Parameter Estimation in a Pharmacokinetic Model

Dave Auckly, *Alyson Deines, †and Joelyn Foy ‡

June 12, 2007

We were approached by Ronette Gehring from the Vet Med program with a mathematical modeling problem from pharmacokinetics. She had done a study on stress levels in cattle undergoing castration and wished to estimate parameters in a model from experimental data. She measured cortisol levels in cattle after various treatments. Her basic model was

$$X(t) = A\cos(\omega(t - t_0)) + Be^{-kt} + C,$$

where $X(t)$ represents the cortisol level at time t, A, B, C, k, t_0 are undetermined parameters, and ω is the frequency corresponding to the frequency of one day. Her question was how to estimate these parameters based on the experimental data. We discovered a way to apply linear regression to this problem, even though it is not linear in the parameters. We compared our method with existing non-linear least squares methods. We found that our method produced good results with a small number of measurements and no *a priori* parameter estimates. The standard methods also worked well but required *a priori* parameter estimates. The method that we discovered was actually introduced earlier by Foss [**?**]. Our major finding is that the model Ronette originally proposed does not accurately fit the data. We recommend that she find a better model. This will be future work.

For our algorithm, we used the linear regression algorithm twice. We began with the model Ronette gave us

$$X(t) = A\cos(\omega(t - t_0)) + Be^{-kt} + C.$$

We used an identity to linearize the model $\cos(a + b) = \cos(a)\cos(b) + \sin(a)\sin(b)$ to get

$$X(t) = A\cos(\omega t_0)\cos(\omega t) + A\sin(\omega t_0)\sin(\omega t) + Be^{-kt} + C$$

and finally, rearranging variables

$$X(t) = A\cos(\omega t) + B\sin(\omega t) + Ce^{-kt} + D.$$

Note that $X(t)$ is a solution to $D(D^2 + \omega^2)(D + k)X(t) = 0$ i.e. $D^4X(t) + kD^2X(t) + \omega^2D^2X(t) + k\omega^2X(t) = 0$. This gives the relations on $D^4, k, \omega^2, k\omega^2$. Since normal differentiation is unstable,

*the first author was partially supported by NSF grant DMS-0604994.
†the second auther was partially supported by the K-State I-Center
‡the third auther was partially supported by the K-State I-Center

we can rewrite this in higher integrals:

$$X^{(0)}(t) := X(t)$$

$$X^{(n+1)}(t) := \int_0^t X^{(n)}(s)ds.$$

Then we can arrive at the equation

$$X(t) + kX^{(1)}(t) + \omega^2 X^{(2)}(t) + k\omega^2 X^{(3)}(t) = c_3 t^3 + c_2 t^2 + c_1 t + c_0.$$

As we are assuming a diurnal model, we can take $\omega = \frac{2\pi}{720}$ where 720 represents minutes. We also take our data in the coordinates (t_j, X_j). We can numerically approximate $X^{(n)}(t)$ and use least squares to determine k, c_0, c_1, c_2, c_3. Note that this works best when the times are at regular intervals.

Statement of the Algorithm:

First we use the trapezoidal rule. Input (t_j, X_j), $j = 0, 1, \cdot, n$. For $m = 0, 1, 2$ set $X_0^{(m+1)} := 0$. From $j = 0$ to $n - 1$,

$$X_{j+1}^{(m+1)} := X_j^{(n)} + \frac{1}{2}(X_{j+1}^{(n)} + X_j^{(n)})(t_{j+1} - t_j).$$

Now we use the least squares algorithm found in Matlab. For this we create a matrix W=

$$\begin{pmatrix} w1 & w2 & w3 & w4 & w5 \end{pmatrix}$$

where $w1 = -X^{(1)}(t) - \omega X^{(3)}$, $w2 = t^3$, $w3 = t^2$, $w4 = t$, and $w5 = 1$ are vectors. Next define $y = X(t) + \omega X^{(2)}(t)$. We can then use the subroutine 'regress(y,W)' to estimate the coefficients, namely, to get k. We then input k as given in the equation

$$X(t) = A\cos(\omega t) + B\sin(\omega t) + Ce^{-kt} + D.$$

This allows us to use the least squares method a second time to find all the coefficients A, B, C, D. The new W matrix, W'=

$$\begin{pmatrix} w12 & w22 & w32 & w42 \end{pmatrix}$$

has entries as column vectors as well, where $w12 = \cos(\omega t)$, $w22 = \sin(\omega t)$, $w32 = \exp(-kt)$, and $w42 = 1$. Define the new y, y' to be $y' = X(t)$. Then again use the regress subroutine 'regress(y',W').'

On the same data, we are also using the Matlab subroutine 'fit' from the Curve Fitting toolbox that has the same effect but uses the Non linear least squares method. For this method, we specify the fit type, i.e. the model, and then input the data and the corresponding times.

To compare the methods, we created data sets with error. The data sets were created by inputting set coefficients for A, B, C, D, k into

$$X(t) = A\cos(\omega t) + B\sin(\omega t) + Ce^{-kt} + D$$

with small time intervals. We then added an error term, $E = \sum_1^1 0e(r_i - \frac{1}{2})$ where e is an addition input and r_i represents a random number generated by the computer in the interval [0,1] to give us a uniformly distributed error term. We set the coefficients A, B, C, D, k and changed then changed E or the time intervals (in both intervals and number) and ran the data on both algorithms.

Comparing the tests, we found they both performed well, even with the error input e as high as 90 and as few as 7 time intervals. However, for the Curve Fit test 'fit' to consistently work, we had to input both an initial start and upper and lower bounds on the coefficients. This leads us to think our algorithm is more robust. If in additional test we find the Curve Fit algorithm to be more accurate, we could use our method to find these initial starting points and bounds. Moreover, when given the data collected in Ronette's first experiment, the Curve Fit model failed to produce a model. Our model was able find coefficients, but they did not match the line generated by the data well. This leads us to believe that the model is not the correct model for the data.

References

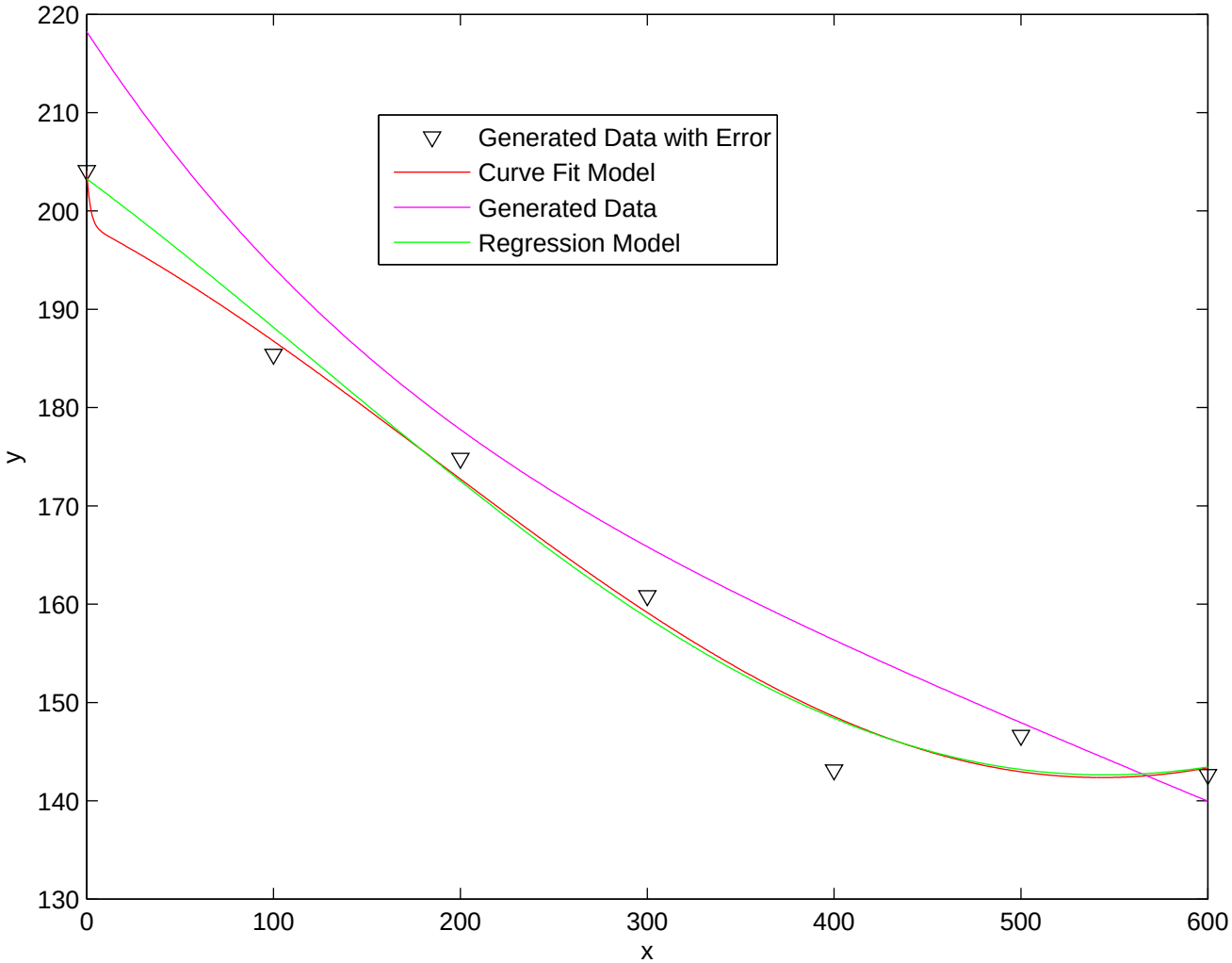
[1] Foss, Stephen D., *A Method of Exponential Curve Fitting by Numerical Integration,* Biometrics, Vol. 26, N0. 4, (Dec) pp. 815-821.

The graphs on pages 1-5 were created from the same initial conditions on A,B,C,D, and k. The error term E changed as did the time scale.

Graphs on Pages 1-3 have the time scale T1=0:100:600, i.e. 0, 100,...,600. What changes is their error term.
On page 1 E=80, on page 2 E=90, and on page 3 E=100.

Graphs on Pages 4 and 5 both have E=100, however they have different time scales.
Page 4 has T2=0,30,60,120,540,570,600 and page 5 has T3=0,30,60,120,180,480,540,570,600.

What these graphs show is that both algorithms are predicting model with decent accuracy,
even with large error terms and different time scales with sparse points.



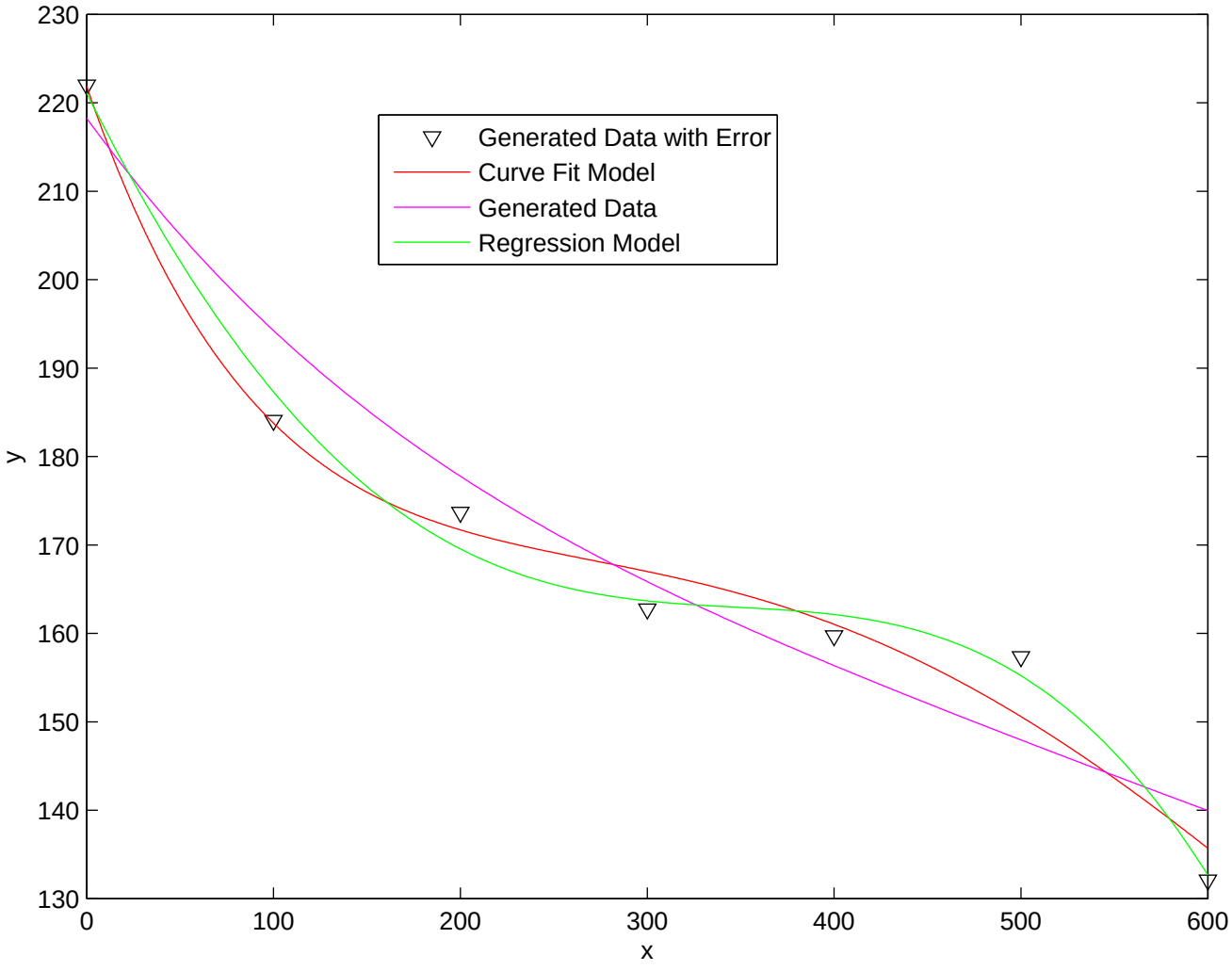
 Note: Initial conditions and boundaries have been implemented in the curve fit algorithm for the graphs on pages 1-5.

The graphs on pages 1-5 were created from the same initial conditions on A,B,C,D, and k. The error term E changed as did the time scale.

Graphs on Pages 1-3 have the time scale T1=0:100:600, i.e. 0, 100,...,600. What changes is their error term.
On page 1 E=80, on page 2 E=90, and on page 3 E=100.

Graphs on Pages 4 and 5 both have E=100, however they have different time scales.
Page 4 has T2=0,30,60,120,540,570,600 and page 5 has T3=0,30,60,120,180,480,540,570,600.

What these graphs show is that both algorithms are predicting model with decent accuracy,
even with large error terms and different time scales with sparse points.



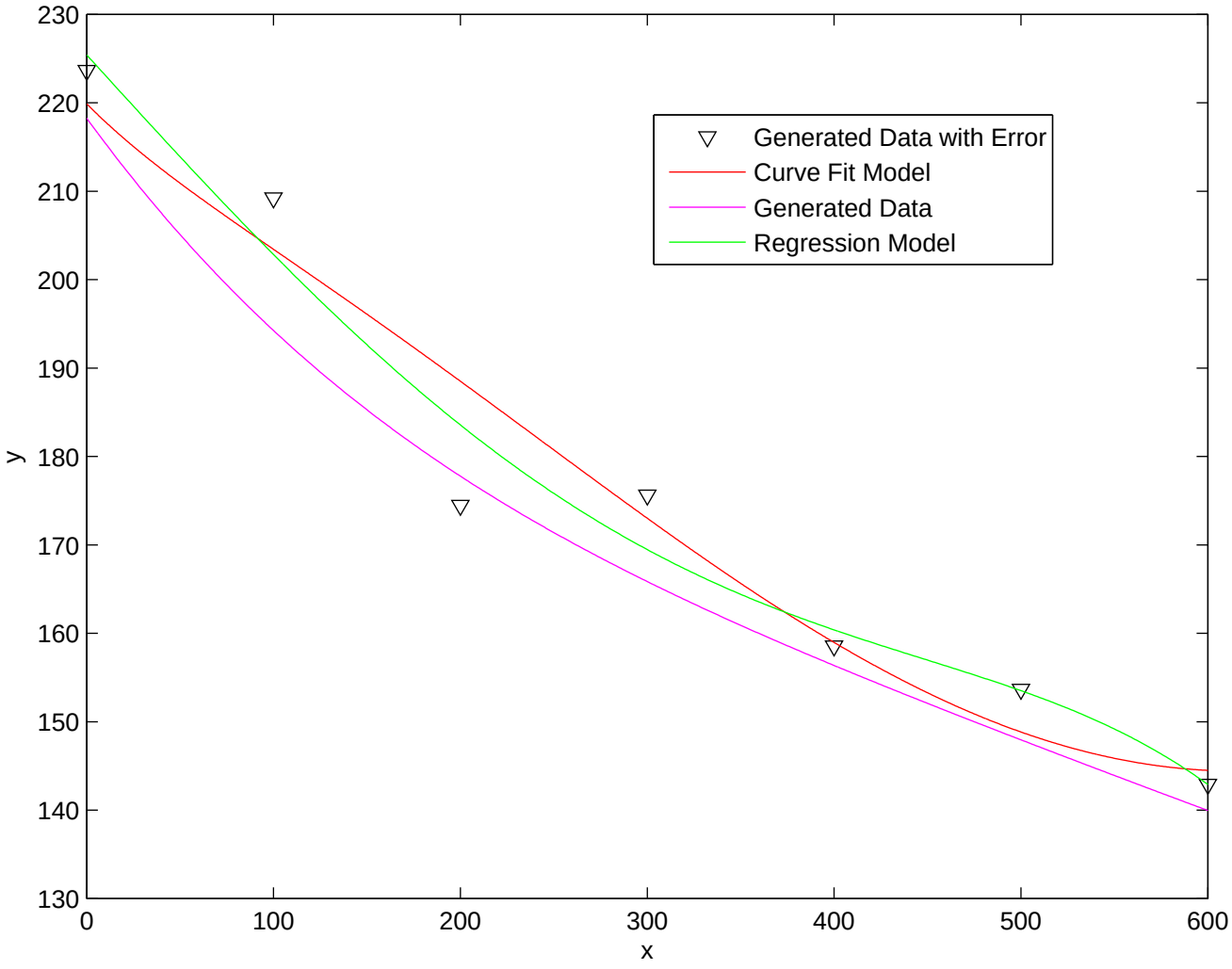
 Note: Initial conditions and boundaries have been implemented in the curve fit algorithm for the graphs on pages 1-5.

The graphs on pages 1-5 were created from the same initial conditions on A,B,C,D, and k. The error term E changed as did the time scale.

Graphs on Pages 1-3 have the time scale T1=0:100:600, i.e. 0, 100,...,600. What changes is their error term.
On page 1 E=80, on page 2 E=90, and on page 3 E=100.

Graphs on Pages 4 and 5 both have E=100, however they have different time scales.
Page 4 has T2=0,30,60,120,540,570,600 and page 5 has T3=0,30,60,120,180,480,540,570,600.

What these graphs show is that both algorithms are predicting model with decent accuracy,
even with large error terms and different time scales with sparse points.



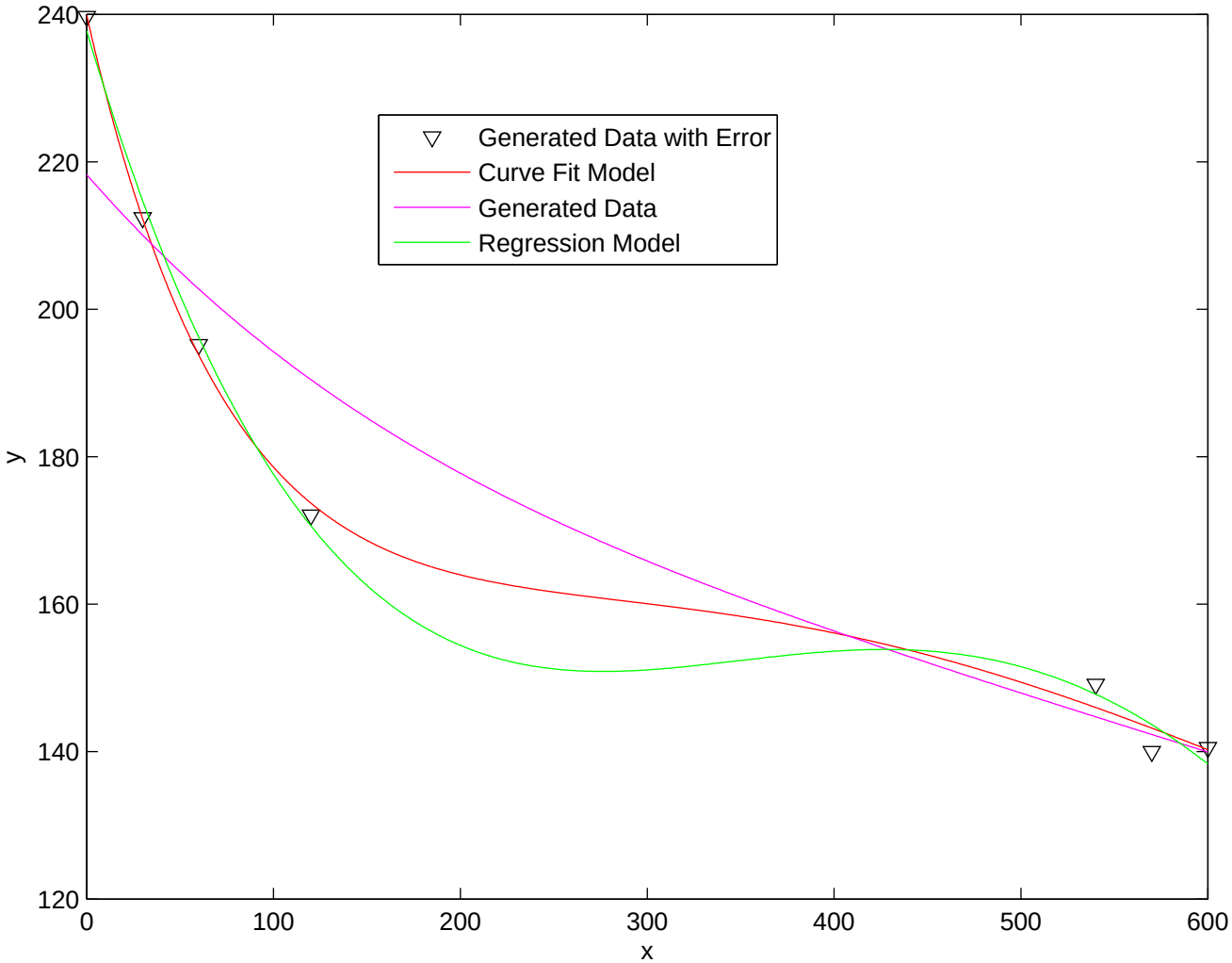
 Note: Initial conditions and boundaries have been implemented in the curve fit algorithm for the graphs on pages 1-5.

The graphs on pages 1-5 were created from the same initial conditions on A,B,C,D, and k. The error term E changed as did the time scale.

Graphs on Pages 1-3 have the time scale T1=0:100:600, i.e. 0, 100,...,600. What changes is their error term.
On page 1 E=80, on page 2 E=90, and on page 3 E=100.

Graphs on Pages 4 and 5 both have E=100, however they have different time scales.
Page 4 has T2=0,30,60,120,540,570,600 and page 5 has T3=0,30,60,120,180,480,540,570,600.

What these graphs show is that both algorithms are predicting model with decent accuracy,
even with large error terms and different time scales with sparse points.



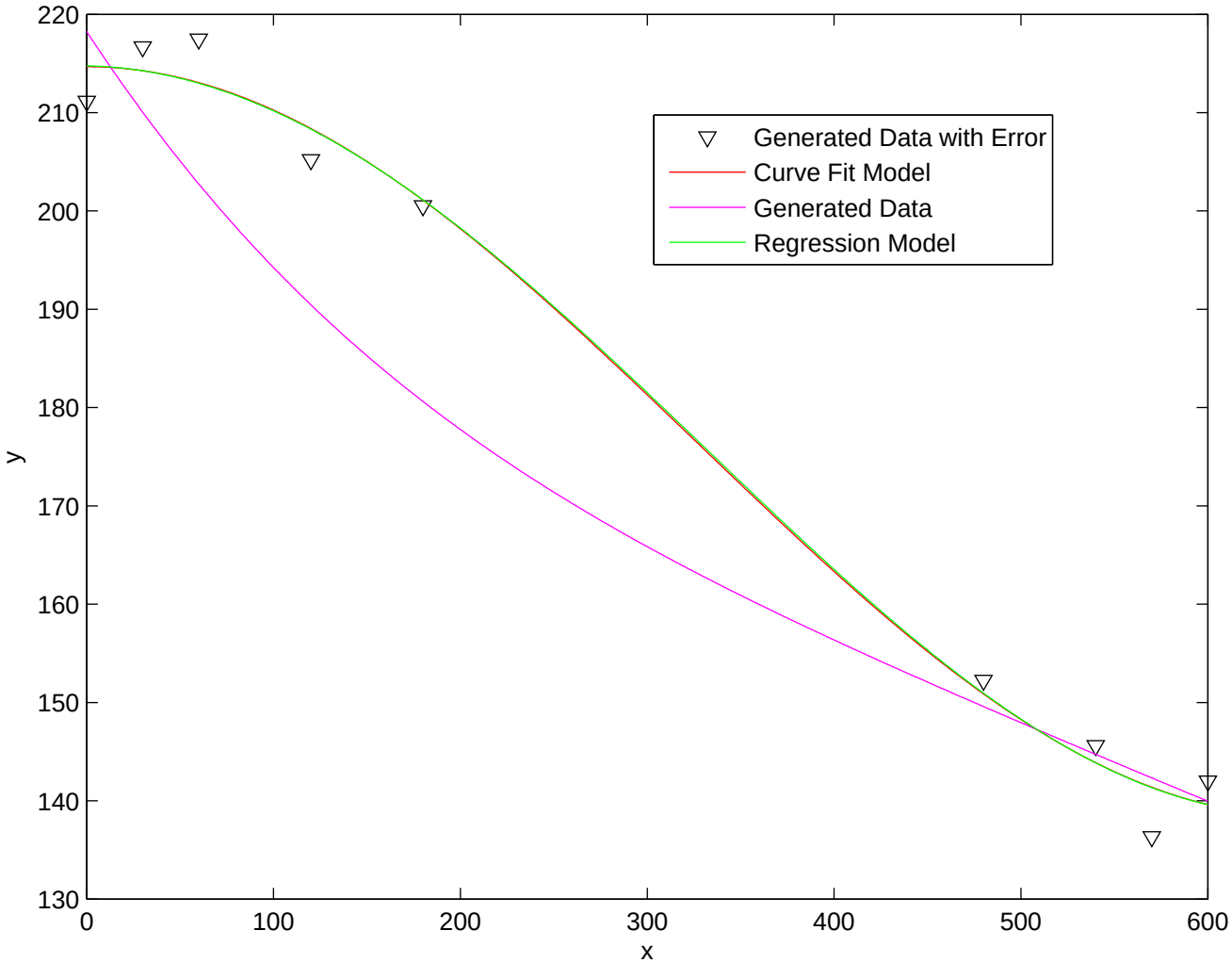
 Note: Initial conditions and boundaries have been implemented in the curve fit algorithm for the graphs on pages 1-5.

The graphs on pages 1-5 were created from the same initial conditions on A,B,C,D, and k. The error term E changed as did the time scale.

Graphs on Pages 1-3 have the time scale T1=0:100:600, i.e. 0, 100,...,600. What changes is their error term.
On page 1 E=80, on page 2 E=90, and on page 3 E=100.

Graphs on Pages 4 and 5 both have E=100, however they have different time scales.
Page 4 has T2=0,30,60,120,540,570,600 and page 5 has T3=0,30,60,120,180,480,540,570,600.

What these graphs show is that both algorithms are predicting model with decent accuracy,
even with large error terms and different time scales with sparse points.



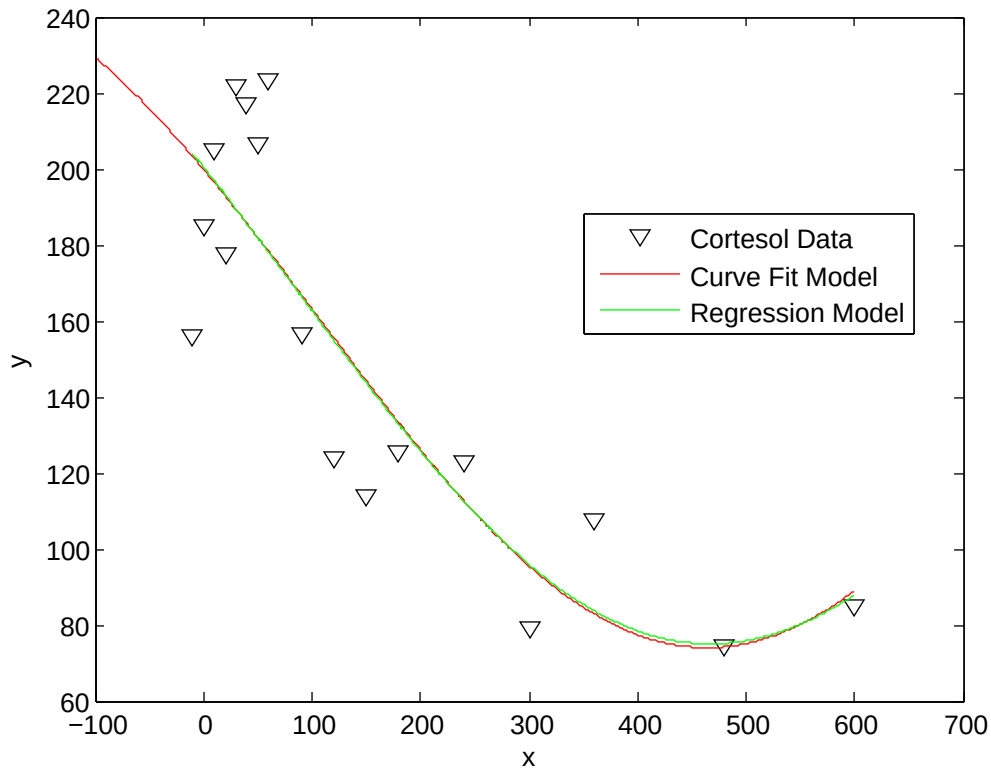
 Note: Initial conditions and boundaries have been implemented in the curve fit algorithm for the graphs on pages 1-5.

The graphs on pages 6 and 7 are of the algorithms run on the Cortesol Data from the study.

On page 6, boundary conditions and initial conditions are imposed on the curve fit algorithm.

On page 7, no boundary or initial conditions are imposed.

These two graphs illustrate two ideas.
First, that the model is not correct, as it is not easily fit.
Second, that even if our regression algorithm does not always fit the data as well as the curve fit algorithm, it is useful because it is more robust and can be used to find a set of initial conditions and boundaries.



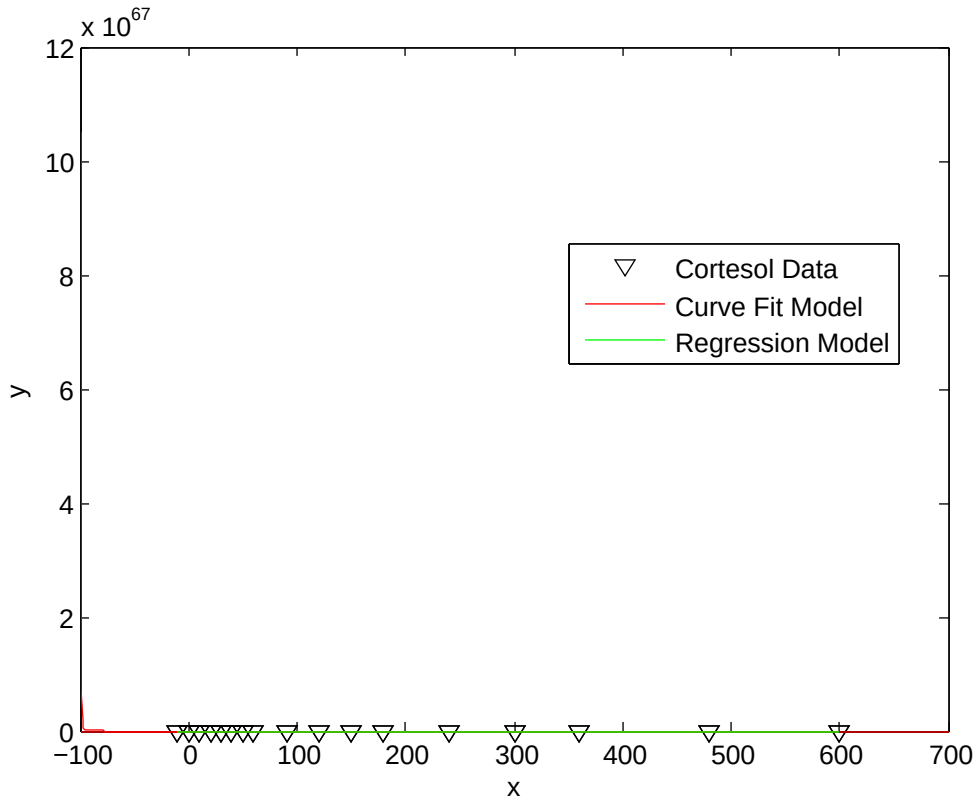
 Note: Initial conditions and boundaries have been implemented in the curve fit algorithm for the graphs on pages 1-5.

The graphs on pages 6 and 7 are of the algorithms run on the Cortesol Data from the study.

On page 6, boundary conditions and initial conditions are imposed on the curve fit algorithm.

On page 7, no boundary or initial conditions are imposed.

These two graphs illustrate two ideas.
First, that the model is not correct, as it is not easily fit.
Second, that even if our regression algorithm does not always fit the data as well as the curve fit algorithm, it is useful because it is more robust and can be used to find a set of initial conditions and boundaries.



 Note: Initial conditions and boundaries have been implemented in the curve fit algorithm for the graphs on pages 1-5.