# Journal of Ryan Botts: Spring 2008

Ryan Botts

June 11, 2008

## 1 2 April 2008-9 April 2008: User Options and GRN's

I am now beginning work on a new project of modeling gene regulatory networks (GRN's), so my work this week has been separated into two parts. The first is attempting to model the GRN's and the second is finishing work on the project from last quarter.

This week I began writing the code to model the GRN's. The data as mentioned in last quarter's journal consists of a vector $\mathbf{X} = (X_1, X_2, \ldots X_n)$, where $\mathbf{X}_i$ represents the concentration of gene $i$. We have a collection of concentrations at various times $t$, and we may represent this as $\{X_t\}$. These substances interact with each other in some unknown manner, either promoting or blocking each other substance, hence we may describe the change in concentration of each of the components as a system of differential equations $\frac{dX_i}{dt} = f_i(X_1, X_2, ..., X_n)$. From the list of concentrations we may approximate these derivatives by using centered differences, or forward or backward differences. Recall that centered differences provide the most accuracy, but lose some data. I wrote routines this week to compute all three of these simultaneously, and output a single vector of the derivatives of each concentration. We know that both the forward and backward differences approximate the error up to an order $h$ term, where $h$ is the time step. By subtracting the backward differences from the forward differences, we obtain an approximation for the magnitude of the error in the approximation of the derivatives. I used the 5 gene network data given to me by Brandi Stigler, and found that the approximations to the derivatives were around 0.3units away from what they should have actually been. When comparing these to the actual values of the derivatives, these were reasonably good, although, we might want to get better approximations. Next week I am going to work on fitting this data better and learning more about the error terms.

This week we also worked on packaging the code for the regression of crystallographic data, actually more of Martin, and less of me. I added a list of user defined options to make it easy to control the fitting and cross-validation of the data. We also attempted to separate all routines that relied on the BCC structure and which routines would be BCC independent. It turns out that we don't use the fact that the structures are BCC very often so this was not that difficult to do. Those posing the questions asked how hard it would be to get these to fit FCC crystals so we anticipate providing this as another option. Dr. Mohlenkamp is going to finish bundling all of this to pass on.

## 2 10 April 2008-16 April 2008: Bugs make for a bad week, but a lot is learned about GRN's

Bad news this week. As Martin was checking the code he noticed that the cross-validation routine did not randomize the function before it learned every time. I learned something very valuable here that I had not encountered before: You cannot merely attempt to duplicate a variable to leave

unchanged by assigning it the name of another, which will be modified. The names are pointers, so the copy you think has been left unchanged will not be. As the cross-validation went through the training results it never re-initialized, so the function learned from one set of training data was given as the start for fitting the next training data, so we effectively never left any data out...Bad news. I need to learn how to duplicate variables in data for use later, but we were able to fix this problem by randomizing the guess function before we begin the training at each step in the cross validation. The cross validation now shows more over-fitting, which will make it necessary to get the regularization working. We can train with the regularization with a regularization factor of size $10^{-16}$, but it will only fit for cubes up to size 2. Once we use larger cube sizes the algorithm will not fit, and I cannot figure out why. I need to understand how the regularization works a little better. We have seen that the ALS is slower when using VectorDiff instead of VectorPlus as our functions, so we will also look into ways of fitting while using the speed of the VectorPlus's and the regularization from the VectorDiff.

On the other project I met with Brandi Stigler and learned many things. Overall there seem to be two big questions we must answer when we attempt to model these networks. The first is whether we can construct the wiring diagram for the gene network, and the second is whether we can accurately predict concentrations, perhaps in the long term. The data does not have many long term results so the second might be more difficult to achieve. This week I took my code from before and fit the data and then used it to predict the long term concentrations. The model we construct seems to fit the data fairly well for several time steps, but not too far into the future. This is understandable as the data we learn is not given at many time steps. The first part of the question does not seem to be too difficult as their work merely assigns edges in the wiring diagram by finding which components play a role in each function. They may then put arrows or blocks on each of the edges based on whether removing one of the components, causes more or less of another component to be produced. For example if we want to consider whether component 1 promotes or inhibits component 2 we simply look at $\frac{f_2(1,X_2,X_3,...)}{f_2(0,X_2,X_3,...)}$, if this ratio is greater than 1, we say component 1 promotes component 2 and if it is less than 1 we say component 1 inhibits. Even thought they cannot run the experiment with exactly the same amounts of components 2,3,4, and so on. As we had earlier considered integrating our learned functions and then taking partial derivatives to obtain these arrows, but then realized that the values of the functions we obtain will depend upon the concentrations of the other components, and hence the wiring diagram would not apply to all concentration levels. They do not really deal with this, but seem to avoid it by using stable concentrations and then perturbing an entry. This might allow the idea of partial derivatives to be useful after all. During this next week I plan to randomly sample the model used to generate our data and see if I can fit those data points. Next I will use the analytic model to generate more points to be used in fitting. I would like to see if having more data allows us to predict long term behavior better.

# 3 17 April 2008-23 April 2008: Identifying the involved genes

This week I simulated more of the gene regulatory network data. I produced data over longer time periods to see if I can fit larger amounts of initial data over longer periods of time. My hope is that given more data we may better identify long term steady states. I have found that the model can fit the data accurately, although I need to find a better way to compare the mse from the model to the data variance. I can fit 1000 data points generated from a known model with a mse of order $10^{-4}$ using only polynomials of order 3. I have also found that the model seems to identify which components are involved in each network without any regularization.

This coming week I will try to check whether the model may find the correct wiring diagrams including the directed arrows.

## 4 24 April 2008-30 April 2008: How our data points influence our results

One of the questions that the biologists ask of us is not only can we fit the data and figure out which genes are involved in the network, but how should I best collect the data in order to find the network. Costs dictate that we cannot collect much data, and the biology dictates that one of the only ways we can collect different clusters of data is by "knocking" genes out and moving the system to different states. This means that our data is largely grouped into different clusters, which poses many new problems for us. The first is that if we only have data in a certain cluster we are extrapolating our model to new regions, which generally cannot be done well, so what types of things might we be able to answer on these other regions. The next question is which genes to knock out in order to gather the most information. I have tried training on many sets of data and using the model to predict on new sets. Sometimes it can predict well and sometimes it can't, so the question is what separates these cases, and how can we use this to our advantage, if we can. This week left me with many questions I can not answer. Very slow going, but now I have more parts of the problem to consider. During this coming week I hope to be able to answer some of these questions and possibly find some past research on this topic.

## 5 1 May 2008-7 May 2008: Why does the regression model learn what it does?

One of the things we observed during this week is when fitting on 32 evenly spaced data points we could fit the functions values very well and that it appeared that most of the times our regression algorithm could identify which components regulated each gene. We did find that most of the time the error that was made was that an extra component was used when this was not needed. We have not yet identified why it would find unnecessary interactions. We had hoped that maybe it was due to the fact that as the data had different ranges the interesting features of the plots may have occurred outside the window that was plotted or that the plot had too large of window. We rescaled the concentrations so that they all take on values between 0 and 1. This did not solve all of the problems. One thing we noted is that all of the differential equations have two terms added together so it might be a good idea to use rank two separable functions.

During the coming week we hope to better understand what makes the model fit this type of data well and why sometimes it doesn't. I will be trying different data sets and different initializations of the fitting function.

## 6 8 May 2008-15 May 2008: Condition Numbers

This week I was a little sidetracked from understanding what causes the regression model to fit well. In trying to understand how large the regularization factor should be, I noticed that several papers mentioned that adding the regularization forces the $A$ matrix in the normal equations to be non-singular. I tried to understand why this is, and see that by adding the regularization we are adding the term $\sum_l s_l^2 \sum_k \lambda_l^k |g_l(k)|^2$ along the diagonal of $A$. This term is always positive and

removes any zeros we have along the diagonal of $A$, which will better condition $A$. Dr. Mohlenkamp had a thought that we should see if $A$ is poorly conditioned when the fit does not model the network we are trying to model. I am going to write in some diagnostic options into the ALS routine, so that we can see if there are any of these problems in $A$.

I have spent some time looking at http://www.comp-sys-bio.org/AGN/, which has a repository of benchmark networks and have been doing more background work. I want to better understand continuous fits of this type of data, how successful others have been in the past and what might be learned from the shortcomings in their models. I have also found that our data should usually be between 0 and 1 as it should represent concentrations, even though our data set does not do this for some reason.

# 7 16 May 2008-21 May 2008: Condition numbers and previous work.

I am still trying to get the code to print out the condition numbers for the $A$ matrix in the normal equations. As $A$ has a nice block structure with many 0 entries, we do not need to store all entries of $A$ to save computational time, however, in order to find the condition number of $A$ we must first find the singular value decomposition of $A$. In order to do this we must first rewrite $A$ in standard matrix form, which is what I am working on now.

During this week I have talked to Fan many times about his project of reconstructing missing data points from a matrix and have thought about how much data we can reconstruct given some amount of data. In [YT02] it is stated that one of the major difficulties in the reverse engineering of gene networks is the lack of data. A common method to overcome this is a technique called clustering, which I am still learning about. As I understand it now this technique relies on the following facts: (1) that gene networks are hierarchical, that is genes regulate mRNA, which regulate proteins and thus genes do not interact with each other, (2) gene networks are relatively sparse (few factors which regulate each one) and (3) that genes performing similar functions have similar expression patterns. For example if the expression of gene 1 is protein 1, we should see that increases in the level of gene 1 will correspond to similar increases in the expression of protein 1. With these assumptions we may break large networks up into much smaller networks, thus reducing the number of data necessary to reconstruct the data. However, data suggests that assumption (1) is not realistic as many proteins do perform some regulation of other proteins and genes also regulate other genes. Assumption (2) is the most valid of all three, although nothing about the structure of the biological systems requires this. The final assumption, (3), also seems overly simplified as it has been observed that the biological systems seem to provide checks against malfunctions by providing multiple pathways to perform the same function. I don't fully understand the technique, but many say that this model is not sufficient as it is only provides information about networks under certain environmental conditions and not any global conditions. Some argue that the networks will not behave the same globally, as we have worried before [D'h97] (find a better source for this), hence this is an unreasonable requirement of the model. On the other hand, a global model would be far more useful to biologists.

A second approach is to reduce the model to a model of linear differential equations of the form:

$$\dot{x_i}(t) = -\lambda_i x_i(t) + \sum_{j=1}^{N} W_{ij} x_j(t) + b_i(t) + \xi_i(t) \qquad \text{for } N = 1, 2, 3, \ldots N \tag{1}$$

where $\lambda_i$ is some self-degradation rates, $W_{ij}$ are interaction coefficients of gene $j$ on gene $i$, $b_i$'s are external influences on gene $i$ and the $\xi_i$ are the noise. This model seems to fit the network data

well near steady states, that is even under different initial conditions (knockout data), the model will fit global data once the systems have run long enough to stabilize. This type of model provides some global conclusions, but does not fit perturbations to the environmental conditions well. This paper suggestions that a more robust regression model would perhaps be better able to capture the dynamics of the network. This is good news as experts are asking for a technique such as ours.

In this same article it is also cited (I have not checked all of the citations, yet) that the real goal would be to reconstruct networks with hundreds to thousands of genes, yet many of the current techniques have been developed around models having fewer than 10 genes, and in these cases the models must be oversimplified in order to be fully reconstructed. It does show that much larger networks have been examined, but I need to find specific examples doing this.

# 8 22 May 2008-28 May 2008: Singular Value Decomposition and Formatting Output

This week, I was finally able to get the singular value decomposition to work. This was more difficult than expected as the $A$ matrix used in the normal equations $Ax = b$ has a natural block structure and hence is not stored entry by entry, but in blocks. The SVD routine could not handle this so we had to find a way to concatenate this matrix into standard form.

We are interested in the singular values of the $A$ matrix as the condition number of $A$, $\kappa(A) = \frac{\sigma_n}{\sigma_1}$, where $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n$ are the singular values of $A$. The condition number of a matrix $A$ tells us how errors in the vector $b$ will effect the solution to $Ax = b$. Large values of the condition number tell us that small perturbations of $b$ will have great effects on $x$. We hypothesize that the reason that our model does not allow us to recover the networks entirely is because the $A$ matrices used in the ALS are poorly conditioned. Hence we may fit the data well, but due to the fact that our data has some error, small errors in the data would cause the fit to be far from perfect.

I am still testing this hypothesis. So far, without any regularization the $A$ matrices seem to always be poorly conditioned. By considering what values of the regularization factor make $A$ well-conditioned we might be able to justify the magnitudes of the regularization parameter that seem to work.

This week I also wrote some code to identify which genes are participating in the interactions by considering the ratio between the maximum and minimum values of each polynomial in the separated representation. We should still think about whether this is the best way to determine whether there is a connection between two genes, but it is one way.

For next week I will be running more tests to understand when $A$ is poorly conditioned. I also hope to test whether fitting each gene's regulation function independently or fitting using vector valued output allows us obtain better wiring diagrams. I have also fallen a little behind in LaTeX-ing my results and literature review so I will be catching up on that.

# 9 29 May 2008-5 June 2008: What Makes the Model Find the Wrong Interactions?

One of the things that I tried this week was to determine what makes the model find the incorrect interactions. I ran the ALS until I fit the data with an error around $10^{-29}$, yet which found no interaction between the first component and itself. I then printed out the ALS diagnostics and found that the A matrix was ill-conditioned, however it appears that the A matrix is always ill-conditioned, even when the model correctly identified that component 1 regulates itself. I observed

that the regression algorithm rarely finds the proper interactions for component 1, most of the time it only identifies that component 3 interacts with component 1, and usually finds interactions between two components which are not there. I tried fitting with several different values of the regularization parameter which can still yield fits with an MSE of $10^{-20}$, but for which the the condition number of the $A$ equation is as low as $10^{10}$. As the condition number improved, there was no improvement of the fit, and I still do not know why the interaction between component 1 and itself cannot be identified. I am going continue looking into why this might be.

## References

[D'h97] P. D'haeseleer. Data requirements for gene network inference. Submitted for publication, 1997.

[YT02] M.K.S. Yeung and J. Tegnèr. Reverse engineering gene networks using singular value decomposition and robust regression. *Proceedings of the National Academy of the Sciences*, 2002.