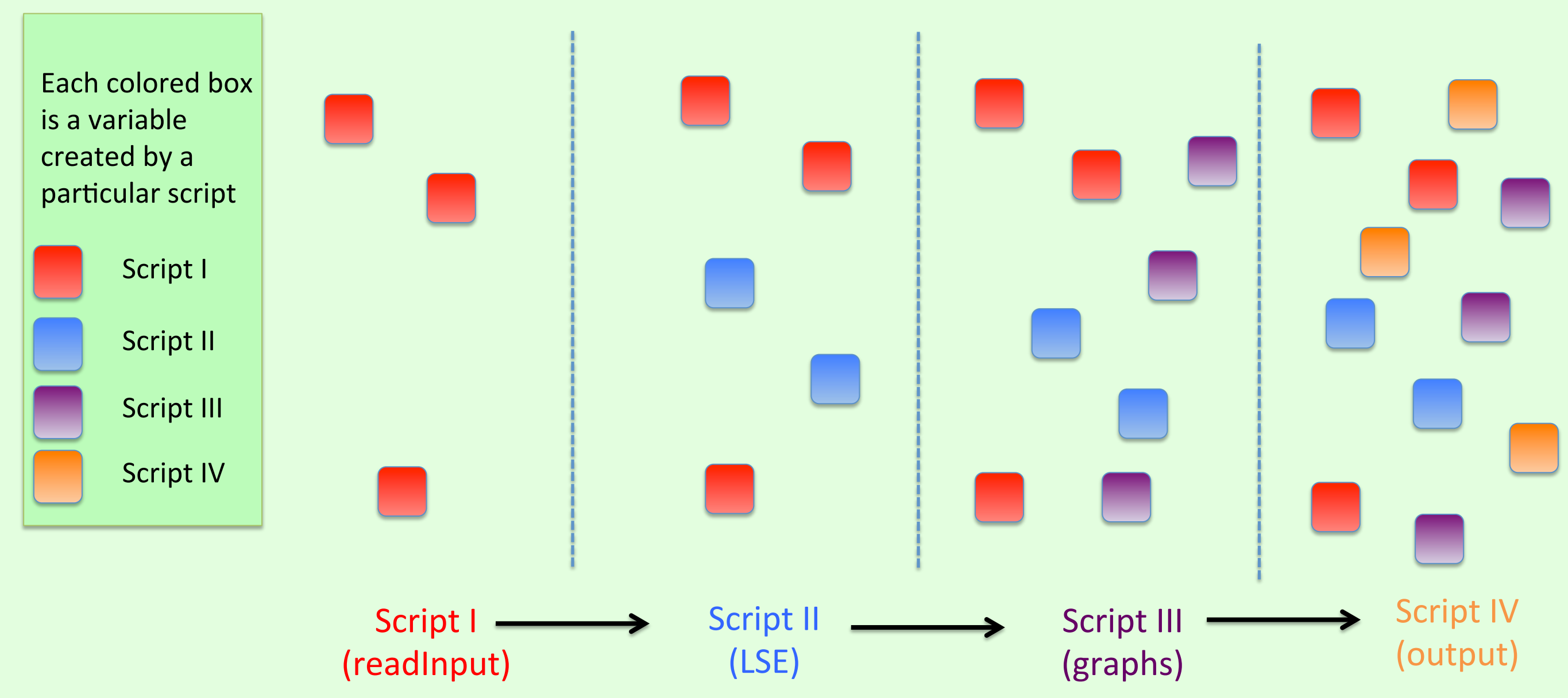


Figure 2. GRNmap's previous implementation

### GRNmap After Refactoring

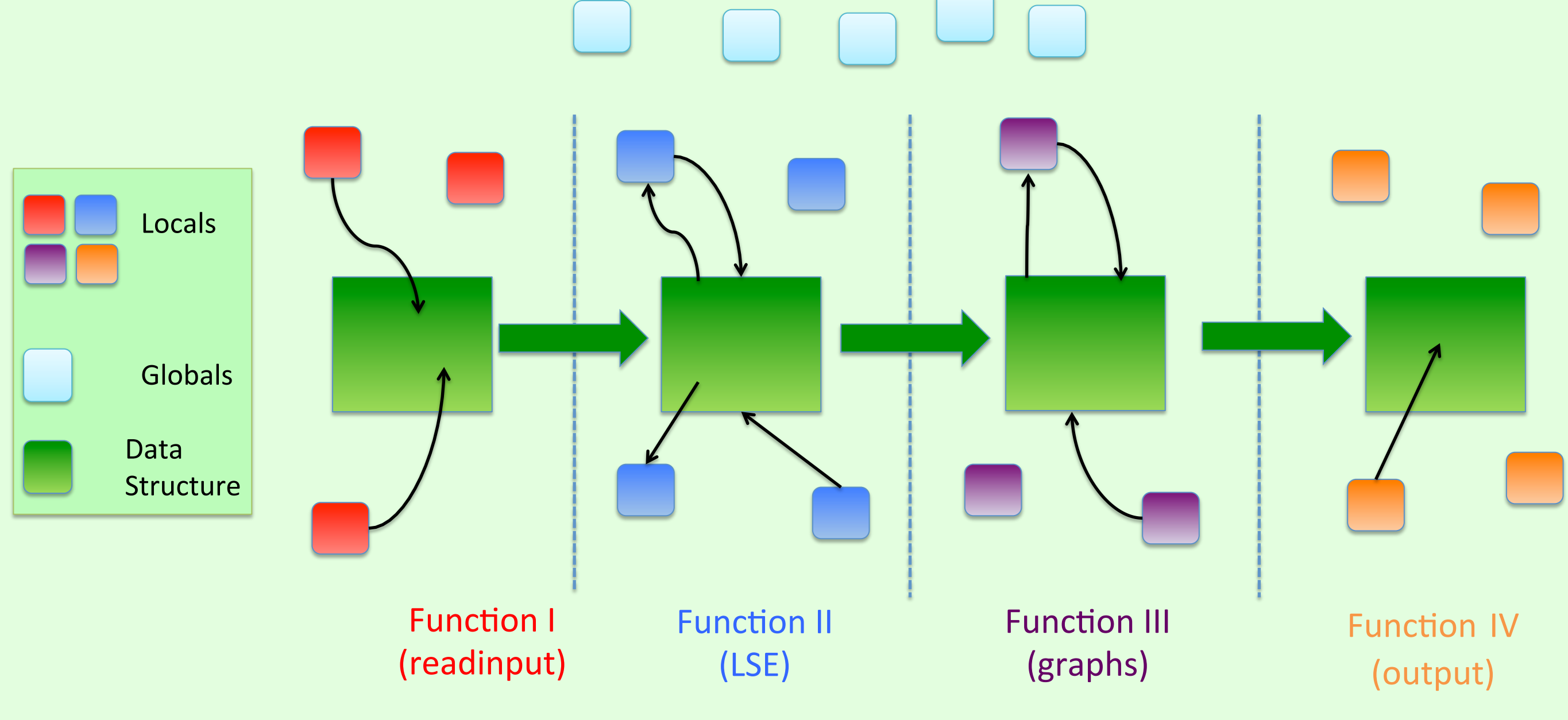


Figure 3. GRNmap's current implementation

- Note global variables persisted in memory even when they lost relevance. (Figure 2)
- The current implementation (Figure 3) makes use of local variables which exist only in their respective function. Any variables that are needed in subsequent functions are stored in the data structure object. These variables are then unpacked in the subsequent functions, used, and possibly modified and packed into the structure again.
- All parameters for GRNmap are stored in the input spreadsheet. Users can choose what model to run, whether or not they want plots, and whether or not they wish to run a forward simulation, without having to manipulate the code.
- We used the MATLAB function UIGETFILE, which opens a dialog box and allows the user to choose the spreadsheet that will be used by GRNmap.
- The new code was tested for correctness by comparing the output from the previous and current implementation.

## Availability

- In addition to refactoring the code, we used the MATLAB compiler to compile GRNmap into an executable file, which can run on any Windows machine without the use of a MATLAB license. The executable requires the free MATLAB Compiler Runtime (MRC) library.

## Running GRNmap

- GRNmap takes in its parameters directly from the spreadsheet. The spreadsheet (Figure 4) is expected to have the following information:
  - Production rates – Initial guess
  - Degradation rates – Provided by user from data
  - Expression Thresholds – Initial guess
  - Microarray data - log<sub>2</sub> fold change of expression
  - Standard deviation for data
  - Adjacency matrix to describe the graph for the GRN
  - Initial guess for the network weights
  - Simulation times
  - Optimization parameters, including which model to use, whether or not to perform a forward simulation, whether or not to set certain parameters, and whether or not to include plots for the genes

	A	B	C	D	E	F
1	optimization_parameter	value	value	value	value	value
2	alpha	0.01				
3	kk_max	1				
4	MaxIter	1.00E+06				
5	TolFun	1.00E-05				
6	MaxFunEval	1.00E+06				
7	TolX	1.00E-05				
8	Sigmoid	1				
9	estimate	1.00E+00				
10	igraph	1.00E+00				
11	fix_p	0				
12	fix_b	1				
13	time	15	30	60		
14	Strain	wt	dcln5	dglu3	dhmo1	dzap1
15	Sheet	3	4	5	6	7
16	Deletion	0	3	6	8	21
17						

Figure 4. Optimization Parameters Sheet

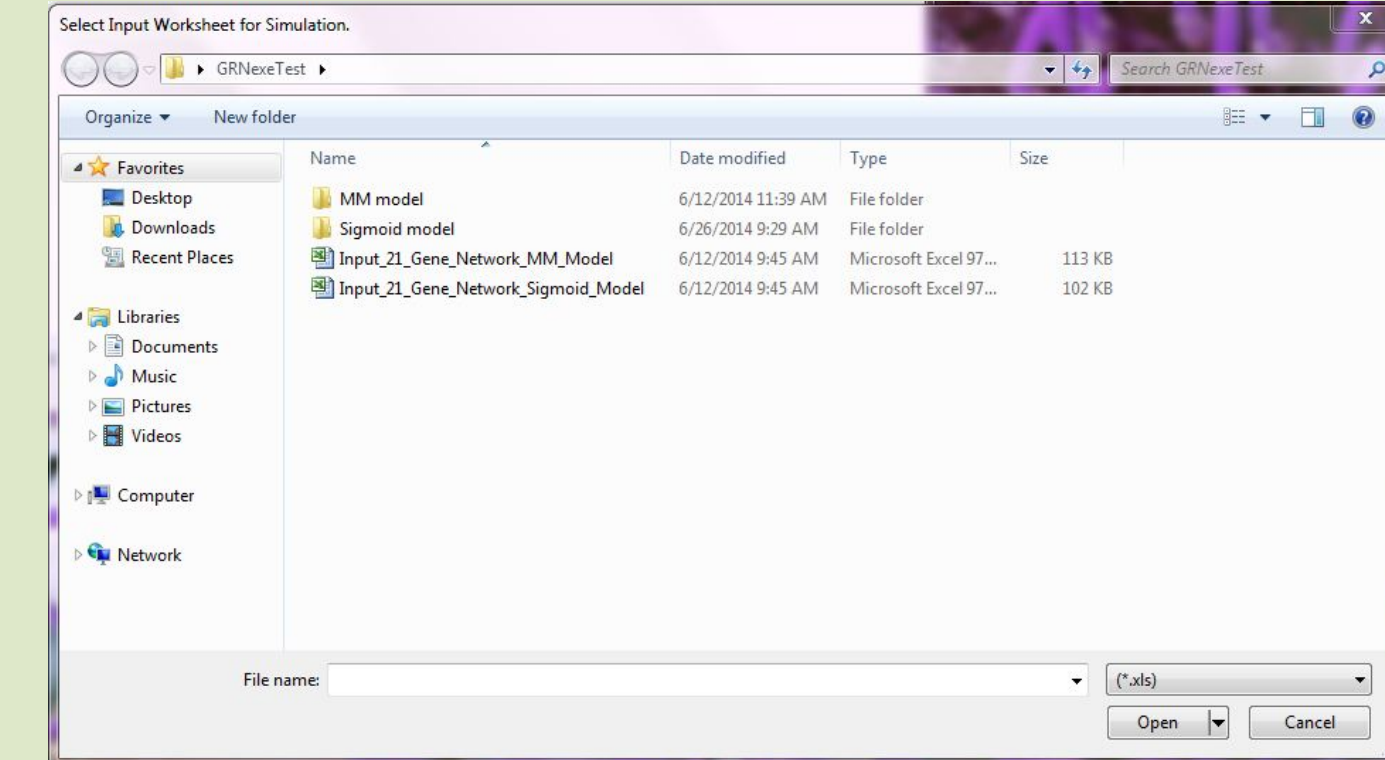


Figure 5. Dialog Box for choosing spreadsheet

- When GRNmap is run a dialog box will open to allow the user to select the spreadsheet (Figure 5). After GRNmap is finished it will output an estimation spreadsheet and .mat file. The estimation workbook contains a sheet with the optimized network weights. GRNmap will also output MATLAB figures containing the plots of the gene expression over time for each gene.

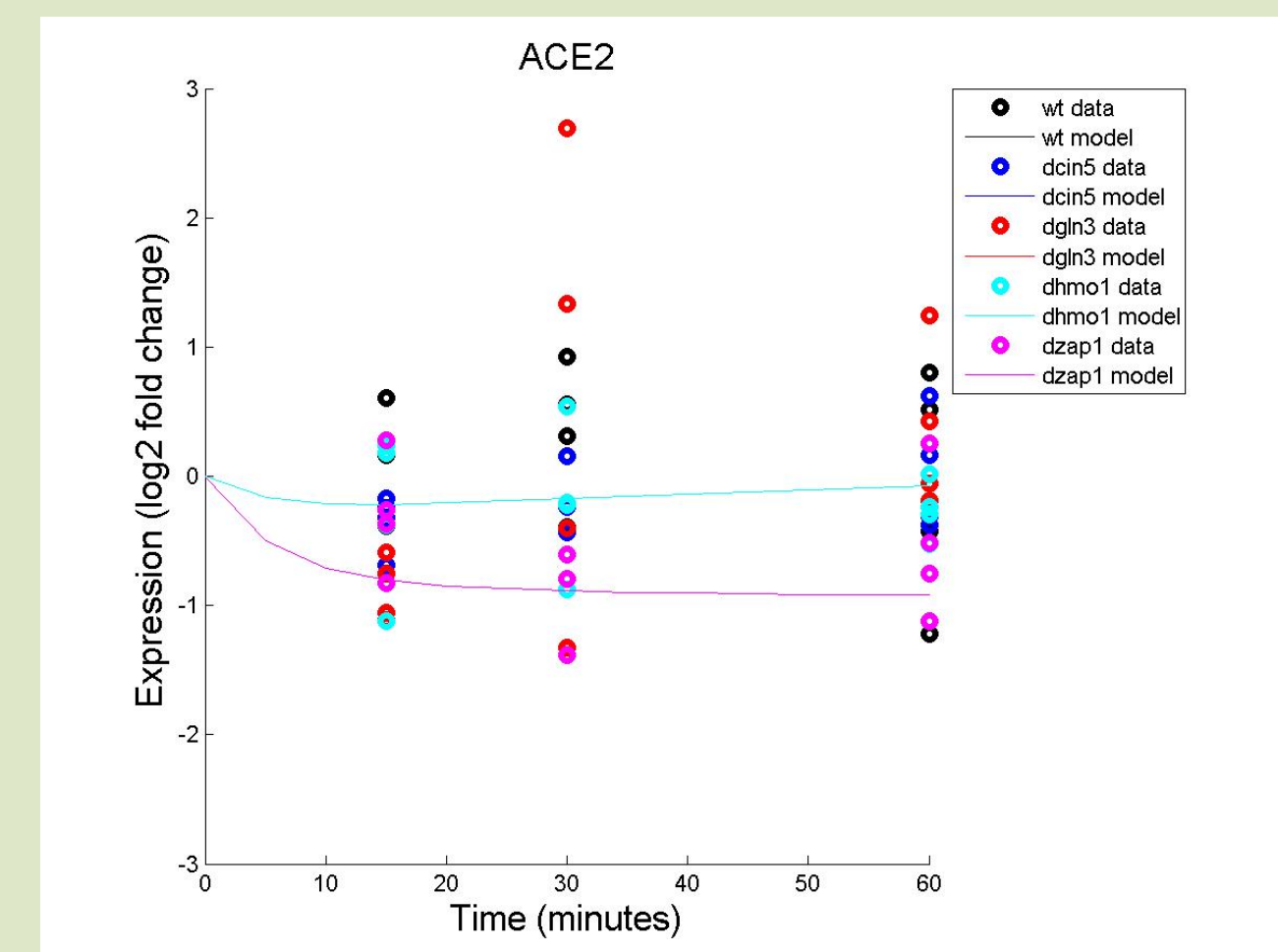


Figure 6. Sample of plot generated by GRNmap

## Summary

- GRNmap, a MATLAB program for gene regulatory network modeling and parameter estimation, had previously been developed by our group but the code needed to be reevaluated and improved.
- The new code is function-based, modular, easier to maintain, and memory efficient.
- GRNmap is user-friendly as it allows users to change parameters for the model without using MATLAB code.
- GRNmap is accessible as it can be run on any Windows machine with the free MRC library installed.
- Visit us at <http://kdahlquist.github.io/GRNmap>

## Future Work

- Making GRNmap more accessible by possibly including radio buttons and check boxes in Excel rather than typing values in cells.
- We will integrate GRNmap with GRNsight (<http://dondi.github.io/GRNsight/>), a web application and service that is being developed to visualize the results of the GRNmap modeling.

## Acknowledgments

We would like to thank Nicholas A. Rohacz, Alondra Vega, Stephanie D. Kuelbs, Nathan C. Wanner, and Erika T. Camacho for previous work on the GRNmap program. This project was supported by the Summer Undergraduate Research Program at Loyola Marymount University (N.S.C.), NSF-DMS award #0921038 (K.D.D., B.G.F., and K.S), and the Clarence Wallen, S.J. Chair in Mathematics (B.G.F.).