Journal of Ryan Botts: Summer 2008

Ryan Botts

August 26, 2008

1 23 June - 29 June: Why it doesn't fit

This week I was trying to understand why we do not obtain the proper interactions. The primary focus was unidentified false negatives. We have already spent much time checking condition numbers and don't fully understand if the conditioning of the A matrix used in the ALS routine might lead to misidentifying the interactions, so we began to inspect the the data we are fitting. We wanted to understand if it was possible for there to be drastically different possibilities for the f_i^l that would fit the data equally well. To verify this we considered a rank 1 sum separable and made a plot of $(x_1^j, y_j/p_j)$. The results are given in Fig. 1. The points are suspiciously similar, so we may want to look into this further, however, as we suspected it does look likely that there would be multiple polynomials of the same degree which could fit these data points equally well.

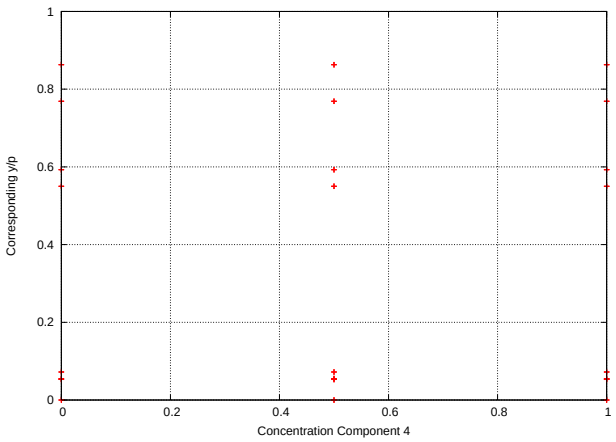
For next week I will look into these further. I have tried these plots for many different spacings of the data points and they all show suspiciously little variation. I will need to check more into this.

2 30 June - 6 July: Good results

I found several new things this week, all of them very good. First I noticed that in the plots from last week that the points were not spaced exactly as I thought. The list of points never gave a concentration of 1. A few weeks ago I had rewritten the routine for generating the list of equally spaced points so that I did not have to space the points the same in each direction, and when I did this I left out the last point, so I fixed this so that i would always have this point. When finding the plots from last week I had also reduced the fitting function to a rank 1 function and reduced the number of points to 2 in each direction, I thought 3, namely concentrations of 0, 0.5 and 1. I decided to turn the check for the condition number back on and increased the number of points to 4 in each direction, 0, 0.333, 0.666, and 1. One of the things that I observed is that the condition number for the matrix seems to change drastically as we go from 3 equally spaced concentrations in each direction to 4 and then seems to level off. The condition numbers for 2 equally spaced points was of order $10^6 - 10^{12}$, but when I increased the number of points to 3 equally spaced points it dropped to something of order $10^2 - 10^3$. I am not sure if this indicates an error in my routine for finding the condition number or if it really happening. The number of data points does effect the number of points in the data driven inner product and as I am using quadratic polynomials it would be possible that 3 points in a direction although enough to account uniquely determine a quadratic, was not enough for a polynomial in higher dimensions with maximum degree of 2. ??????? When I checked this for a rank 2 condition numbers went from something of order 10^18 when using 2 equally spaced points down to a value of 10^3 when using 3 equally spaced points.

Figure 1: W

e attempt to understand why a rank 1 sum separable would not fit the data well We plot $(x_1^j, y_j/p_j)$ for a data set with 7 equally spaced concentrations in each direction.



After noticing that the condition numbers were much better I wanted to see if this changed any of the results. Before I did this, though I decided that I needed to further develop the technique for determining if an interaction is present. Previously we proposed that in order to determine if component j is involved in regulating component i we consider the ratio $|\frac{max_{x_j\in[0,1]}f_j(x_j)}{min_{x_j\in[0,1]}f_j(x_j)}|$ in the regulation function for i. I had considered setting a threshold to determine if this was significantly larger than 1, however what I realized is that we must take the max and min prior to taking the absolute value, hence in the case where the max is a small positive value and the min is a large negative this ratio turns out to be a fraction less than 1. Previously, when considering a higher rank fitting function I would record $max_{l\in\{1\ldots r\}}|\frac{max_{x_j\in[0,1]}f_j^l(x_j)}{min_{x_j\in[0,1]}f_j^l(x_j)}|$. in which case many interactions were not significantly different than 1 simply because one of the products in one of the summands was nearly constant and the others deviated significantly, but were of the prior form, hence they were incorrectly identified as being uninvolved in the regulation of component i. I fixed this by first finding the max and the min, then taking their absolute values and finally taking the ratio of the larger to the smaller, which must be greater than or equal to 1. If we assume that we are taking maximums and minimums on the appropriate intervals we now use

$$max_{l\in\{1\ldots r\}}\frac{max\{|maxf_j^l|,|minf_j^l|\}}{min\{|maxf_j^l|,|minf_j^l|\}}. \tag{1}$$

The results were very good.

Using only the concentration levels of 0 and 1, 9 out of 10 times you will correctly identify all components involved in regulating genes 0-3, and 8 out of 10 times you will correctly identify the components involved in regulating gene 4. It appears that it is sometimes difficult to capture that a component is self-regulated. When I increased the number of data points to 3 equally spaced values

in each direction I could correctly identify the components that regulate gene in 10 trials! I might want to spend more time to see what makes this direction unique and why the self-regulation terms are the ones which seem to be more difficult to identify.

With these outstanding results I thought, well maybe it is the fact that I have nicely spaced data, so I went to the simulated concentrations as my $\mathbf{x}$ values and then evaluating the functions to obtain the y values. When using all 77 data points (7 initial starting concentrations) I could correctly identify all interactions. This was not surprise as this data contains the concentration levels of 0 and 1 for each of the components, although not at the same time. I then worked backwards and added each file to the training data set. By the the time there were 33 data points in the training data set I could correctly identify which components were involved in which interactions in every test that I ran. During future weeks I should probably try to run more tests. Maybe 100 or a 1000 and see how many times I obtain the correct interactions. I have not done this yet as I hoped to avoid setting a threshold which I considered to be a significant interaction, but this might be unavoidable. I will also note that on these data sets the condition numbers were not quite as good as before, usually around one power of 10 higher. It is also worth mentioning that although we might easily be able to identify interactions, we do not obtain as accurate of fits when using smaller training data sets.

Next week I will be focusing on my thesis proposal and writing a more formal description of these results.

3 7 July - 18 July: How the spacing of the data effects the fit

I tried fitting evenly spaced data points with values of 0,.5 and 1 for a total of 3^5 data points. I ran the fitting 100 times with rank 2 and linear polynomials and it correctly identified the interactions every time. Excellent! I increased the polynomials to 2nd degree and achieved the same results. Again excellent! When I moved to cubic polynomials, however, I began to see false positives. If you look at the regulation functions, there is always a self-degradation term (I think that is what the term that only involves the component which is being regulated is called), and in the regression models that I checked, one summand would reflect this. When things begin to behave erratically, it is in the other summand, in which all of the regulation takes place. We do have an underdetermined system at this point, and as there is nothing forcing our polynomials to be flat (no regularization, yet), It would make sense that we have multiple polynomials that would interpolate the same data, often they do become very flat, but sometimes they don't.

To see if it did have to do with the fact that we have an underdetermined system. I tried to fit data with only 2 concentrations in each direction: 0 and 1. Here false positives began to appear in the degree 2 case. I also found that if I changed the data to 2^5 data points having concentrations of 0 and 0.5, even more false positives occurred. This makes sense, as I am looking at the polynomials on the entire interval [0,1], so I have no data from which to fit near some of the edges of the interval and polynomials exhibit infinite behavior at their tails.

Something else that is interesting is that, when I use the time series data for my concentrations (but still use the actual regulation functions to fit) it appears that I can recover the entire network for any one of these degrees with only 33 data points, even though they tend to cluster around a few values. I want to run this many times to make sure that I haven't just had "good luck," but it is interesting that even though the number of data makes this a far more underdetermined system, because we don't really have any repeats of the same concentrations in any direction, we obtain much more accurate interactions. In fact the repeats seem to almost hurt us.

I have also been trying to identify something I might be able to prove for some analytical results.

I have been working on fully understanding the Weierstrass Approximation Theorem. I have also noticed that in all of the standard network models I have seen in the literature, they seem to always be a single product, at least the only term containing the interactions is a single product of functions of one variable. I wonder if I might be able to prove something like: Given an aribtrary ϵ and $f(\mathbf{x})) = f_1(x_1)f_2(x_2)\dots f_n(x_n)$, whose domain is a finite rectangle or perhaps a compact set in a measure space and where each f_i is continuous, then there is a function $g(\mathbf{x})) = g_1(x_1)g_2(x_2)\dots g_n(x_n)$ where each g_i is a polynomial and $||f-g|| < \epsilon$.

4 Why products of polynomials make good approximations

We will now show a result similar to the Weierstrass Approximation Theorem in higher dimensions. We do not need all polynomials, but may consider only those polynomials which can be factored as a product of polynomials each of one variable.

Theorem 4.1 *The space of rank 2 separable polynomials of one variable is dense in* $C[-1,1] \times [-1,1] \times \cdots \times [-1,1]$.

We will prove the result in two dimensions and note how it extends to higher dimensions.

First define the functions $K_n(x,y) = c_n(1-x^2)^n(1-y^2)^n$, for $(x,y) \in [-1,1]\times[-1,1]$ and 0 elsewhere, and where the c_n are such $\int_{-1}^{1}\int_{-1}^{1} K_n dydx = 1$. That is to say a 2 dimensional product analogue to the Landua kernel functions. We will also use the integral convolution of two functions, $f * g(x,y) = \int_{-1}^{1}\int_{-1}^{1} f(x-s,y-t)g(s,t)dsdt$.

To prove this result we will first need the following lemma.

Lemma 4.2 *If* $f \in C[-1,1]^2$*, then* $K_n * f$ *is a sum of rank 2n separable polynomial.*

Proof: We observe that

$$K_n(x-s,y-t)f(s,t) = c_n(1-(x-s)^2)^n(1-(y-t)^2)^n f(s,t) \tag{2}$$

$$= (g_0(s)+g_2(s)x^2+\cdots+g_{2n}(s)x^{2n})(h_0(s,t)+h_2(s,t)y^2+\cdots+h_{2n}(s,t)y^{2n}) \tag{3}$$

$$\tag{4}$$

where $f(s,t)$ *is distributed to the coefficients in the second product. With this fact we can see that*

$$K_n * f(x,y) = \int_{-1}^{1}\int_{-1}^{1} K_n(x-s,y-t)f(s,t)dsdt \tag{5}$$

$$= \int_{-1}^{1}\int_{-1}^{1} (g_0(s)+g_2(s)x^2+\cdots+g_{2n}(s)x^{2n})(h_0(s,t)+h_2(s,t)y^2+\cdots+h_{2n}(s,t)y^{2n})dsdt \tag{6}$$

$$= \int_{-1}^{1} (g_0(s)+g_2(s)x^2+\cdots+g_{2n}(s)x^{2n}) \int_{-1}^{1} (h_0(s,t)+h_2(s,t)y^2+\cdots+h_{2n}(s,t)y^{2n})dtds. \tag{7}$$

We can see that

$$\int_{-1}^{1} (h_0(s,t)+h_2(s,t)y^2+\cdots+h_{2n}(s,t)y^{2n})dt = h_0^*(s)+h_2^*(s)y^2+\cdots+h_{2n}^*(s)y^{2n}, \tag{8}$$

which is still a polynomial of y. Hence

$$K_n * f(x,y) = \int_{-1}^{1} (g_0(s) + g_2(s)x^2 + \cdots + g_{2n}(s)x^{2n})(h_0^*(s) + h_2^*(s)y^2 + \cdots + h_{2n}^*(s)y^{2n})ds. \quad (9)$$

Write this more clearly *As the terms $g_i(s)$ are polynomials of degree at most $2n$ in s we may apply the integration by parts formula at most $2n$ times with*

$$U = (g_0(s) + g_2(s)x^2 + \cdots + g_{2n}(s)x^{2n}), \quad (10)$$
$$dV = (h_0^*(s) + h_2^*(s)y^2 + \cdots + h_{2n}^*(s)y^{2n}), \quad (11)$$
$$(12)$$

to obtain a rank $2n$ separable function.

Now we will show that these polynomials converge to f in L^∞.

Lemma 4.3 *$K_n * f \to f$ in C^∞.*

Proof: First we observe that ss f is continuous on $[-1,1]\times[-1,1]$, f is uniformly continuous on the rectangle so there is a $1 > \delta > 0$ such that if $s,t < \delta$ then $|f(x,y) - f(x-s,y-t)| < \frac{\epsilon}{2}$. Further $\int_{-1}^{1} K_n dxdy = 1$ and convolution is commutative we have *need to correct integrals*

$$|f(x,y) - K_n * f(x,y)| = |f(x,y)\int_{-1}^{1}\int_{-1}^{1} K_n(s,t)dsdt - \int_{-1}^{1}\int_{-1}^{1} f(x-s,x-t)K_n(s,t)dsdt| \quad (13)$$
$$= \int_{-1}^{1}\int_{-1}^{1} |f(x,y) - f(x-s,y-t)|K_n(s,t)dsdt \quad (14)$$
$$= \int_{-\delta}^{\delta}\int_{-\delta}^{\delta} |f(x,y) - f(x-s,y-t)|K_n(s,t)dsdt \quad (15)$$
$$+ \int_{|t|>\delta}\int_{|s|>\delta} |f(x,y) - f(x-s,y-t)|K_n(s,t)dsdt \quad (16)$$
$$< \frac{\epsilon}{2}\int_{-1}^{1}\int_{-1}^{1} K_n(s,t)dsdt + \int_{|t|>\delta}\int_{|s|>\delta} |f(x,y) - f(x-s,y-t)|K_n(s,t)dsdt \quad (17)$$
$$< \frac{\epsilon}{2} + \int\int_{[-1,1]\times[-1,1]-[-\delta,\delta]\times[-\delta,\delta]} |f(x,y) - f(x-s,y-t)|K_n(s,t)dsdt. \quad (18)$$

If $|t|or|s| \geq \delta$, then without loss of generality $K_n(s,t) = c_n(1-x^2)^n(1-y^2)^n < c_n(1-x^2)^n$. We can estimate c_n using the fact that $\int_{-1}^{1}\int_{-1}^{1} K_n dydx = 1$. We se that

$$\int_{-1}^{1}\int_{-1}^{1} K_n dydx = c_n \int_{-1}^{1}\int_{-1}^{1} (1-x^2)^n(1-y^2)^n dydx \tag{19}$$

$$= 4c_n \int_{0}^{1}\int_{0}^{1} (1-x^2)^n(1-y^2)^n dydx \tag{20}$$

$$\geq 4c_n \int_{0}^{1}\int_{0}^{1} (1-x)^n(1-y)^n dydx \tag{21}$$

$$\geq \frac{4c_n}{(n+1)^2}. \tag{22}$$

So $c_n \leq \frac{(n+1)^2}{4}$. *We may now use this to establish a bound for* $\int\int_{[-1,1]\times[-1,1]-[-\delta,\delta]\times[-\delta,\delta]} |f(x,y)-f(x-s,y-t)|K_n(s,t)dsdt$. *As* $(1-x^2)^n$ *and* $(1-y^2)^n$ *are at their largest when* $x, y = \delta$, *and* f *is bounded by some constant, say* M, *we have the following inequality:*

$$\int\int_{[-1,1]\times[-1,1]-[-\delta,\delta]\times[-\delta,\delta]} |f(x,y)-f(x-s,y-t)|K_n(s,t)dsdt \leq \frac{M(n+1)^2}{2}\int\int_{[-1,1]\times[-1,1]-[-\delta,\delta]\times[-\delta,\delta]} (1-\delta^2)^{2n} dsd$$

(23)

$$\leq \frac{c_n M(1-\delta^2)^{2n+2}}{2(n+1)^2} \tag{24}$$

$$\leq \frac{\epsilon}{2}, \tag{25}$$

for all $n \geq N$.

Putting these together, we now see that $K_n * f \to f$ pointwise.

5 19 July - 26 July: More about how the data spacing effects the results

This week I wanted to look more into why we could correctly identify the network with 33 time series data points and needed so many more when the data were evenly spaced. The first thing I wanted to look into was the effects of removing the value of 0 and 1 from the concentration list. Recall that previously when using the concentrations of 0,0.5 and 1 for a total of 3^5 data points, I could correctly identify the network with quadratic functions every time. Using concentrations of 0.25, 0.5, and 0.75 I began to obtain false negatives and false positives, about 30% of the time. One possible explanation might be that the end behavior of polynomials is not very nice and if we only know values of the function on a small subinterval of $[0,1]$, some of the rapid growth of our polynomial might begin to occur within the interval $[0,1]$. I could correctly identify the network when I sampled the system without the endpoints and using concentrations of $1/6, 1/3, 1/2, 2/3$, and $5/6$.

I next tried to see how many random concentrations were needed to correctly reconstruct the network. With quadratic fit functions and 50 data points I couldn't correctly identify any interactions. When I increased the number of data points to 3^5 I still could not reconstruct the network. I then added endpoints to these lists to see if these had any effect. With 3^5 random data points plus the endpoints I could not correctly reconstruct the network. The sampling of the data has a

large effect on how many data points we need to reconstruct the network. It appears that there is something in the dynamics of the system that helps us recover the network better. If you look at the data values in the time series, there will be many data points when the value of the function we are learning is small and not many data points when this value is large. I am still trying to think of any reason that this might be to our benefit. I need to think of any reasons why this might help and ways to test them. We obtain the best results in the following manner, ordered from best to worst: time series data → evenly spaced data → random data with endpoints → random data.

I have also been looking into invariant measures. So far only have a definition. An invariant measure is any measure μ such that for a function f and any measurable set A, $\mu(f^{-1}(A)) = \mu(A)$. I do not yet understand its applications to dynamical systems, and I plan on looking into this a little more.

6 27 July - 2 August: Time Series Data

This week has been mostly spent doing book research. I have been looking into previous work on data sampling, and how the data can effect the results. So far I have mostly found results for one dimensional functions and not multivariate functions. The counter examples that apply for the failure of polynomial interpolation in one dimension would also hold in higher dimensions. For example, Runge's phenomenon would still occur in higher dimensions if we consider polynomials of one-dimension as multivariate functions. The Chebyshev nodes will provide better polynomial approximations by having more nodes near the boundaries. I would like to see if this concept holds in higher dimensions as well. Runge's phenomenon is an example of a situation in which a polynomial interpolates a particular set of nodes well, but does not approximate the underlying function. This situation could easily arise in higher dimensions as well. The only comfort here is that any continuous function in one dimension has a set of nodes for which the polynomials approximating the function at these nodes also provide a good approximation of the function. The Chebyshev nodes are such nodes. As in the case of one dimension I would suspect that having more nodes near the boundaries is better for fitting than having equally spaced nodes, which would likely explain why the data from the time series achieves good results. In our case we also have a fixed set of nodes, and we will increase the degree of our interpolating polynomials until we fit the nodes well. One of the difficulties that we have is that we have little control over the location of the data in the experimental data. In the upcoming weeks I would like to establish some analogues to these results in higher dimensions. I would also like to extend the results of approximating continuous functions in higher dimensions with separable polynomials to separable functions which are sums of hat functions.

7 3 August - 9 August:

This week I tried fitting the time series data and and using finite differences from this data to approximate the underlying regulation functions. The results were not very good. In some of the components false positives were given 70% of the time and in other components false negatives were given 30% of the time. Martin suggested that I make sure the time series data is accurate by generating it myself. I will check into this in the next couple of weeks. Most of my time this week I have spent looking at how to extend the results about separable polynomials to using the hat functions. Several ideas didn't work out, however the idea that seem most promising is that as we know that we can approximate any continuous function, g, well as a separable polynomial, we may obtain one such approximation, say f, where f is a separable polynomial expressed as we have expressed them before. Each polynomial in f may be well approximated by a step function, and as

the rank is finite, we can then obtain an approximation for f that is a product of step functions of one variable. Hence we would now have a good approximation of g using products of step functions. The next step would be to approximate these step functions using hat functions. I can't do this yet and need to better understand how to combine hat functions and what sums of them look like. I might then be able to directly construct a hat function which approximates these step functions on an interval. Another idea about this is that hat functions are the integrals of particular step functions, so there might be a way to approximate them using this fact. I need to spend more time thinking on this.

8 10 August - 16 August:

This week I have been working on my thesis proposal. I am very thankful that I have the final reports from previous quarters to incorporate into this. The journals have also been very useful for reminding me of things that I have already done. I would highly recommend practices such as these. I have also found that even though you have things already typed up, there is still a lot of work to combining section. I hope that I will finish this sometime during the coming week.