Non-Linear Fitting of First Order Linear Ordinary Differential Equations

Alex Tennant

I. INTRODUCTION

We explore the use of non-linear least squares in order to fit systems of first order coupled systems of linear differential equations to measured data. This document will consist entirely of synthetic data to test the performance of these techniques on minimal data sets to determine the feasibility of using these techniques with measured isotopic data of specific systems. Should these trials seem promising, these trials with synthetic data will prove useful in experimental design in order to calculate metabolic rate differences between isotopic species of biologic systems. If parameters can be determined to reasonable accuracy then it could be possible to use recovered fit parameters as a measure of kinetic fractionation, or differences in isotopic composition as a result of one-way processes. Fitting data in this way may prove useful as kinetic fractionation can be difficult to measure accurately, and even more difficult to calculate theoretical values for [1, 2]. Should this method prove promising, it may be possible to use it to infer bulk kinetic fractionation between measurable regions.

Here we will consider a compartmentalized model of what presumably represents different compartments of a mouse, represented schematically in FIG. I. Compartments can be written as a system of coupled first order linear differential equations in matrix form as in EQ.1,

$$\frac{d}{dt}\begin{pmatrix} N_c(t) \\ N_L(t) \\ N_w(t) \\ N_1(t) \\ \vdots \\ N_n(t) \\ N_s(t) \end{pmatrix} = \begin{pmatrix} -\alpha_c & & & & & & 0 \\ \alpha_c & (-\alpha_L - \alpha_w) & & & & & \beta_L \\ & \alpha_w & 0 & & & & \\ & & & -\alpha_1 & & & \beta_1 \\ & & & & \ddots & & \vdots \\ & & & & & -\alpha_n & \beta_n \\ 0 & 0 & \alpha_L & \alpha_1 & \cdots & \alpha_n & -\sum_i^n \beta_i \end{pmatrix} \begin{pmatrix} N_c(t) \\ N_L(t) \\ N_w(t) \\ N_1(t) \\ \vdots \\ N_n(t) \\ N_s(t) \end{pmatrix}, \tag{1}$$

where we note that all blank entries are zero, and that each function N is the number of atoms of a certain isotope in a location at time t and the α and β terms are transfer rates between each compartment. Immediately we can see that this problem is very sparse, and it should be noted

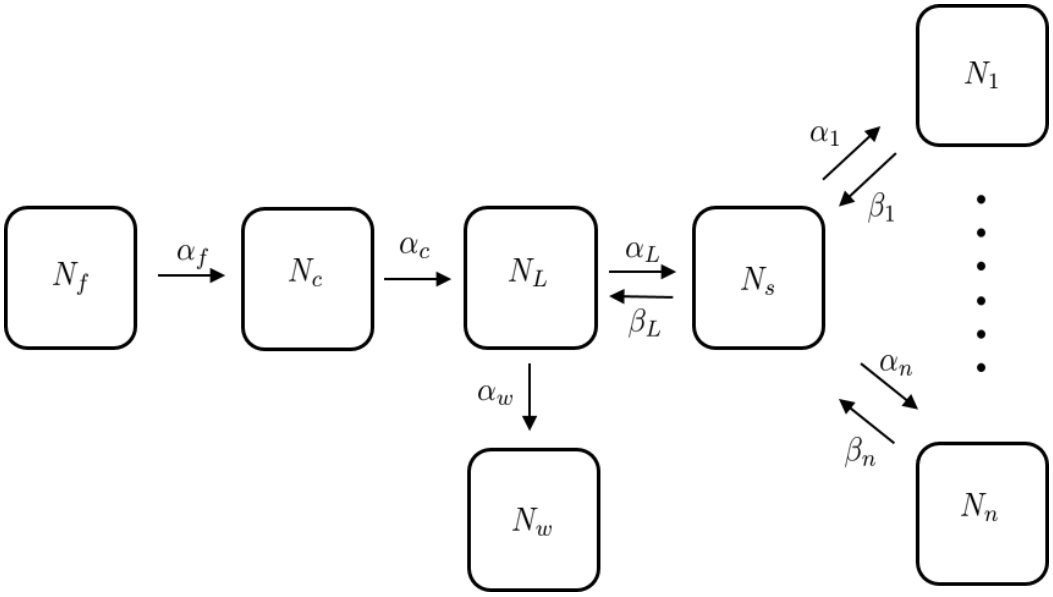

FIG. 1: A schematic diagram of the measurable regions of a mouse and their perceived couplings. Here N_x is the number of atoms of a particular isotope at some time in each region, and the α and β terms are rates to and from particular regions respectively. While the biological function of each compartment is not of importance for this document, it should be noted that N_s is the blood serum, and is the main coupling region between between most regions of the body in regards to trace metal transport.

that the eigenvalues of the problem become much simpler to calculate if the approximation of

$$\frac{dN_s(t)}{dt} = 0 \tag{2}$$

is appropriate, as the matrix becomes nearly block diagonal, greatly simplifying the calculation of eigenvalues and vectors. The problem at hand is to fit these equations to a minimal set of measurements of the various abundances $N(t)$ at different times t to recover all rates α and β in order to model the the kinetics of trace metal transport in systems which can be modeled in this way.

II. METHODS

In order to fit EQ. 1 to measured data, we will combine nonlinear least squares and numerical ODE solvers, in particular Levenberg-Marquardt to minimize the residual of the least squares fit, and LSODA, an ODE solver which uses a 4-th order Adam's method which automatically switches to the backwards difference formula should the system become stiff during optimization. These methods are popular in applications of biology and chemistry for fitting equations of a similar nature [3].

A. Levenberg-Marquardt

Levenberg-Marquardt is a local optimizer that combines both the steepest descent and Gauss-Newton methods which allows for quick and stable convergence. The so-called Levenberg step can be shown to be [3, 4]

$$p_k = -(J(x)_k^T J(x)_k + \lambda D)^{-1} J^T(x) r(x). \tag{3}$$

Where p_k is the step direction and size, λ is a positive number known as the damping parameter, J_k is the Jacobian defined as

$$J_{ij} = \frac{\partial f(x_i, \alpha_j)}{\partial \alpha_j}, \tag{4}$$

where $f(x, \vec{\alpha})$ is the model function, α is of the optimization parameters, $r(x)$ is the residual vector to be minimized, finally, D is a diagonal matrix which is often defined as

$$D = \text{diag}(J(x)^T J(x)). \tag{5}$$

The addition of λD conditions the Hessian matrix $J^T J$ to be diagonally dominant, ensuring that far from a minima the matrix remains well conditioned and numerically stable. As the damping parameter λ approaches zero, the algorithm approaches the Gauss-Newton method and converges quickly. However, if λ is large, the algorithm will take short directed steps according to what is essentially a steepest descent algorithm. If λ is updated well, this algorithm is stable and will converge to a local minimum readily in most cases. However, it cannot be stressed enough that the Levenberg-Marquadardt algorithm converges to local minima only. If global minima are of interest, it is important the initial guess at a solution is relatively close to the global minima.

B. LSODA

During optimization in nonlinear least squares, the system defined by EQ. 1. will more than likely become stiff at certain points of evaluation. It is difficult to predict in advance as to when during optimization the system will become stiff, and as such it important to choose an ODE solver which can automatically switch between methods appropriate for both stiff and non-stiff problems. A popular choice for this problem is LSODA, or the Livermore Solver for Ordinary Differential equations, standard in the ODEPACK libraries. LSODA solves an initial

value problem of the form

$$\frac{d\vec{y}(t)}{dt} = \vec{f}(t), \tag{6}$$
$$\vec{y}(a) = \vec{g}(a),$$

but to ensure stability and efficiency, LSODA switches automatically between Adams method and Backwards Difference Formula (BDF) for non-stiff and stiff problems respectively.

1. *Adams Methods*

An Adams method aims to solve the problem of EQ. 6. with an iterative approach of the form [5]

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_i+1} y'(t)dt = y(t_i) + \int_{t_i}^{t_i+1} f(t, y(t))dt. \tag{7}$$

Where i is an integer and the solution of $f(t, y(t))$ is unknown as $y(t)$ is still undetermined. However, it is reasonable to assume that within $[t_i, t_{i+1}]$ we can approximate $y(t)$ with an interpolating polynomial P such that

$$y(t_{i+t}) \approx \int_{t_i}^{t_i+1} P(t)dt. \tag{8}$$

In principle, any interpolating polynomial can be used but a common choice is to define the interpolating polynomial in terms of the system by the Newton Backwards Difference formula for a polynomial of order $m-1$

$$P_{m-1}(x) = \sum_{k=1}^{m-1}(-1)^k \binom{-s}{k} \nabla^k f(x_n) \tag{9}$$

where

$$\nabla f(x_n) = f(x_n) - f(x_{n-1}), \tag{10}$$
$$\nabla^k f(x_n) = \nabla^{k-1}(\nabla f(x_{n-1})), \text{ for } k \geq 2, \tag{11}$$

and s is a dummy parameter for the order of the interpolating polynomial. It can then be shown that upon making the change of variables $t = t_i + sh$ that

$$\int_{t_i}^{t_i+1} f(t, y(t))dt = \sum_{k=0}^{m-1} \nabla^k f(t_i, y(t_i))h(-1)^k \int_0^1 \binom{-s}{k} ds, \tag{12}$$

where h is the step size to be taken. Using EQ. 12, we may now express our solution as

$$y(t_{i+1}) = y(t_i) + h\sum_{k=0}^{m-1} \nabla^k f(t_i, y(t_i))(-1)^k \int_0^1 \binom{-s}{k} ds, \tag{13}$$

where one is free to chose the order k of any degree that they like. However, typical choices of k are 3 or 4, the choice of k determines the order of the Adams method used. A fourth order Adams method requires the previous solutions $y(t_{i-4})...y(t_{i-1})$, as such to begin any solution using an Adams method, the first solutions are often obtained by other means such as Runga-Kutta.

2. Backwards Difference Formula

A system of differential equations is said to be stiff in the case where the ratio of its largest and smallest eigenvalues becomes large, mathematically

$$\frac{\lambda_{max}}{\lambda_{min}} >> 1. \tag{14}$$

In cases where this ratio is large, the system becomes poorly conditioned and is often subject to large numerical errors if care is not taken to approach the system cautiously and deliberately [5]. BDF solvers are similar to Adams methods discussed above with the exception that the stepsize h is often significantly smaller than the Adams method and the interpolating polynomial is chosen for greatest stability. For reference, the BDF formula for arbitrary order n can be written as [5]

$$\sum_{k=0}^{n} a_k y(t_{i+k}) = h\beta f(t_{n+s}, y(t_{i+s})), \tag{15}$$

where a_k and β are chosen so that the method can achieve the highest possible order. Typically, the higher the order the more numerically stable the solution obtained is. The consequence of the stability of BDF methods is that the stepsize h is typically small and that more points are needed for interpolation. LSODA attempts to balance this by switching to a much faster Adams method with adaptive stepsize for instances where the system is well conditioned, achieving a balance between speed and stability.

III. FITTING

In order to experiment with the fitting procedure to determine the most appropriate sample times, a synthetic data set was generated from EQ.1 by assuming rates and initial conditions generated using a random number generator. Random Gaussian noise with $\mu = 0$ and $\sigma = 0.25$

was added to each point of the data set to provide noisy data with which to fit. While typical measurement noise is significantly less than this, with precision to a minimum of three decimal places, this noise was rather arbitrarily chosen in attempt to model the worst-case scenario in terms of analytical measurement. Ideal minimal "sampling" times were determined through trial and error and found to be best when the sampling points capture the actual dynamics of the equations before the system reaches equilibrium, in this case typically between $0 < t < 4$. A single measurement of the system when it has reached equilibrium seemed to be sufficient to ensure the equilibrium results matched the generated data set as best as possible. The results of such a fit are displayed in FIG. 2.

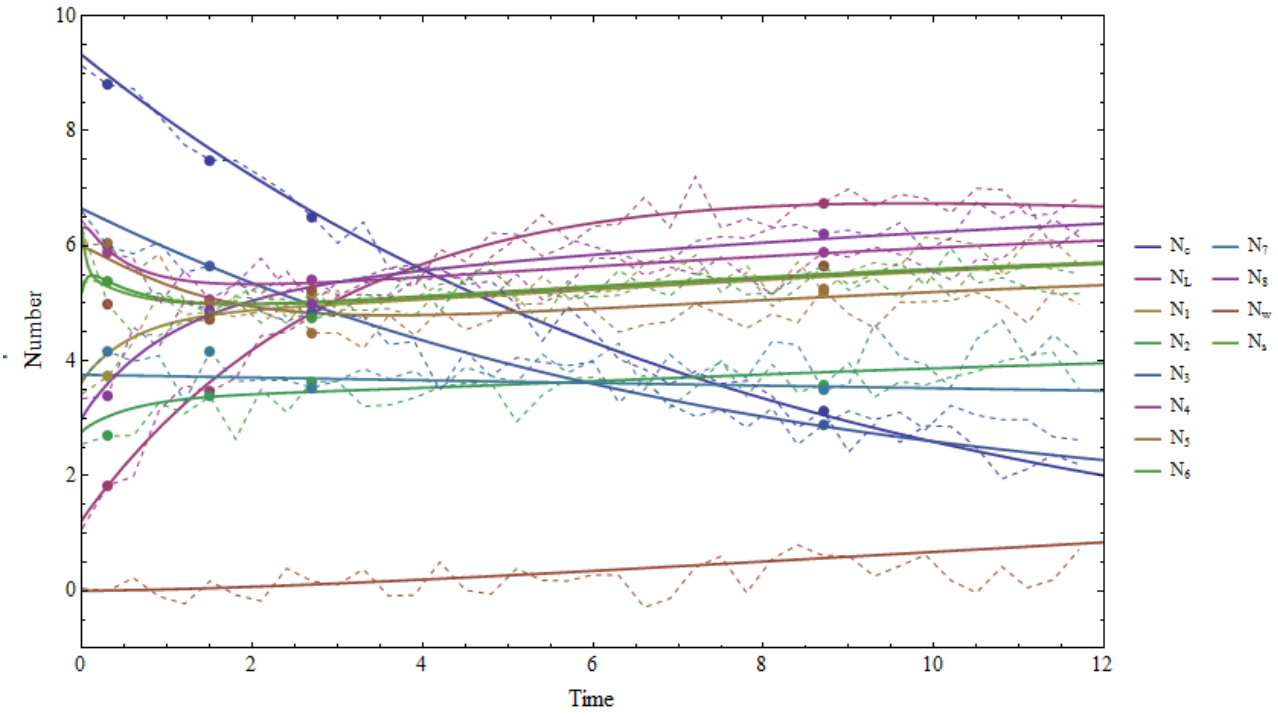

FIG. 2: The best estimated fit of noisy data of EQ.1. Not shown in the legend is the points used for the actual fitting procedure shown as large dots, and the full noisy data set shown as dashed lines. Each of these quantities are in the same color as that shown in the legend.

With reference to FIG. 2 we see that the fitting procedure does an admirable job of producing results close to the data on a minimal set of points. It is also worth noting that the region of N_w was not found using any measured points, simply using a known initial condition arbitrarily set to zero. If the initial state of N_w was not known however, it would still be possible to fit an equation to it, however its initial concentration would also become an optimization parameter. While this fit appears to be quite optimal, the quantity of interest is the rate coefficients α and β, as such the next section's focus on the determining accuracy to which our known rate parameters are recovered.

$N_c(0) = 11.23$	$N_L(0) = 7.83$	$N_1(0) = 9.61$	$N_2(0) = 9.53$	$N_3(0) = 4.35$	$N_4(0) = 7.76$	$N_5(0) = 11.60$	$N_6(0) = 8.51$	$N_7(0) = 9.42$	$N_8(0) = 5.79$	$N_w(0) = 0$ $N_s(0) = 6.85$
$\alpha_c = 0.125$	$\alpha_L = 0.245$	$\alpha_1 = 0.749$	$\alpha_2 = 0.441$	$\alpha_3 = 0.172$	$\alpha_4 = 0.923$	$\alpha_5 = 0.712$	$\alpha_6 = 0.590$	$\alpha_7 = 0.850$	$\alpha_8 = 0.778$	$\alpha_w = 0.007$
	$\beta_L = 0.240$	$\beta_1 = 0.790$	$\beta_2 = 0.312$	$\beta_3 = 0.07$	$\beta_4 = 0.980$	$\beta_5 = 0.658$	$\beta_6 = 0.600$	$\beta_7 = 0.640$	$\beta_8 = 0.891$	

TABLE I: The initial conditions and parameter values used to generate the synthetic data for the bootstrapping procedure. Please note that the initial conditions of this system were also subject to the addition of random Gaussian noise when these values were passed to the nonlinear least squares fitting routine.

IV. BOOT STRAPPING OF WELL DEFINED DATA

Here we investigate the quality of the recovered parameters with respect to how many data points are used in the fitting procedure. To do this we will use a typical method of testing accuracy of our sample estimates known as bootstrapping, where by the system is solved many times as different random noise is added to the data set before each fit. As the section title suggests, what we mean by well defined data is that the system of equations used to generate the data is the same used to fit it after the addition of random noise. This should supply us with a best-case scenario for recovering the parameters as the fitting system matches the generating functions exactly. However, it should be stated that using the model to fit back to itself is incredibly artificial as there are no other factors contributing to the evolution of the functions, a situation that will likely never be encountered using real data. As this represents the unique situation of having the best possible data, this boot strapping procedure should establish how many points are required to recover input parameters to reasonable accuracy in the most ideal case possible. The following is representative of generating 500 simulated data sets each with different random Gaussian noise ($\mu = 0$ and $\sigma = 0.25$) added to each point before the fitting procedure. In addition, no points from $N_w(t)$ were used with the exception of an initial condition arbitrarily set to zero. Table I is a list of all the initial conditions and values for parameters used to generate the data. It is important to note that these rates were chosen to be between 0 and 1 to ensure that the Jacobian matrix for the least squares procedure is as well conditioned as possible.

When dealing with two dimensional data it is often convenient to represent the 95% confidence region of two-dimensional subdivisions of that data as an ellipse. This ellipse represents a two dimensional Gaussian curve around the two data sets centered on the mean of the two data set. This ellipse between p data sets of sample size n is defined as [3]

$$n(\bar{y} - \mu)' S^{-1} (\bar{y} - \mu) \leq \chi^2_{N_p}(\alpha) \tag{16}$$

where y is the total data set, μ is the standard deviation, S^{-1} is the inverse covariance matrix of all data sets, χ^2 is the typical χ^2 distribution with N_p degrees of freedom and $(1 - \alpha) * 100\%$

determines the confidence region. While this defines a p dimensional hyper-ellipse the above relation can still be utilized to create 2-dimensional confidence ellipses comparing the results of the fit parameters of EQ.1. This ellipse can be thought of as a planar cut out of a two dimensional Gaussian distribution defined by the variance and covariance of each data set. To determine which pairs of parameters to focus our comparison, the correlation matrix of the fit was used to determine which parameters would be most sensitive to a change in another. Unsurprisingly, the α and β terms of a specific region were most highly correlated, and as such, the focus of our analysis will be on the comparison of these complementary parameters.

In order to determine the minimum sample points required for "reasonable" fit parameters, the 95% confidence ellipses between a fit of 3, 4, 5, 6 and 7 points were placed on the same axes to in order to compare them directly. These results were "filtered" in that any data point more than 2 standard deviations away from the mean of each given data set was removed in order to define the ellipse. The data was filtered as large erroneous data points are resultant of the Levenberg-Marqardt algorithm converging to a local minima far from ideal, meaning that the initial guess to those sets needed to be refined further. Unfortunately due to the automated nature of this simulation, the initial guess was not always ideal. On average 8 of the 500 data points were removed. This bootstrapping experiment, shown in FIG. 3, was quite informative, and determined that using only 3 data points has such a large confidence region that it cannot be observed on the same scale as using more data points in the fit, easily allowing us to reject using three data points as a suitable minimum number of points. Once four measurement points were used the 95% confidence interval shrinks significantly compared to the three point case.

From FIG. 3, it appears that using five data points appears to be the most effective minimum for recovering our rate constants to a somewhat constrained region. The confidence region for five data points is well constrained in most cases and there is not much to gain by using all seven points, and as far as possible measurements with a real system, having a total of five is optimistic, but reasonable. The actual data as a result of the bootstrapping procedure with five points is presented in FIG. 4. This figure shows the relative spread of recovered solutions are quite well defined within the ellipse with very few outliers. Some of these confidence intervals also include unphysical negative regions, however this is not possible as EQ. 1 is constrained to always be greater than zero. In these cases, the distribution of points is likely not Gaussian as assumed by the ellipse. It is worth noting that the six point simulation tends to have a larger confidence region than that of the four or five point test. This is likely a result as the sixth point used in the fit is near the transition of the system from change to equilibrium. This raises an interesting point as to the ideal time to sample, however this discussion is left for section VII.

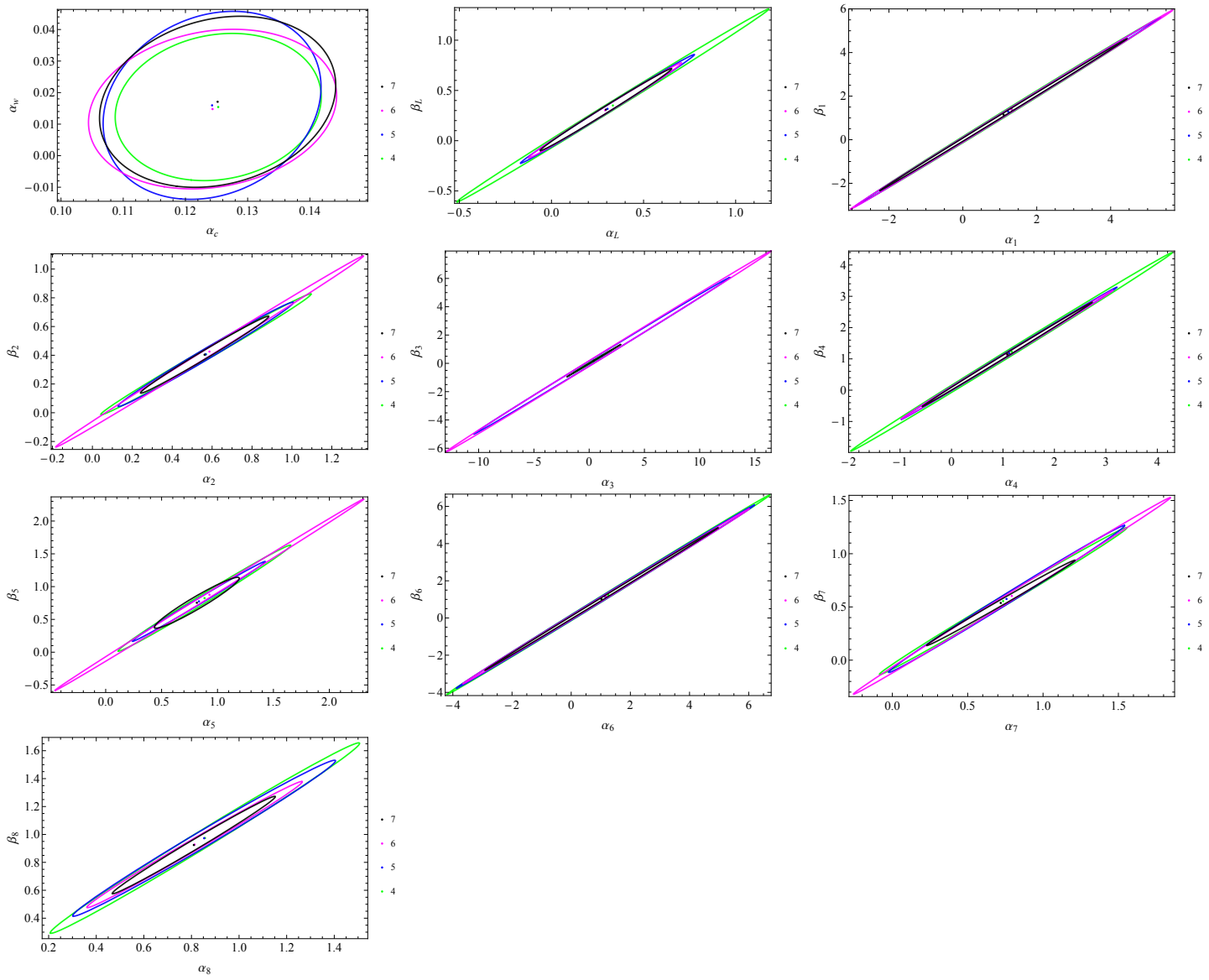

FIG. 3: Here we see relative uncertainty ellipses of 500 fit simulations with different Guassian random noise added to the system. Each ellipse represents the 95% confidence region of each data set. The individual data sets have been neglected for clarity. The confidence ellipse of the 3 point simulation is too large to be seen on this scale.

V. BOOT STRAPPING WITH LESS IDEAL DATA

In the previous section, the fitting function was exactly the function that was used to generate the data. While this is an ideal test situation, it is hardly a test that should be used in order to determine an approximate range in uncertainty in fitted parameters. This section deals with the much more honest, yet still optimistic, case where the data was generated in accordance to EQ.1, however the function used to fit this data ignored the existence of $N_5(t)$, and its parametersα_5, and β_5. Not only is this a slightly more realistic test case, but it will also be useful to observe the effect of using a model which neglects a major metabolic region. $N_5(t)$ was chosen as it has one of the largest initial concentrations and one of the largest differences between

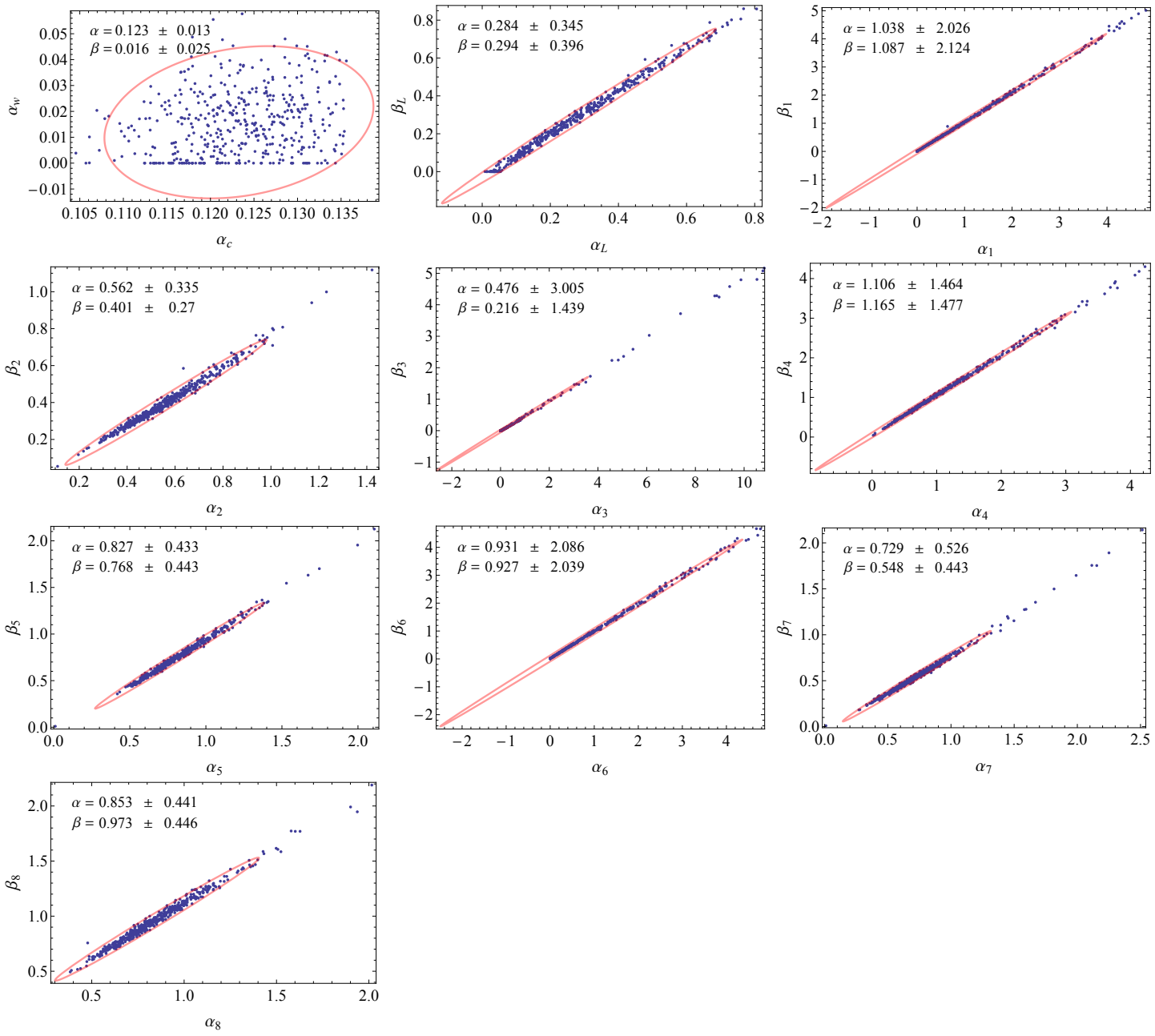

FIG. 4: Displayed here are the results of fitting the system five hundred times using only five data points. The values displayed on each plot are the mean of the bootstrapping procedure with a relative uncertainty reported as $2s$ which displays the range of the ellipse. The small width of the ellipse along its major axes is indicative of the high correlation between these parameters.

rates in hopes that it would have the greatest effect on the overall system. As with the previous case, all point and initial conditions are subject to the addition of random Gaussian noise (with $\mu = 0$ and $\sigma = 0.25$) in order to assess the effect of poor measurement of each data point. Finally, it was again assumed that no measurements of $N_w(t)$ were obtained. FIG. 5 displays the 95% confidence region of the fit parameters. The results of this bootstrapping procedure are similar to that of FIG. 3 in that the regions are comparable.Also included is a more instructive plot of the results a five point bootstrapping procedure in FIG. 6.

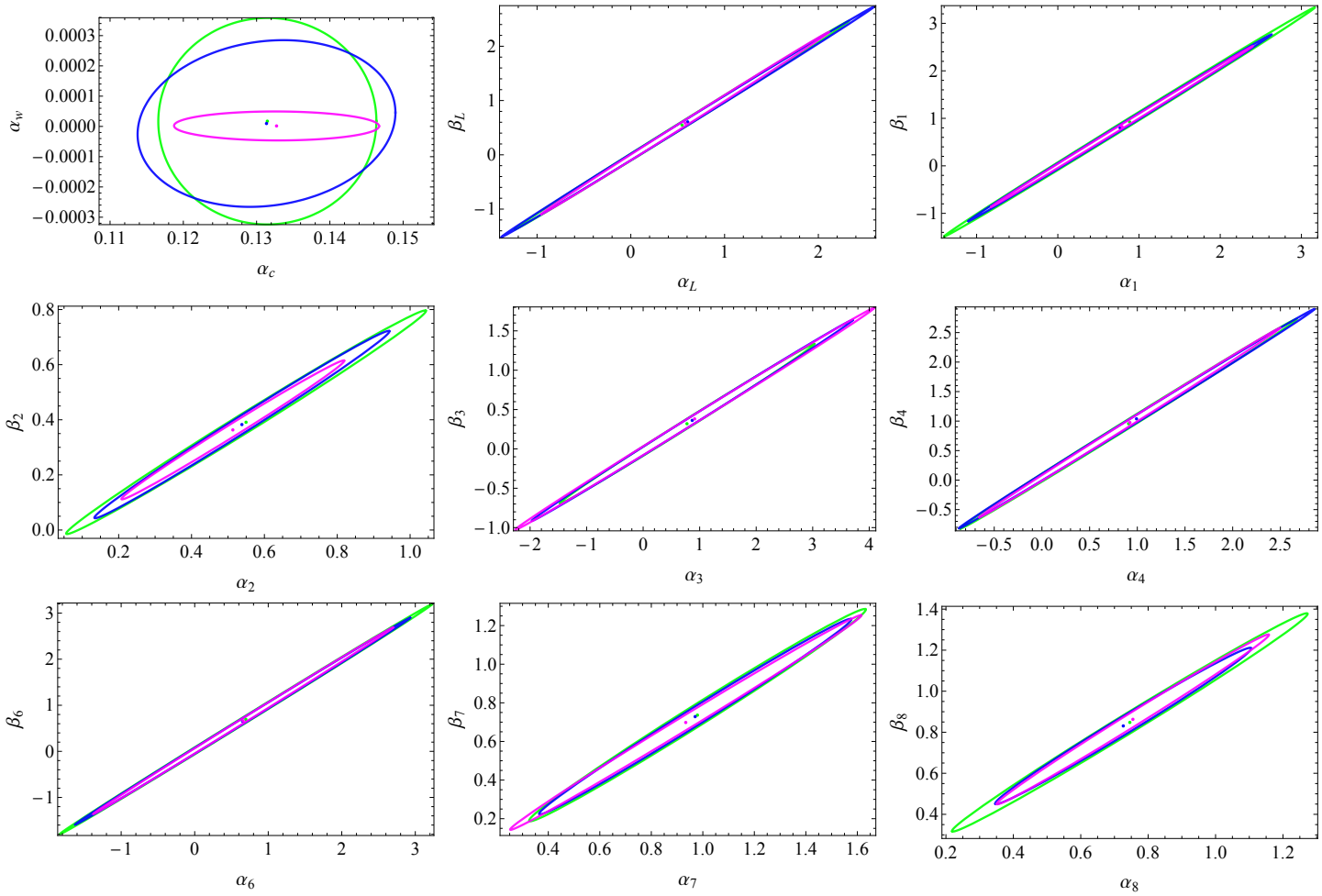

FIG. 5: This figure shows the results of a fit where the model equations do not perfectly match that of the system that was used to generate the data. It should be noted that fits using 5, 6 and 7 points are displayed in green, blue, and magenta respectively. Each ellipse is representative of the 95% confidence region of the fit parameters.

VI. DE-PARAMETERIZATION/SIMPLIFICATION

Often better fits can be achieved with a simpler approximate model with fewer parameters of a simplified system of equations. One of the most successful approximations one can make in order to de-parameterize a system is known as the equilibrium assumption, or an approximation that can be made if one region reaches equilibrium significantly faster than other regions. EQ. 1. is not the perfect candidate for this procedure in terms of de-parameterization, however, the region $N_s(t)$ tends to be constant in time after a very short period. In such a case, it is reasonable

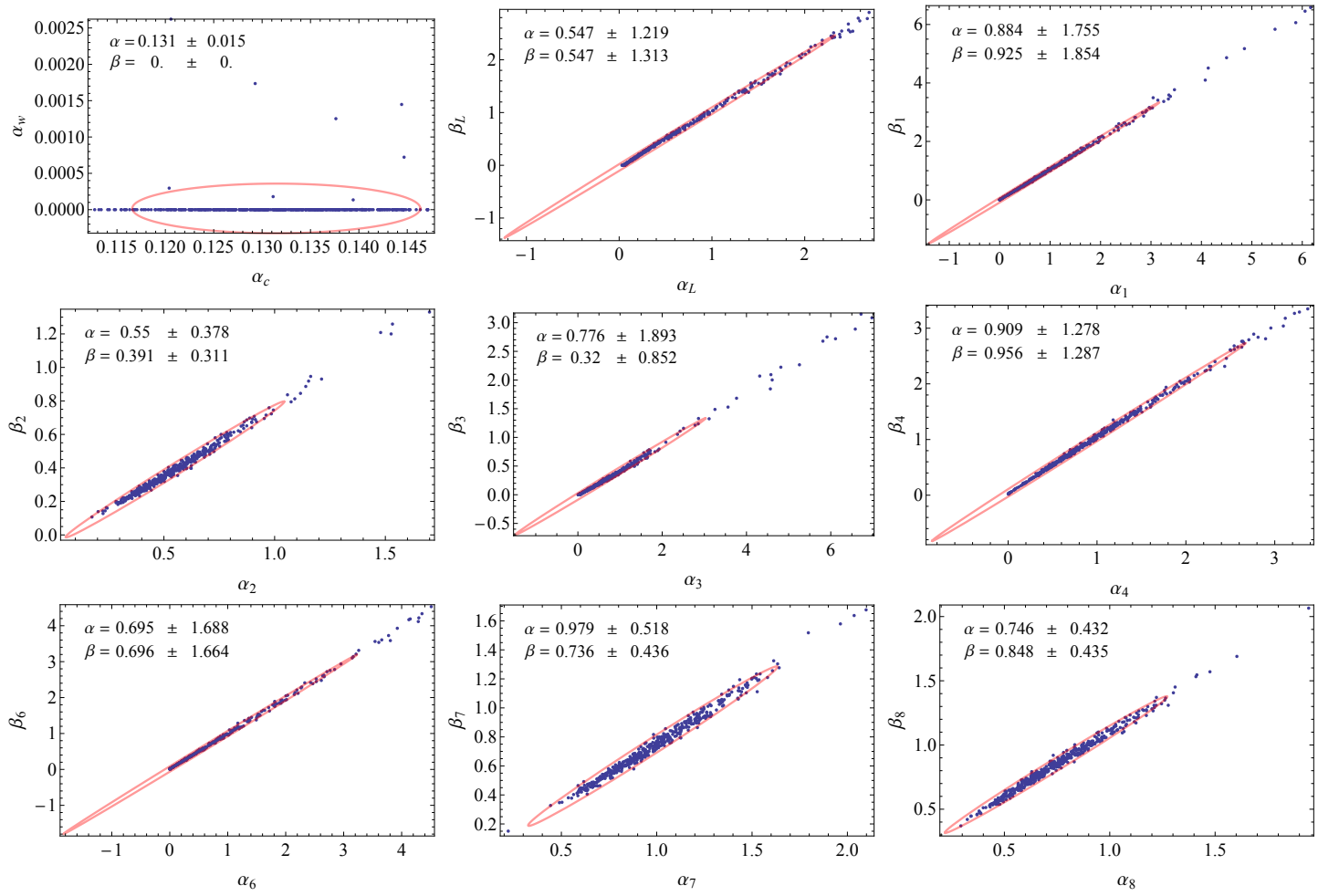

FIG. 6: Here we see the results of the bootstrapping procedure ignoring the existence of the $N_5(t)$ compartment and its associated parameters. Again, the mean of each data set is displayed on the figure, as well as its uncertainty reported to $2s$.

to rewrite EQ. 1. as

$$\frac{d}{dt}\begin{pmatrix} N_c(t) \\ N_L(t) \\ N_w(t) \\ N_1(t) \\ \vdots \\ N_n(t) \\ N_s(t) \end{pmatrix} = \begin{pmatrix} & & & & & & 0 & 0 \\ -\alpha_c & & & & & & 0 & 0 \\ \alpha_c & (-\alpha_L - \alpha_w) & & & & & \beta_L & \\ & \alpha_w & 0 & & & & & \\ & & & -\alpha_1 & & & \beta_1 & \\ & & & & \ddots & & \vdots & \\ & & & & & -\alpha_n & \beta_n & \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & \end{pmatrix} \begin{pmatrix} N_c(t) \\ N_L(t) \\ N_w(t) \\ N_1(t) \\ \vdots \\ N_n(t) \\ N_s(t=0) \end{pmatrix}. \tag{17}$$

While this procedure unfortunately does not remove parameters from the system, it does however greatly simply the system of differential equations that needs to be optimized. Due to time constraints, the bootstrapping procedure of this system was only done using a fit of 5 points on the ideal case in order to best compare the results of the two fits. It was again assumed that

there were no measurements made which were associated with $N_w(t)$

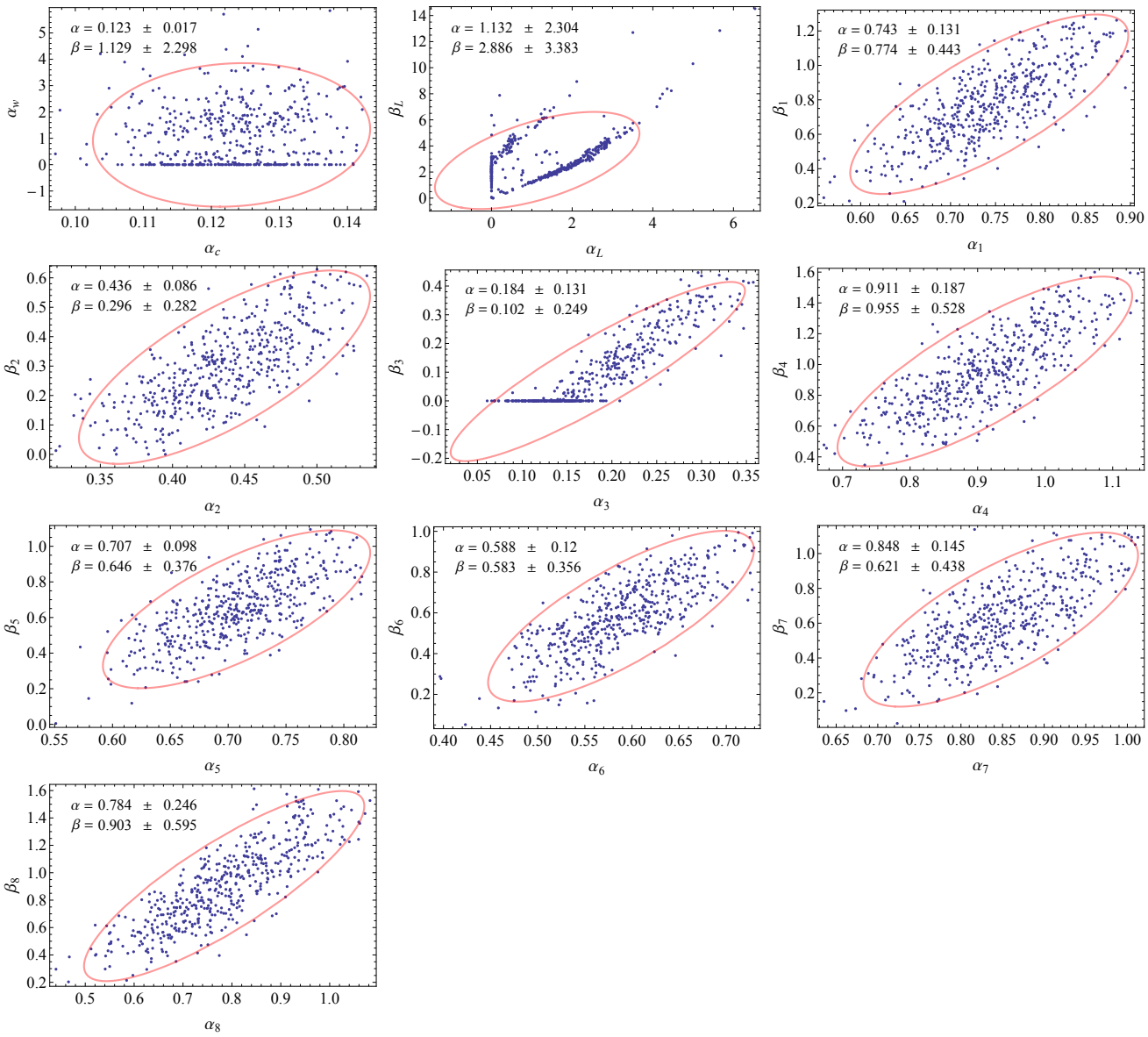

FIG. 7: 5 sampling points of the approximate system

FIG. 7 shows interesting performance in the regions $N_1 - N_8$ in that there is more significant spread between the respective α and β parameters, but the solution obtained for these rate constant converges more accurately than that of the unsimplified model. Interestingly, the approximate model does fail to find correct parameters of regions which are directly affected by the un-sampled function N_w, as can be seen by the bootstrapping procedure of β_L vs α_L and α_w vs. α_c. This is likely due to the fact that our equilibrium assumption nearly uncouples this system of equations from the remaining regions, and as we have included no measurements of N_w, this system no longer has enough information to accurately determine rate parameters. There is also a very clear region of failure in several of the parameters where they are simply optimized to zero, and get stuck regardless of the behavior of their sister rate. However, with the exception of these regions at the very least, this approximate model shows promise for use as an initial guess

at a solution for EQ. 1. due to their strong performance in recovering the rates in seven of ten regions. Unfortunately the de-parameterization procedure was done too late in order to test its performance as an inital guess at solutions of the full system of equations.

VII. DISCUSSION/POTENTIAL EXPERIMENTAL DESIGN

The relative performances of each of the tests involving a fit with 5 data points from each function performed in this document are summarized in Table II including $2s$ standard deviation of each of the data sets obtained via bootstrapping. Using this table one can compare the relative performance of each method based on how well the parameters were recovered. For the most part, each parameter is recovered to accuracy where one can confidently say that it is either positive or negative. The performance of the full system and the system modeled missing a region are very comparable, with means converging to "near" the true solution and relatively large standard deviations which include the actual input value. This is likely a result of the system trying to compensate for the missing region at the expense of the region of which no measurements were made.

Parameter	**Generator Value**	**Full Sys Standard Dev** $(2s)$		**Missing Region Standard Dev** $(2s)$		**Approx Sys Standard Dev** $(2s)$	
α_c	0.1251	0.124	0.013	0.131	0.015	0.123	0.017
α_L	0.2453	0.304	0.345	0.547	1.219	1.132	2.304
α_1	0.7492	1.226	2.026	0.884	1.755	0.745	0.134
α_2	0.4412	0.568	0.335	0.55	0.378	0.437	0.089
α_3	0.1719	1.137	3.005	0.776	1.893	0.187	0.136
α_4	0.9227	1.122	1.464	0.909	1.278	0.911	0.187
α_5	0.7124	0.834	0.433	N/A	N/A	0.709	0.101
α_6	0.5899	1.154	2.086	0.695	1.688	0.591	0.124
α_7	0.8498	0.758	0.526	0.979	0.518	0.851	0.149
α_8	0.7778	0.853	0.441	0.746	0.432	0.786	0.249
α_w	0.0073	0.016	0.024	0.	0.	1.129	2.298
β_L	0.2403	0.294	0.396	0.547	1.313	2.886	3.383
β_1	0.7902	1.087	2.124	0.925	1.854	0.743	0.443
β_2	0.3117	0.401	0.27	0.391	0.311	0.296	0.2.82
β_3	0.0666	0.216	1.438	0.32	0.852	0.102	0.249
β_4	0.9796	1.165	1.478	0.956	1.287	0.995	0.528
β_5	0.6581	0.768	0.442	N/A	N/A	0.646	0.376
β_6	0.5998	0.927	2.04	0.696	1.664	0.583	0.356
β_7	0.6399	0.548	0.442	0.736	0.436	0.621	0.438
β_8	0.8908	0.973	0.446	0.848	0.435	0.903	0.595

TABLE II: This table compares the recovered parameters of the three different methods and displays their respective standard deviations. Please note that the standard deviations are reported to $2s$ and that the mean and standard deviation of each parameter are with respect to the total data set of the bootstrapping, not of the standard error of each parameter for a single fit.

It is interesting however, that the fitting procedure working under the equilibrium assumption for the region N_s performed far superior to the exact method in recovering the rates to reasonable accuracy for the regions bound only to N_s. While this assumption did not de-parameterize the system, it did however significantly simplify the solutions to the differential equation. With this in mind, if only those rates are of interest to actual data, this method provides superior solutions.

At the very least the equilibrium solutions provide an obvious candidate for an initial guess at for parameter optimization for the full model, rather than simply automating the process with a random number generator. All in all however, these results are less than promising as in most cases these rates can be either positive or negative within uncertainty, which is too large to make any strong claims as to the behavior of the system based on the minimal points used for fitting as a difference in sign completely changes the dynamics of the system.

However, with an end goal in mind of actually accounting for the *difference* in the optimized rate parameters, it becomes of interest to compare the performance of each of these methods at recovering the appropriate difference between rate parameters of a given region. From Table

Parameters	**Generator Diff.**	**Full Sys**	**Standard Dev** $(2s)$	**Approx Sys. Standard Dev** $(2s)$		**DeParam Sys. Standard Dev.** $(2s)$	
$\alpha_L - \beta_L$	4.95×10^{-3}	-2.79×10^{-4}	1.04×10^{-1}	-9.79×10^{-3}	5.96×10^{-2}	-1.75	3.84
$\alpha_1 - \beta_1$	-4.10×10^{-2}	-4.98×10^{-2}	1.18×10^{-1}	-4.92×10^{-2}	1.17×10^{-1}	-3.14×10^{-2}	3.39×10^{-2}
$\alpha_2 - \beta_2$	1.29×10^{-1}	1.58×10^{-1}	7.56×10^{-2}	1.60×10^{-1}	7.47×10^{-2}	1.41×10^{-1}	4.60×10^{-1}
$\alpha_3 - \beta_3$	1.05×10^{-1}	2.64×10^{-1}	1.04	2.60×10^{-1}	1.56	8.56×10^{-2}	3.13×10^{-1}
$\alpha_4 - \beta_4$	-5.68×10^{-2}	-5.93×10^{-2}	5.78×10^{-2}	-5.87×10^{-2}	4.45×10^{-2}	-4.45×10^{-2}	3.07×10^{-1}
$\alpha_5 - \beta_5$	5.43×10^{-2}	5.90×10^{-2}	5.79×10^{-2}	N/A	N/A	6.12×10^{-2}	5.41×10^{-1}
$\alpha_6 - \beta_6$	-9.87×10^{-3}	4.15×10^{-3}	9.85×10^{-2}	5.91×10^{-2}	5.05×10^{-2}	-6.07×10^{-2}	3.92×10^{-1}
$\alpha_7 - \beta_7$	2.10×10^{-1}	-1.81×10^{-1}	4.45×10^{-2}	4.17×10^{-3}	2.27×10^{-1}	2.65×10^{-1}	3.86×10^{-1}
$\alpha_8 - \beta_8$	-1.13×10^{-1}	-1.21×10^{-1}	5.45×10^{-2}	-1.02×10^{-1}	4.27×10^{-3}	-1.19×10^{-1}	3.49×10^{-1}

TABLE III: Here we see a comparison of how each method recovers the difference between their respective rate parameter. The standard deviation is representative of the standard deviation of the set created by subracting each α and β data set element by element.

III, the performance of the method in regards to recovering the difference between the rates is surprising given the poor performance of recovering the actual values, but unsurprising given the tight correlation between each parameter in the exact models. While by most standards, the uncertainty associated with these parameters is too large to be useful, to an isotope scientist this range is actually quite promising. The net difference in each rate is in many cases accurate to being able to determine if a compartment is having a net gain or loss of an isotope, which is exactly the information of interest in order to determine if a region is going to be enriched or depleted in a certain isotope. Unlike Table II, Table III shows us that the difference in recovered rates is more accurate and precise with the exact models, including the fit to an incorrect system of equations. This however is not unexpected with reference to FIG 7 as compared to FIGs 4 and 6 noting the width of the ellipse of each respective 95% region, the exact models show a much tighter relationship between each parameter than that of the equilibrium assumption. In these exact models coupled α and β terms are significantly more correlated than in the approximate case, and as such resulting in a much smaller standard deviation in the difference of each parameter.

If the difference between respective rate parameters is the quantity of interest in order to determine the net gain or loss of a certain isotope in a particular region, using a system of rate equations and minimal fitting points shows promise to recover accurate to sign net rates to or

from a specific region. Of course, an experimental set up is not without challenges different than those encountered using the synthetic data of this document. The most prominent issue being that one has to have at least approximate knowledge of the transfer rates between each compartment in order to determine ideal sampling times in attempt to capture snapshots of the system before it reaches equilibrium, as well as at least a single point after the system has reached equilibrium. In real world situations this can be quite difficult as the systems are not necessarily closed as they were here, and often approximate knowledge of transfer rates is unknown.

There is also significant issue in that using synthetic data generated from a known system to fit either the system used to generate it, or a system relatively similar in mathematical form is more than likely going recover accurate parameters. Real data is not as easily controlled and subject to a much greater range in measurement errors as there may be significantly more factors attributing to the evolution of individual compartments, not to mention that any mathematical model used to fit the data is often not an accurate representation of the system from which the data came. Even further, measurement errors may not be Gaussian in nature, which can cause significant issues when it comes to fitting to noisy data. Based on these possible issues, the "minimum" of five data points in each region deemed acceptable here may not be the case with real data. Also assumed with the synthetic data chosen here is that the abundance in each region is of the same order of magnitude and the net change between each region is slow. This may not be the case with real data, and may change the behavior of the fit parameters significantly. Of course, this potential steadily increases should it be possible to obtain a more

VIII. CONCLUSION

In conclusion, using Levenberg-Marquadardt as a minimization method for nonlinear least squares to fit a system of first order differential equations to measured data is possible, and works reasonably well with minimal data sets. It is possible to recover the difference in transfer rates between two regions to enough accuracy to state whether or not a region is having a net gain or loss of a certain isotope. It must be stressed that synthetic data is always a best case scenario and less optimistic tests using synthetic data which could lead to poor conditioning of the Jacobian matrix as well as fits attempted with significantly flawed systems of equations are needed to test parameter recovery under more realistic conditions. If these simulations still prove promising, testing with real data under appropriate experimental controls may potentially be fruitful. Even so, real data has none of the controls that any synthetic test has in that there are no guarantees of a closed system or that the system of equations used to model a real system is an accurate enough representation of the system itself. With all this in mind it seems quite

hopeless for this to be applied to a real system, however with further synthetic testing and an appropriately designed experiment with rigid controls, a method similar to that presented here shows potential. Of course, the odds of success increase dramatically should many high-quality measurements of a system be possible.

IX. REFFERENCES

[1] Te'louk, P. et. al (2014) Copper isotope effect in serum of cancer patients. A pilot study, *Metallomics. Advance Article.* doi: 10.1039/C4MT00269E

[2] Young E. D., Galy A., and Nagahara H. (2002) Kinetic and equilibrium mass-dependent isotope fractionation laws in nature and their geochemical and cosmochemical significance, *Geochim. Cosmochim. Acta* **66**, 6, 1095-1104.

[3] Rawlings J. (2002) *Chemical Reactor Analysis and Design Fundamentals.*, Nob Hill Publishing, US.

[4] Gershenfeld N. (1999) Function Fitting. In *The Nature of Mathematical Modeling* Cambridge University Press, US.

[5] Burden R. and Faires J. D. (1985) Initial-Value Problems for Ordinary Differential Equations. In *Numerical Analysis* PWS Publishers, US.

X. APPENDIX - MATHEMATICA CODES

A. Defining the system of equations

```
alp = Table[
  Symbol["a" <> ToString@i], {i, 1,
   12}]; (*Define our system of equations*)
bet = Table[Symbol["b" <> ToString@i], {i, 1, 12}];
ns = Table[Symbol["n" <> ToString@i], {i, 1, 12}];
diffs = Flatten[
  {Table[-Abs@alp[[i]]*ns[[i]][t], {i, 1, 1}],(*Uncoupled stuff*)
   -(Abs@alp[[11]] + Abs@alp[[2]])*ns[[2]][t] +
    Abs@bet[[2]]*ns[[Length@ns]][t] + Abs@alp[[1]]*ns[[1]][t],
   Table[
    Abs@bet[[i]]*ns[[Length[ns]]][t] - Abs@alp[[i]]*ns[[i]][t], {i, 3,
      10}],(*To serum stuff*)
   Abs@alp[[Length[alp] - 1]]* ns[[2]][t],(*Waste stuff*)
   Sum[Abs@alp[[i]]*ns[[i]][t], {i, 2, 10}] -
    Sum[Abs@bet[[i]], {i, 2, 10}]*
     ns[[Length[ns]]][t]};(*my man the serum*)
derv = Table[ns[[i]]'[t], {i, 1, 12}];
derv2 = Delete[derv, {7}] (*approximate system with D[n12] == 0*)
system = Table[derv[[i]] == diffs[[i]], {i, 1, Length[derv]}];
alp2 = Delete[alp, {7}];
bet2 = Delete[bet, {7}];
ns2 = Delete[ns, {7}];
diffs2 = Flatten[
  {Table[-Abs@alp2[[i]]*ns2[[i]][t], {i, 1, 1}],(*Uncoupled stuff*)
   -(Abs@alp2[[10]] + Abs@alp2[[2]])*ns2[[2]][t] +
    Abs@bet2[[2]]*ns2[[Length@ns2]][t] + Abs@alp2[[1]]*ns2[[1]][t],
   Table[
    Abs@bet2[[i]]*ns2[[Length[ns2]]][t] -
     Abs@alp2[[i]]*ns2[[i]][t], {i, 3, 9}],(*To serum stuff*)
   Abs@alp2[[Length[alp2] - 1]]* ns2[[2]][t],(*Waste stuff*)
   Sum[Abs@alp2[[i]]*ns2[[i]][t], {i, 2, 9}] -
    Sum[Abs@bet2[[i]], {i, 2, 9}]*
     ns2[[Length[ns2]]][t]};(*my man the serum*)
system2 = Table[derv2[[i]] == diffs2[[i]], {i, 1, Length[derv2]}];
```

B. Functions Used to fit data

```
sol = ParametricNDSolveValue[
   Riffle[system, ic],
   Table[ns[[i]], {i, 1, 12}],
   {t, 0, 15},
   Flatten[{
     Table[alp[[i]], {i, 1, 11}],
     Table[bet[[i]], {i, 2, 10}]
     }],
   Method -> "LSODA"]; (*set up equations to optimize*)
```

```
sol2 = ParametricNDSolveValue[
    Riffle[system2, ic2],
    Table[ns2[[i]], {i, 1, 11}],
    {t, 0, 15},
    Flatten[{
      Table[alp2[[i]], {i, 1, 10}],
      Table[bet2[[i]], {i, 2, 9}]
      }],
    Method ->
     "LSODA"]; (*set up equations to optimize with different initial \
guess*)
apsol2 =  ParametricNDSolveValue[
  Riffle[approxsystem2, ic2],
  Table[ns2[[i]], {i, 1, 11}],
  {t, 0, 15},
  params2,
  Method -> "LSODA"]; (*set up approximate equations to optimize*)
```

C. Generation of Synthetic Data with Assumed Rates

```
points = Range[0.001,12,.3];
ordinates=With[{a1=0.12512044723935`,a2=0.2453021,a3=0.7491892366497901`,a4
    =0.44118386223923545`,a5=0.17193008586682357`,a6=0.922720059201386`,a7
    =0.7124062074680186`,a8=0.5898966317547063`,a9 = 0.8498145306514904`, a10
    =0.7778254794939592`,a11=0.007256062876177221,b2=0.240343, b3
    =0.7902230762443223`,b4=0.31166228728883012`,b5=0.06655575840953443`,b6
    =0.9795814487355847`,b7=0.6580558294204786`,b8=0.599771396211936`,b9
    =0.6398638788598377`,b10=0.8907779882884594`}
,Through[sol[a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,b2,b3,b4,b5,b6,b7,b8,b9,b10][
    points],List]];
ordinates = Delete[ordinates,{7}];
data = ordinates+RandomVariate[NormalDistribution[0,.25],Dimensions[ordinates]];
transformedData={ConstantArray[Range@Length[ordinates],Length[points]]//Transpose
    ,ConstantArray[points,Length[ordinates]],data}~Flatten~{{2,3},{1}};
lessData = DeleteCases[transformedData,{Length@ns2 - 1,__}]
```

D. Data Sampling and Fit Solutions

```
DataRandomSampler[regions_,equations_,nump_]:=
(*Grab random elements from the data, sort and partition
it correctly for use in the model fititng*)
Table[
Sort[
RandomSample[
Partition[regions,Length[regions]/equations][[i]],nump],
#1[[2]]<#2[[2]]&],
{i,1,equations}]~Flatten~{1,2}


EqualDataRandomSample[regions_,equations_,nump_]:=
```

```
(*Need to have evenly spaced data, sampled randomly, which apparently is way
    easier to do than I thought before*)
SortBy[
RandomSample[
Transpose[Partition[regions,Length[regions]/equations]],nump]~Flatten~{1,2},
    First]
ChooseData[regions_,equations_,nump_]:=
SortBy[
Transpose[Partition[regions,Length[regions]/equations]][[nump]]~Flatten~{1,2},
    First]

ModelSys[args_][i_,t_]:=Through[(sol@@args)[t],List][[i]]/; (*Use previousl
    defined function to define a model of the system to use for fitting*)
And@@NumericQ/@Flatten[{args,i,t}];

ApModelSys[args_][i_,t_]:=Through[(apsol@@args)[t],List][[i]]/; (*Same, idfferent
     system*)
And@@NumericQ/@Flatten[{args,i,t}];

ModelSys2[args_][i_,t_]:=Through[(sol2@@args)[t],List][[i]]/;
And@@NumericQ/@Flatten[{args,i,t}];

ApModelSys2[args_][i_,t_]:=Through[(apsol2@@args)[t],List][[i]]/;
And@@NumericQ/@Flatten[{args,i,t}];

params = Flatten[{Table[alp[[i]],{i,1,11}],Table[bet[[i]],{i,2,10}]}];
params2 = Delete[params,{{7},{17}}];
```

E. Total BootStrapping procedure

```
listgrabs = {{2, 4, 8, 30}, {2, 4, 6, 8, 30}, {2, 4, 6, 8, 12, 14,
   30}, {2, 4, 6, 8, 12, 14, 16, 30}}
ParamEstimates = Table[{}, {j, 1, Length@listgrabs}]
points = Range[0.001, 12, .3];
ic = Table[ns[[i]][0] == initcond[[i]], {i, 1, Length@initcond}];
sol = ParametricNDSolveValue[
   Riffle[system, ic],
   Table[ns[[i]], {i, 1, 12}],
   {t, 0, 15},
   Flatten[{
     Table[alp[[i]], {i, 1, 11}],
     Table[bet[[i]], {i, 2, 10}]
     }],
   Method -> "LSODA"]; (*set up equations to optimize*)
Do[
 count = 1;
 dump = {};
 While[count < 501,
  other =
   Abs[Delete[initcond, {7}] +
     RandomVariate[NormalDistribution[0, .25],
```

```
      Dimensions[Delete[initcond, {7}]]];

   (*Also add noise to the inital conditions of the parameters*)
  ic2 = Table[ns2[[i]][0] == other[[i]], {i, 1, Length@other}];

  sol2 = ParametricNDSolveValue[
    Riffle[system2, ic2],
    Table[ns2[[i]], {i, 1, 11}],
    {t, 0, 15},
    Flatten[{
      Table[alp2[[i]], {i, 1, 10}],
      Table[bet2[[i]], {i, 2, 9}]
      }],
    Method -> "LSODA"]; (*set up equations to optimize*)

  apsol2 =  ParametricNDSolveValue[
    Riffle[approxsystem2, ic2],
    Table[ns2[[i]], {i, 1, 11}],
    {t, 0, 15},
    params2,
    Method -> "LSODA"]; (*set up equations to optimize*)

  ordinates = With[{a1 = 0.12512044723935`,
     a2 = 0.2453021,
     a3 = 0.7491892366497901`,
     a4 = 0.44118386223923545`,
     a5 = 0.17193008586682357`,
     a6 = 0.922720059201386`,
     a7 = 0.7124062074680186`,
     a8 = 0.5898966317547063`,
     a9 = 0.8498145306514904`,
      a10 = 0.7778254794939592`,
     a11 = 0.007256062876177221,
     b2 = 0.240343,
     b3 = 0.7902230762443223`,
     b4 = 0.31166228728883012`,
     b5 = 0.06655575840953443`,
     b6 = 0.9795814487355847`,
     b7 = 0.6580558294204786`,
     b8 = 0.599771396211936`,
     b9 = 0.6398638788598377`,
     b10 = 0.8907779882884594`}
    , Through[
     sol[a1, a2, a3, a4, a5, a6, a7, a8, a9, a10, a11, b2, b3, b4, b5,
        b6, b7, b8, b9, b10][points], List]
    ];

  ordinates =
   Delete[ordinates, {7}]; (*get rid of data set we're pretending we \
don't know about*)
  data = ordinates +
```

```
    RandomVariate[NormalDistribution[0, .25], Dimensions[ordinates]];
  transformedData = {ConstantArray[Range@Length[ordinates],
       Length[points]] // Transpose,
     ConstantArray[points, Length[ordinates]], data}~
    Flatten~{{2, 3}, {1}};
  lessData = DeleteCases[transformedData, {Length@ns2 - 1, __}];

  fitdata = ChooseData[lessData, Length@ns2 - 1, listgrabs[[k]]];

  apfit2 = NonlinearModelFit[
    fitdata,
    ApModelSys2[params2][i, t],
    params2, {i, t},
    Method -> "LevenbergMarquardt",
    PrecisionGoal -> \[Infinity],
    MaxIterations -> 600
    ];
  guess = Riffle[params2, Table[
      If[
       apfit2["BestFitParameters"][[i, 2]] > -0.1 &&
        apfit2["BestFitParameters"][[i, 2]] < 1.5,
       apfit2["BestFitParameters"][[i, 2]],
       .5]
       , {i, 1, Length@params2}]]~Partition~2;
  fit2 = NonlinearModelFit[
    fitdata,
    ModelSys2[params2][i, t],
    guess, {i, t},
    Method -> "LevenbergMarquardt",
    PrecisionGoal -> \[Infinity],
    MaxIterations -> 600
    ];

  AppendTo[ParamEstimates[[k]],
   Table[fit2["BestFitParameters"][[i, 2]], {i, 1, Length@params2}]];
  count++
  ];
 Print[k];
 , {k, 1, Length@listgrabs}]
```