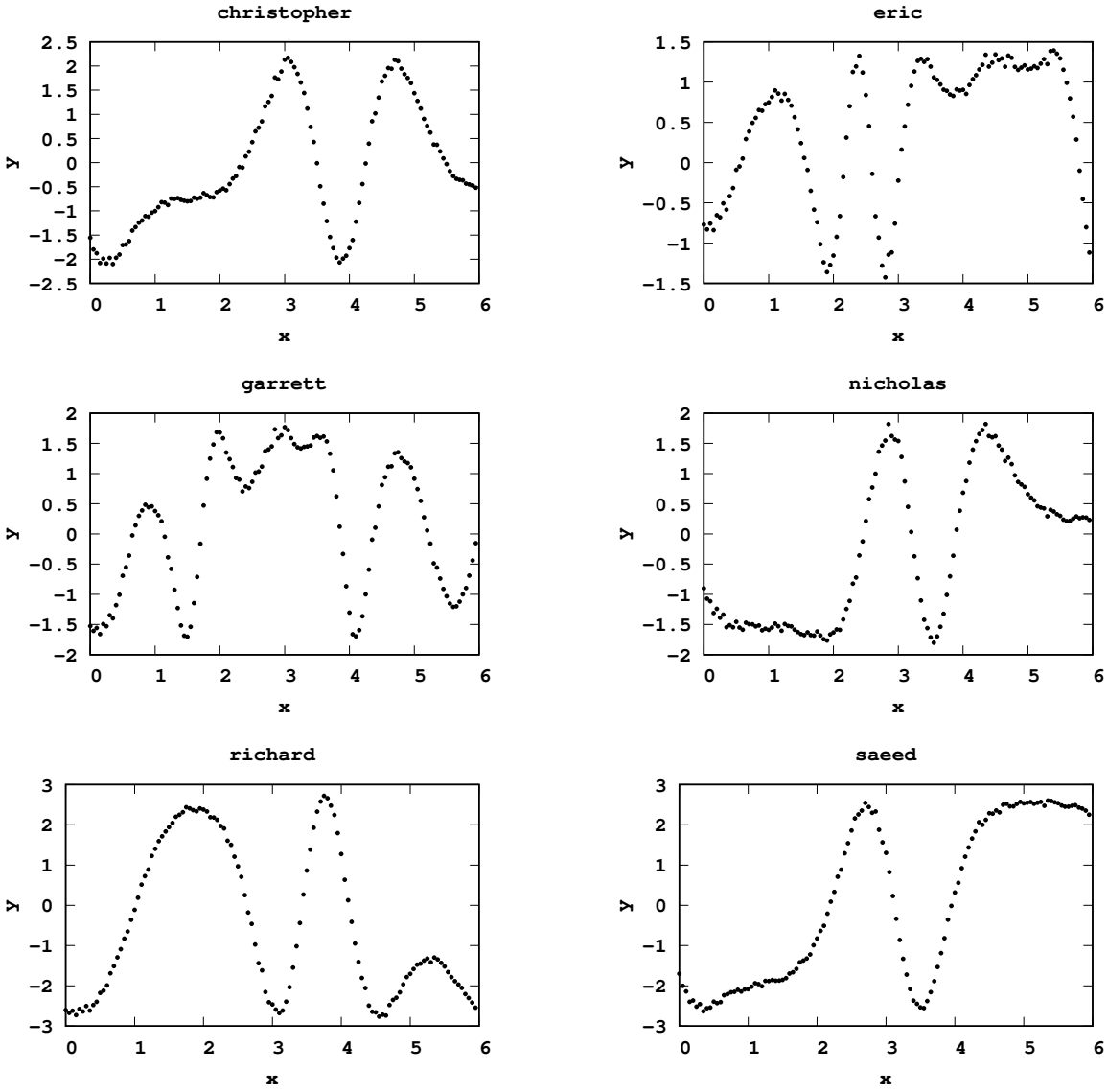
Non-linear Least Squares

1. Plot the points in the data file. Describe the qualitative behavior of the frequency modulated wave and try to guess the frequencies c_3 and c_6.



Recall that the model is given by

$$f(c,t) = c_1 \sin(c_2 + c_4 \sin(c_5 + c_6 t) + c_3 t).$$

Guessing the values of c_3 and c_6 just from looking at the graphs is difficult. The effective frequency represented by c_6 is further multiplied by c_4 in the non-linear model. Moreover, the values for any of the constants may not be related by any simple ratios. Qualitatively, all the waveforms have large scale features and smaller scale features. Although no assumptions have been made about whether $c_3 < c_4 c_6$ or $c_3 > c_4 c_6$ it is reasonable to assume $c_4 c_6$ is larger and therefore corresponds to the small scales.

To make our guesses for c_3 and c_6 we first look at each graph and manually determine the wavelength τ_{small} of the small scales by doubling the width of the smallest discernible peak (or valley). The value of c_{small} corresponding to that wavelength is is then given by the equation

$$\tau_{\text{small}} \cdot c_{\text{small}} = 2\pi.$$

A similar procedure may be used to obtain τ_{large} and c_{large}. Now assume $c_4 c_6 = c_{\text{small}}$ and $c_6 = c_{\text{large}}$. In order to obtain an estimate for c_6 we assume $c_4 = c_6$ so that $c_6 = \sqrt{c_{\text{small}}}$. This results in the following table of values:

	τ_{small}	c_{small}	c_6	τ_{large}	c_3
christopher	1.50	4.19	2.05	6.20	1.01
eric	0.88	7.14	2.67	4.50	1.40
garrett	1.12	5.61	2.37	4.28	1.47
nicholas	1.46	4.30	2.07	4.90	1.28
richard	1.36	4.62	2.15	3.24	1.94
saeed	1.88	3.34	1.83	3.15	1.99

It should be noted there are many other ways to guess the values for c_3 and c_6 and no claim is made that the one discussed above is the best or even reasonable. Therefore, other ways of obtaining values for c_3 and c_6 are not only interesting but will receive full credit provided they are based on a correct interpretation and understanding of the problem.

2. Make other guesses for c_1, c_2, c_4 and c_5 and perform the Gauss–Newton non-linear optimization algorithm. Does it converge? If so, to what?

As stated already we shall assume $c_4 = c_6$. The value of c_1 represents the amplitude of resulting wave which we find by examining the graphs and taking the maximum of the absolute values of the peaks and valleys. The remaining values c_2 and c_5 represent phases.

It may be possible to guess the phase c_2 by finding the first intersection of the waveform with the x axis. To do this, the term involving c_4, c_5 and c_6 is needed which means a value for c_5 is needed first. Lacking intuition how to estimate c_5 we will take $c_5 = 2$ which is the midpoint of the interval $[1, 3]$ of allowable choices. Since this may lead to significant errors in the exact position of the intercept with the x axis, we estimate c_2 by similarly setting $c_2 = 2$. This results in the guesses

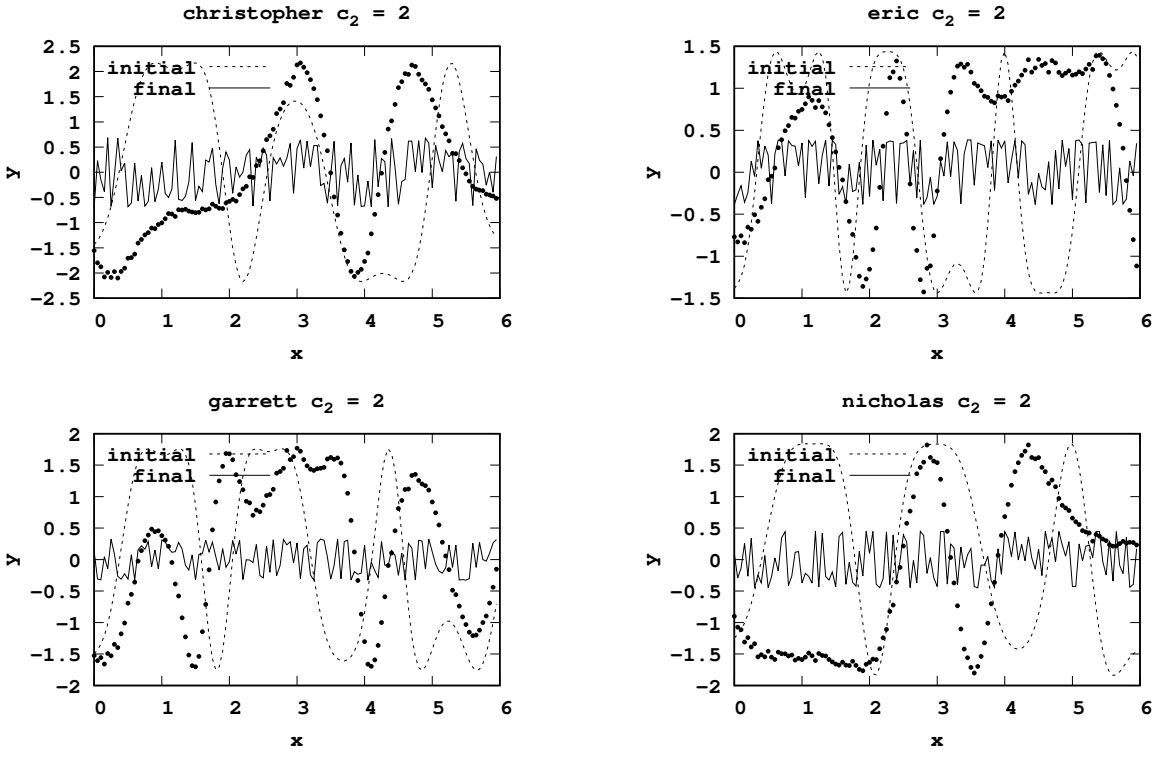
	c_1	c_2	c_3	c_4	c_5	c_6
christopher	2.17	2.00	1.01	2.05	2.00	2.05
eric	1.44	2.00	1.40	2.67	2.00	2.67
garrett	1.76	2.00	1.47	2.37	2.00	2.37
nicholas	1.84	2.00	1.28	2.07	2.00	2.07
richard	2.80	2.00	1.94	2.15	2.00	2.15
saeed	2.63	2.00	1.99	1.83	2.00	1.83

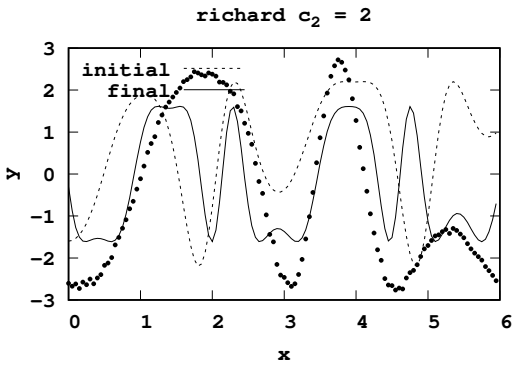
While the way of determining c_1 seems quite natural, as already mentioned, there may be significant variation in the guesses for the other parameters. We now perform the Gauss–Newton non-linear optimization algorithm with initial value for c given by the above guesses

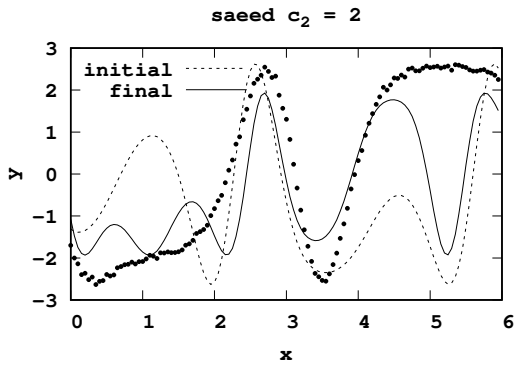
using each of the respective data sets and obtain

		c_1	c_2	c_3	c_4	c_5	c_6
christopher	guess	2.17	2.00	1.01	2.05	2.00	2.05
	final	0.69	−71.40	12.82	32.20	−179.50	51.48
eric	guess	1.44	2.00	1.40	2.67	2.00	2.67
	final	0.38	598.99	−237.81	36.74	−65.32	20.20
garrett	guess	1.76	2.00	1.47	2.37	2.00	2.37
	final	0.33	121.36	−18.53	−155.27	43.69	−3.96
nicholas	guess	1.84	2.00	1.28	2.07	2.00	2.07
	final	−0.45	3.71	−0.64	150.21	6.81	−2.92
richard	guess	2.80	2.00	1.94	2.15	2.00	2.15
	final	1.61	0.79	2.64	3.33	0.87	2.51
saeed	guess	2.63	2.00	1.99	1.83	2.00	1.83
	final	1.93	2.59	1.71	2.12	0.57	2.24

It should be noted that in all cases the final choice of parameters obtained by the Gauss–Newton non-linear optimization did not satisfy the requirement $c_j \in [1, 3]$ for $j = 1, \ldots, 6$. Therefore, even though the algorithm converged, it did not converge to a permissible choice of parameters. The initial and final fit are depicted in the following graphs:





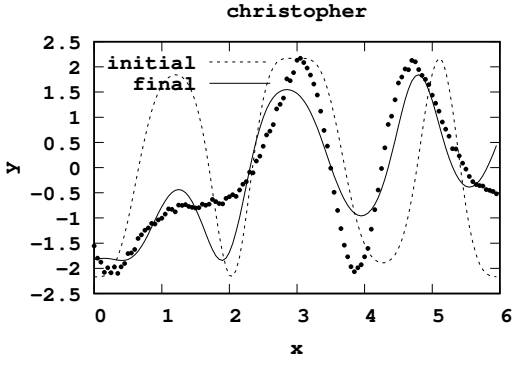


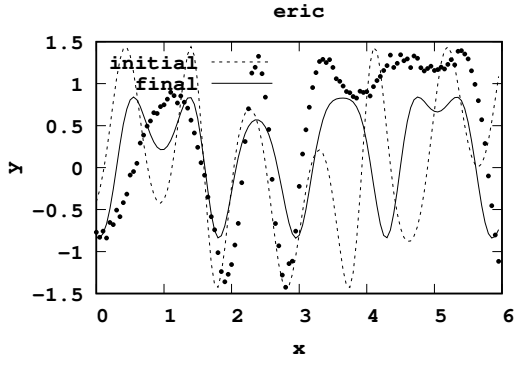
Not only was the fit visually poor in all cases, but in most cases the final fit exhibited oscillations which indicate that the numerical algorithm produced a nonsensical answer.

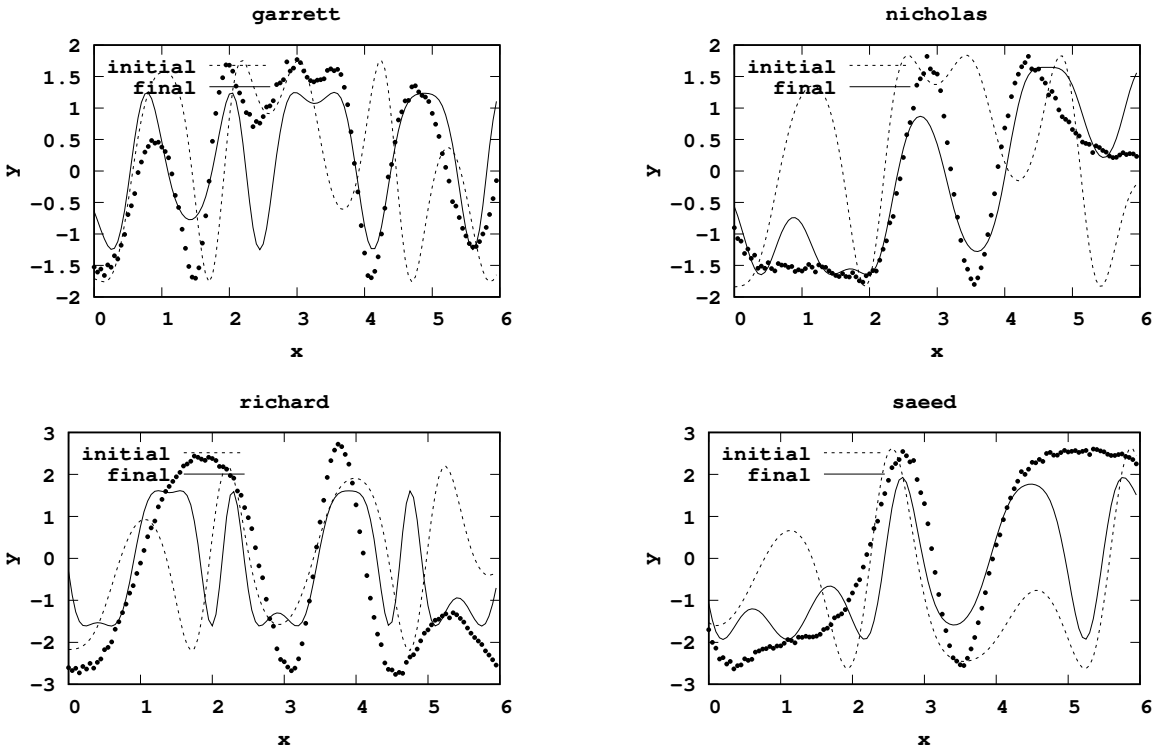
Before moving on to the next question, a different guess for the phase c_2 was tried in order to obtain a hopefully better fit. In particular, c_2 was varied throughout the interval $[1, 3]$ while keeping the other parameters fixed until a solution that did not have oscillations was obtained. The following table indicates the initial guess for c along with the final result obtained by the Gauss–Newton non-linear optimization algorithm.

		c_1	c_2	c_3	c_4	c_5	c_6
christopher	guess	2.17	2.90	1.01	2.05	2.00	2.05
	final	1.84	3.54	0.87	1.31	1.51	2.33
eric	guess	1.44	1.00	1.40	2.67	2.00	2.67
	final	0.84	2.71	0.56	3.01	2.35	2.35
garrett	guess	1.76	2.80	1.47	2.37	2.00	2.37
	final	−1.25	5.39	0.69	3.92	8.32	1.81
nicholas	guess	1.84	2.80	1.28	2.07	2.00	2.07
	final	1.65	−1.89	0.54	−0.96	−16.91	3.35
richard	guess	2.80	2.60	1.94	2.15	2.00	2.15
	final	1.61	0.79	2.64	3.33	0.87	2.51
saeed	guess	2.63	2.10	1.99	1.83	2.00	1.83
	final	1.93	2.59	1.71	2.12	0.57	2.24

Unfortunately, again in all cases the final choice of parameters obtained by the Gauss–Newton non-linear optimization did not satisfy the requirement $c_j \in [1, 3]$ for $j = 1, \ldots, 6$. For reference, the initial and final fit are depicted in the following graphs:







Even though the algorithm converged and the final curves look more plausible, a permissible choice of parameters was not found in any of the cases. It appears the nature of the non-linear model makes difficult to find a suitable initial guess for which for which the Gauss–Newton algorithm converges to c such that $c_j \in [1,3]$ for $j = 1, \ldots, 6$.

3. Write a program that randomly chooses $c \in \mathbf{R}^6$ such that $c_j \in [1,3]$ for $j = 1, \ldots, 6$ and for each random choice of c perform the Gauss–Newton optimization algorithm. You may use any programming language you prefer. Please include a listing of the code in your report.

The code is contained in multiple files as follows:

```
% eric_s3.m -- This is a script that reads the data that will be
%               fit using the Gauss-Newton non-linear optimization
%               algorithm and then chooses random points in the
%               parameter space until 10 guesses are found which
%               converge to a point in the parameter space.
clear all
load("eric.dat");
x=eric(:,1)';
y=eric(:,2)';
k=0;
fprintf(stdout,"%-10s %5s %7s %7s %7s %7s %7s %7s\n",...
    "eric","","c1","c2","c3","c4","c5","c6");
while k<10
```

```
    c0=rand(6,1)*2+1;
    cn=gaussnewton(x,y,c0);
    if and(max(cn)<=3,min(cn)>=1)
        k=k+1;
        fprintf(stdout,...
            "    k=%-4d guess %7.4f %7.4f %7.4f %7.4f %7.4f %7.4f\n",...
            k,c0(1),c0(2),c0(3),c0(4),c0(5),c0(6));
        fprintf(stdout,...
            "%10s final %7.4f %7.4f %7.4f %7.4f %7.4f %7.4f\n",...
            "",cn(1),cn(2),cn(3),cn(4),cn(5),cn(6));
    end
end
```

```
% gaussnewton.m -- Perform Gauss-Newton non-linear optimization
%                  algorithm until the resulting choice of
%                  parameters has converged to within 1e-8 or
%                  until 2000 iterations, whichever occurs first.
%
function cn=gaussnewton(x,y,c)
    cn=c;
    for k=1:2000
        A=makeA(cn,x);
        B=makeB(cn,x,y);
        xi=A\B;
        cn=cn+xi;
        if norm(xi)<1e-8
            return
        end
    end
end
```

```
% model.m -- This routine gives the non-linear model that will be
%            solved using the Gauss-Newton algorithm.  If you
%            change this function you will also have to update
%            the file makeA.m with the gradient of this function.
%
function y=model(c,t)
    y=c(1)*sin(c(2)+c(4)*sin(c(5)+c(6)*t)+c(3)*t);
end
```

```
% makeA.m -- This routine makes the left side of the linearized
%            problem A xi = B equivalently the gradient of the
%            function appearing in model.m for use with the Gauss-
%            Newton non-linear optimization algorithm.
%
function A=makeA(c,t)
```

```
 A=[
  sin(c(2)+c(4)*sin(c(6)*t+c(5))+c(3)*t);
  c(1)*cos(c(2)+c(4)*sin(c(6)*t+c(5))+c(3)*t);
  c(1)*t.*cos(c(2)+c(4)*sin(c(6)*t+c(5))+c(3)*t);
  c(1)*sin(c(6)*t+c(5)).*cos(c(2)+c(4)*sin(c(6)*t+c(5))+c(3)*t);
  c(1)*c(4)*cos(c(6)*t+c(5)).*cos(c(2)+c(4)*sin(c(6)*t+c(5))+c(3)*t);
  c(1)*c(4)*t.*cos(c(6)*t+c(5)).*cos(c(2)+c(4)*sin(c(6)*t+c(5))+c(3)*t)]';
end
```

```
% makeB.m -- This routine makes the right side of the linearized
%            problem A xi = B for use with the Gauss-Newton
%            non-linear optimization algorithm.
%
function B=makeB(c,x,y)
    B=(y-model(c,x))';
end
```

4. Run the code as many times as needed to find ten different guesses for c that converge to values such that $c_j \in [1, 3]$ for all j. Print those initial guesses along with the resulting vectors they converge to.

Running the code from the previous question yields

christopher		c1	c2	c3	c4	c5	c6
k=1	guess	1.2163	1.4940	2.3757	2.6582	1.1911	1.1801
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=2	guess	1.1437	1.5188	2.6492	2.4594	1.1315	1.1325
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=3	guess	1.1422	2.0299	2.2086	1.4551	1.6279	1.1211
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=4	guess	1.0966	1.4418	2.5975	1.1078	1.2377	1.2419
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=5	guess	2.4159	1.8305	2.7921	2.0827	1.6635	1.0198
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=6	guess	1.9545	2.2615	2.3692	2.0985	1.6295	1.0944
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=7	guess	2.0896	1.7289	2.2268	2.2724	1.1819	1.2317
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=8	guess	2.8389	1.6528	2.4573	1.4910	1.0821	1.2821
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=9	guess	2.2768	2.8582	2.0240	1.2001	1.8632	1.4177
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419
k=10	guess	2.2579	1.5335	2.8582	1.6833	1.2990	1.0411
	final	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419

eric		c1	c2	c3	c4	c5	c6
k=1	guess	1.2860	1.4769	2.6099	2.3018	2.7623	1.5743

```
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=2      guess  2.0963  1.1610  2.9882  2.1131  2.8606  1.4852
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=3      guess  1.3525  2.1861  2.6134  2.2000  1.3979  1.8330
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=4      guess  1.5030  2.4380  2.4667  2.9282  2.6260  1.5940
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=5      guess  1.5842  1.3056  2.7010  2.4086  1.2541  1.9844
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=6      guess  2.6279  2.9555  2.4021  1.6153  2.5506  1.6009
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=7      guess  1.7171  1.6442  2.4533  1.6977  2.6400  1.6571
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=8      guess  2.5009  1.4309  2.9256  2.3538  2.2253  1.6516
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=9      guess  2.3067  1.8489  2.4762  2.7261  2.2482  1.7322
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136
    k=10     guess  2.7607  2.1799  2.4437  2.3651  2.7818  1.5548
             final  1.3187  1.5658  2.8692  2.9502  2.2516  1.6136

garrett                c1      c2      c3      c4      c5      c6
    k=1      guess  2.7408  1.4225  2.6933  2.1786  2.5899  2.2811
             final  1.6709  2.7403  1.9068  1.7971  1.8580  2.6697
    k=2      guess  2.4585  2.0208  1.9296  2.2873  1.1105  2.7663
             final  1.6709  2.7403  1.9068  1.7971  1.8580  2.6697
    k=3      guess  2.7711  1.3874  2.4468  1.9637  1.4736  2.4444
             final  1.5796  2.1945  2.3015  1.5804  2.1785  2.3531
    k=4      guess  2.7100  2.3483  2.2325  1.4283  1.6848  2.3353
             final  1.5796  2.1945  2.3015  1.5804  2.1785  2.3531
    k=5      guess  2.5590  2.3155  2.4127  1.0455  2.0317  2.3302
             final  1.5796  2.1945  2.3015  1.5804  2.1785  2.3531
    k=6      guess  1.0027  2.5500  1.8743  1.8269  2.3276  2.7936
             final  1.6709  2.7403  1.9068  1.7971  1.8580  2.6697
    k=7      guess  2.8658  1.4065  2.5734  2.5102  2.5287  2.3595
             final  1.5796  2.1945  2.3015  1.5804  2.1785  2.3531
    k=8      guess  2.8088  1.2503  2.6739  1.6652  1.3200  2.6331
             final  1.5796  2.1945  2.3015  1.5804  2.1785  2.3531
    k=9      guess  2.7564  2.1591  2.3107  1.7145  2.5884  2.4321
             final  1.6709  2.7403  1.9068  1.7971  1.8580  2.6697
    k=10     guess  1.2680  1.9664  2.4847  2.7539  1.8748  2.4494
             final  1.5796  2.1945  2.3015  1.5804  2.1785  2.3531

nicholas               c1      c2      c3      c4      c5      c6
    k=1      guess  2.9434  1.7273  2.2503  1.2387  1.4696  1.5443
             final  1.6761  2.9231  1.9761  2.0277  1.5499  1.5351
    k=2      guess  2.0236  2.1062  2.3642  1.0851  2.5517  1.4905
```

```
             final  1.6761  2.9231  1.9761  2.0277  1.5499  1.5351
    k=3      guess  1.8570  1.6206  2.4152  1.7466  1.4750  1.3698
             final  1.7293  2.0249  2.3785  1.7990  1.9080  1.3226
    k=4      guess  2.4623  1.1664  2.5317  1.8385  1.4349  1.1951
             final  1.6761  2.9231  1.9761  2.0277  1.5499  1.5351
    k=5      guess  1.6486  1.3422  2.7637  1.2118  1.8782  1.2289
             final  1.7293  2.0249  2.3785  1.7990  1.9080  1.3226
    k=6      guess  1.8340  2.1573  1.8873  2.4773  1.9880  1.3600
             final  1.6761  2.9231  1.9761  2.0277  1.5499  1.5351
    k=7      guess  2.1972  2.1510  2.5479  1.0686  1.0090  2.1249
             final  1.7293  2.0249  2.3785  1.7990  1.9080  1.3226
    k=8      guess  1.3704  2.4717  2.3605  1.3665  1.0426  1.6358
             final  1.7293  2.0249  2.3785  1.7990  1.9080  1.3226
    k=9      guess  1.7217  1.2339  2.3073  1.9281  1.5379  1.5805
             final  1.6761  2.9231  1.9761  2.0277  1.5499  1.5351
    k=10     guess  1.1590  2.6537  2.2532  1.0269  2.8425  1.3016
             final  1.6761  2.9231  1.9761  2.0277  1.5499  1.5351

richard                c1      c2      c3      c4      c5      c6
    k=1      guess  1.0905  2.7360  2.7622  2.5800  1.4297  1.3671
             final  2.5979  2.9956  2.7895  1.2027  1.0404  1.5657
    k=2      guess  2.9417  1.7808  1.8016  1.2168  1.2934  2.9968
             final  1.7377  2.6667  1.5583  1.8215  1.3612  2.9166
    k=3      guess  2.2311  1.9460  1.2406  2.5785  2.0796  1.1293
             final  2.7173  2.0627  1.3899  2.8765  2.0691  1.1727
    k=4      guess  2.0325  2.2409  1.6361  2.8115  2.6463  2.7540
             final  1.7377  2.6667  1.5583  1.8215  1.3612  2.9166
    k=5      guess  1.8388  1.3813  1.5247  2.4338  2.6797  1.1656
             final  2.7173  2.0627  1.3899  2.8765  2.0691  1.1727
    k=6      guess  1.6361  2.1554  1.5010  1.0524  2.0626  2.9205
             final  1.7377  2.6667  1.5583  1.8215  1.3612  2.9166
    k=7      guess  2.6344  1.7113  2.8998  2.9140  1.2303  1.5130
             final  2.5979  2.9956  2.7895  1.2027  1.0404  1.5657
    k=8      guess  2.1505  2.7423  1.3058  1.7231  1.8403  2.7888
             final  1.7377  2.6667  1.5583  1.8215  1.3612  2.9166
    k=9      guess  2.8214  2.6346  1.5496  1.3107  1.1131  2.6290
             final  1.7377  2.6667  1.5583  1.8215  1.3612  2.9166
    k=10     guess  1.7528  2.3353  2.9060  1.9883  2.0408  1.2276
             final  2.5979  2.9956  2.7895  1.2027  1.0404  1.5657

saeed                  c1      c2      c3      c4      c5      c6
    k=1      guess  1.1085  1.8726  2.1614  1.8125  1.8689  1.6631
             final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
    k=2      guess  2.2735  1.0476  2.6580  2.0756  1.7791  1.3025
             final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
    k=3      guess  1.5228  1.5698  2.7664  1.8171  1.8411  1.4402
```

```
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
k=4       guess  2.6098  1.2648  2.6177  1.9636  1.4957  1.4866
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
k=5       guess  2.1600  2.4083  2.0485  1.0599  1.1301  1.8305
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
k=6       guess  1.0412  2.0581  2.8062  2.1512  1.6583  1.5623
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
k=7       guess  1.4744  2.2015  2.1547  1.5667  1.2011  1.6824
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
k=8       guess  1.5970  1.0018  2.3994  2.9614  1.2196  1.4158
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
k=9       guess  2.2857  2.6068  2.2509  1.3955  1.1204  1.6039
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
k=10      guess  1.5571  2.8285  2.0921  1.4406  1.4951  1.5394
          final  2.5304  2.7346  2.1908  1.2411  1.1978  1.5917
```

5. Denote by p^k where $k = 1, \ldots, 10$ the ten resulting vectors obtained by Gauss–Newton optimization in the previous step. Are all the p^k equal or are some different? Note that different limits indicate the presence of local minima for the non-linear optimization problem.

Counting how many different choices of parameters appear as `final` in the table for previous question leads to the following summary:

christopher	All the p^k were equal.
eric	All the p^k were equal.
garrett	There were 2 different p^k found.
nicholas	There were 2 different p^k found.
richard	There were 3 different p^k found.
saeed	All the p^k were equal.

We note that finding the relative minima for a particular non-linear optimization problem using the above method is subject to chance as the initial guesses used to initialize the Gauss–Newton algorithm are chosen randomly. As a result, it may happen that a different number of minima are found.

To make sure all the minima have been found, the program could be run for longer to produce more possible values of p^k. Taking $k = 1, \ldots, 300$ results in

christopher	All the p^k were equal.
eric	All the p^k were equal.
garrett	There were 3 different p^k found.
nicholas	There were 2 different p^k found.
richard	There were 3 different p^k found.
saeed	All the p^k were equal.

which found only one additional local minima. Although it is possible some local minima have still been missed, since it is natural to suppose the basin of attraction for the global

minimum to be large compared to the others, it is highly likely the value for c sought after which minimizes $E(c)$ is included among the values of p^k already found.

6. For each distinct choice of parameters $c = p^k$ compute $E(c)$ and arrange the results in a well-formatted table. From these results suggest the choice of parameters c from which the data in the file was most likely to have been generated.

Plugging in the unique values of p^k found in the previous step we obtain

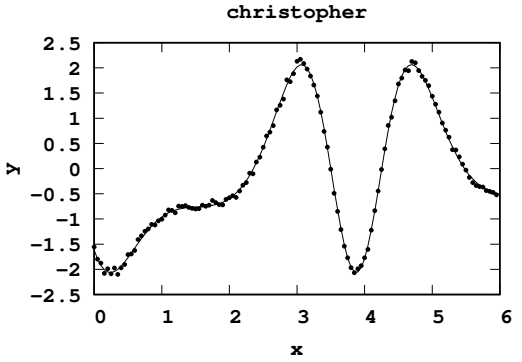
	c_1	c_2	c_3	c_4	c_5	c_6	$E(c)$
christopher	2.0637	2.6376	2.1485	1.5737	1.1006	1.3419	0.3010
eric	1.3187	1.5658	2.8692	2.9502	2.2516	1.6136	0.2994
garrett	1.2882	2.8989	2.9176	2.5951	2.6906	1.5653	87.4489
	1.5796	2.1945	2.3015	1.5804	2.1785	2.3531	80.8217
	1.6709	2.7403	1.9068	1.7971	1.8580	2.6697	0.2938
nicholas	1.6761	2.9231	1.9761	2.0277	1.5499	1.5351	6.1118
	1.7293	2.0249	2.3785	1.7990	1.9080	1.3226	0.2988
richard	1.7377	2.6667	1.5583	1.8215	1.3612	2.9166	265.9543
	2.5979	2.9956	2.7895	1.2027	1.0404	1.5657	33.4250
	2.7173	2.0627	1.3899	2.8765	2.0691	1.1727	0.3052
saeed	2.5304	2.7346	2.1908	1.2411	1.1978	1.5917	0.2928

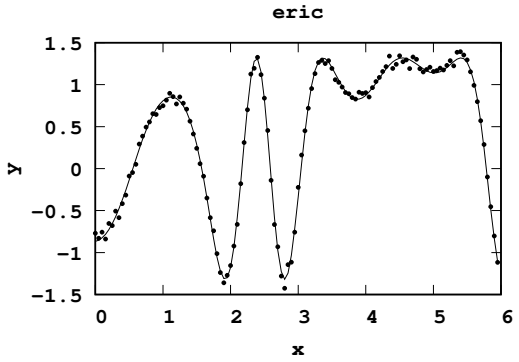
Observe for each data set in the above table that there is a choice for the parameters c such that $E(c) < 1$. These are the choices for which the data in the file was most likely to have been generated. Since the random errors η_i were independently distributed with $\sigma = 0.05$, the expected value of $E(c)$ may be estimated by

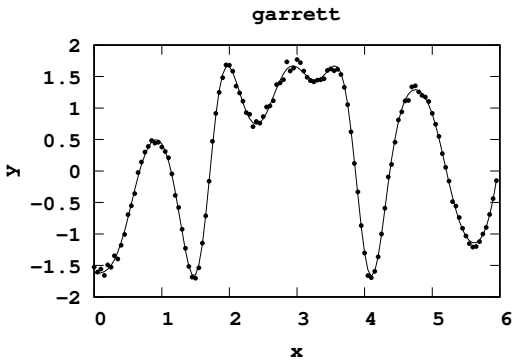
$$\mathbf{E}\big[E(c)\big] \le \mathbf{E}\Big[\sum_{i=1}^{120} \eta_i^2\Big] = \sum_{i=1}^{120} \mathbf{E}\big[\eta_i^2\big] = 120\sigma^2 = 0.3.$$

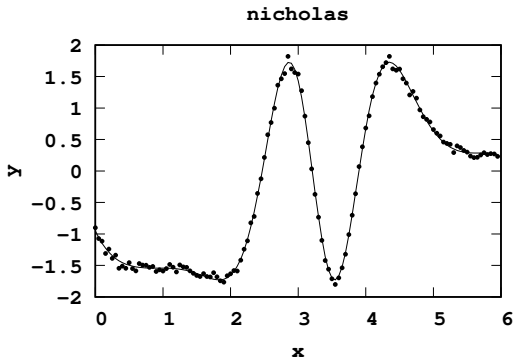
Therefore the minimum values of $E(c)$ found using Gauss–Newton non-linear optimization are consistent with the expected errors based on the noise in the data. This suggests that we have indeed found the desired global minimums.

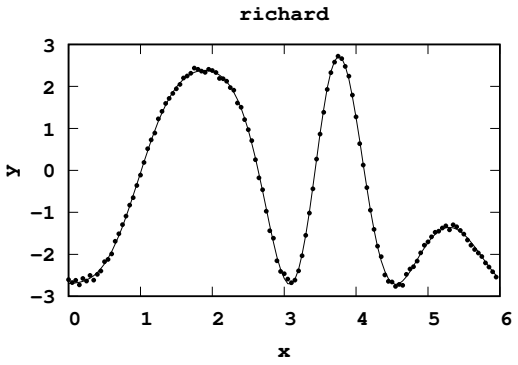
Further verification of the goodness of fit can be found by visually inspecting the data plotted along side the model for each of the corresponding choices of c found above.

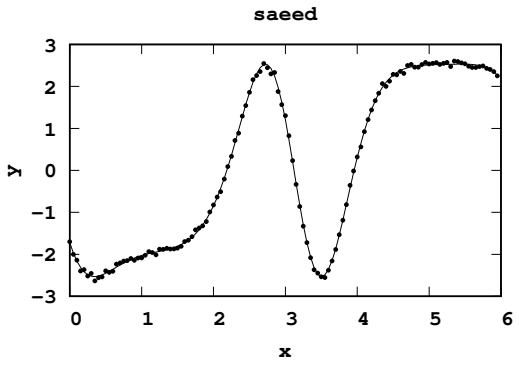





garrett
y
x
2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
0
1
2
3
4
5
6


nicholas
y
x
2
1.5
1
0.5
0
−0.5
−1
−1.5
−2
0
1
2
3
4
5
6


richard
y
x
3
2
1
0
−1
−2
−3
0
1
2
3
4
5
6


saeed
y
x
3
2
1
0
−1
−2
−3
0
1
2
3
4
5
6