

Yusuf Afolabi,
Department of Mathematics, ULL
MATH 655 FALL 2021
Project

1 Parameters Fitting To The FitzHugh-Nagumo Equations

1.1 Background

The FitzHugh-Nagumo equation is a simplification of the Hodgkin-Huxley model devised in 1952. The Hodgkin-Huxley has four variables and the FitzHugh-Nagumo equation is a reduction of that model. The reduction is from four variables to two variables where phase plane techniques may be used for the analysis of the model. The variables kept in the reduction of the model are the excitable variable and the recovery variable which are characterized as being the fast and slow variables respectively. The FitzHugh-Nagumo equations are used in computational neuroscience to model the potential in the nerve. The equations in this case are given by:

$$\frac{dv}{dt} = c(v - \frac{1}{3}v^3 + w) := f(t, v, w) \quad (1)$$

$$\frac{dw}{dt} = -\frac{1}{c}(v - a - bw) := g(t, v, w) \quad (2)$$

where v is the membrane potential and w is the recovery variable.

1.2 Synthesis Data

There are three parameters in the model, namely; a , b , and c . We are going to assume that a is fixed and fit the other parameters to the system. To get raw data, we will mimic a real life data set by first solution the equation with some parameters and perturb the solution to generate a data set that we shall take as data. Also, since the equation can not be solved exactly, we shall use the RK-4 formula to find the approximate the solution. The

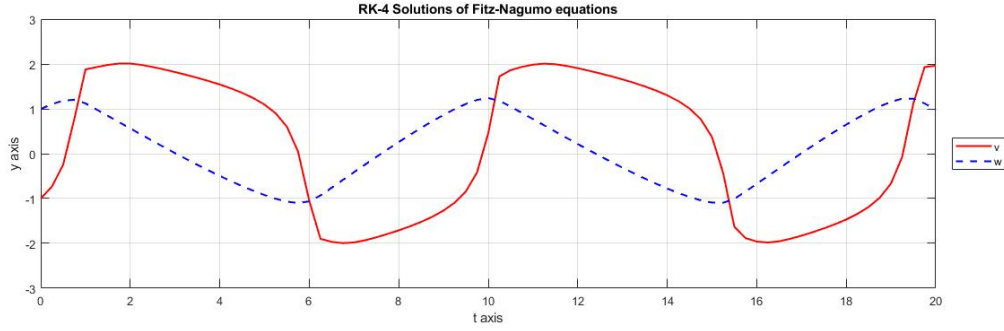


Figure 1: The numerical solution when $v(0) = -1$, $w(0) = 1$, $a = b = 0.2$, and $c = 3$.

numerical solution when $v(0) = -1$, $w(0) = 1$, $a = b = 0.2$, and $c = 3$ is given by figure (1). This is then randomize to get the data given by figure (2).

1.3 Numerical Solution by RK4

To solve the system given by (1) and (2), we use the popular Runge-Kutta formula of order 4 (RK4) given by:

$$v_{n+1} = v_n + \frac{h}{6}(m_1 + 2m_2 + 2m_3 + m_4) \quad (3)$$

$$w_{n+1} = w_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \quad (4)$$

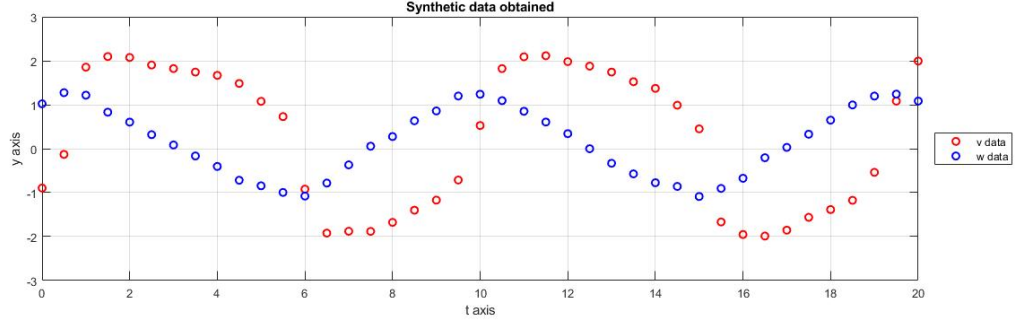


Figure 2: Data obtained by randomization

where,

$$\begin{aligned}
 m_1 &= f(t_n, v_n, w_n); \quad k_1 = g(t_n, v_n, w_n) \\
 m_2 &= f(t_n + \frac{1}{2}h, v_n + \frac{1}{2}hm_1, w_n + \frac{1}{2}hk_1); \quad k_2 = g(t_n + \frac{1}{2}h, v_n + \frac{1}{2}hm_1, w_n + \frac{1}{2}hk_1) \\
 m_3 &= f(t_n + \frac{1}{2}h, v_n + \frac{1}{2}hm_2, w_n + \frac{1}{2}hk_2); \quad k_3 = g(t_n + \frac{1}{2}h, v_n + \frac{1}{2}hm_2, w_n + \frac{1}{2}hk_2) \\
 m_4 &= f(t_n + h, v_n + hm_3, w_n + hk_3); \quad k_4 = g(t_n + h, v_n + hm_3, w_n + hk_3)
 \end{aligned}$$

1.4 Error Function

The error function to be minimized is the $l-2$ norm of the difference between the solutions of (1) and (2) and the data is denoted by $E(c, b)$ and

$$E(c, b) = \|(v(t_i), w(t_i)) - (v_i, w_i)\| \quad (5)$$

where (t_i, v_i) and (t_i, w_i) are the data, and $v(t)$ and $w(t)$ are the solutions to the system (1) and (2).

1.5 Parameters Fitting

We are going to assume that $a = 0.2$ is known, and the objective is to fit the parameter c and b to the system (1), (2). For this purpose, we are going to minimize the error function given by (5), hence we have a nonlinear regression problem to deal with.

We will use the interpolation method, that is, in order to evaluate the error function for a particular (c, b) , the system (1) and (2) are first solved numerically using RK4. The solution is then used to construct a clamped cubic spline interpolation function that can be evaluated to obtain values for v and w at the various t'_i s in the error function. Next, we are going to use different minimization routines to find the optimum and compare the performances of these routines.

1.6 Results

We now use different optimization routines to solve the problem and the results are tabulated below.

	fminsearch	fminunc	ga
(c, b) optimal	(3.0182, 0.3619)	(3.0182, 0.3619)	
Minimum $E(c, b)$	0.587467	0.587467	
Number of iteration	41	15	

1. We used the same initial condition for both "fminsearch" and "fminunc" routine and as expected "fminunc" was faster.
2. Genetic algorithm results keeps change at every run of the code. So, I could not compare it with the earlier routines.
3. I tried to use the particle swarm optimization as well, but I keep getting error. It keeps complaining about the function evaluation.

1.7 Appendices

(A.) fminsearch minimization

```

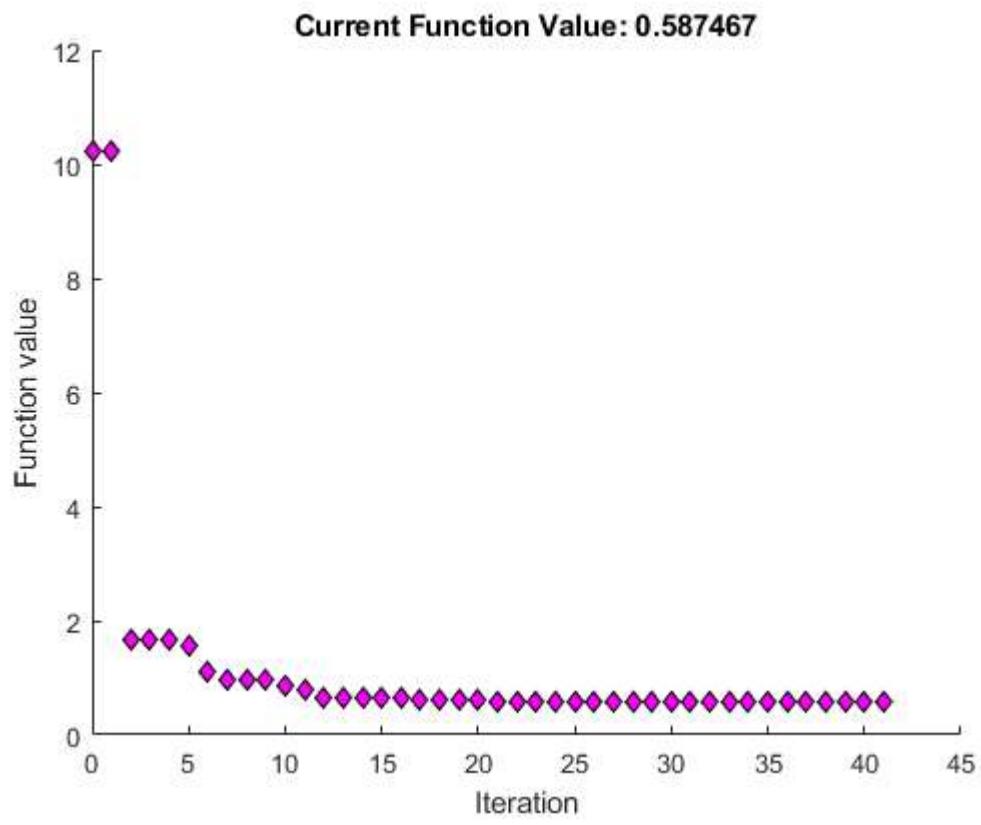
% fminsearch with Nelder-Mead
% here we use fminsearch to find the minimizer of the error function E(c,b)
% and display the iteration procedure as well as the function plot
clear
clc

load('synthetic.mat')
x0 = [3 0.5];
options = optimset('Display','iter','PlotFcns',@optimplotfval);
fun = @(t)(errorFunction2(t,dataav,dataaw,T,M));
[x,fval,exitflag,output] = fminsearch(fun,x0,options);

```

Iteration	Func-count	min f(x)	Procedure
0	1	10.2268	
1	3	10.2268	initial simplex
2	5	1.65302	expand
3	6	1.65302	reflect
4	8	1.65302	contract inside
5	10	1.55463	contract inside
6	12	1.08974	contract inside
7	14	0.959423	reflect
8	16	0.959423	contract inside
9	18	0.959423	contract inside
10	20	0.847515	expand
11	22	0.803015	reflect
12	24	0.640871	expand
13	25	0.640871	reflect
14	27	0.640871	contract inside
15	29	0.640871	contract outside
16	30	0.640871	reflect
17	32	0.597021	expand
18	33	0.597021	reflect
19	35	0.597021	contract inside
20	37	0.597021	contract inside
21	39	0.588255	reflect
22	41	0.588255	contract outside
23	42	0.588255	reflect
24	44	0.587738	contract inside
25	46	0.587738	contract inside
26	47	0.587738	reflect
27	49	0.587599	contract inside
28	51	0.587515	contract outside
29	53	0.587515	contract inside
30	55	0.587478	contract inside
31	56	0.587478	reflect
32	58	0.587478	contract inside
33	60	0.587475	contract outside
34	62	0.587469	contract inside
35	64	0.587468	contract inside
36	66	0.587468	contract outside
37	68	0.587467	contract inside
38	70	0.587467	contract inside
39	72	0.587467	contract inside
40	74	0.587467	contract outside
41	76	0.587467	contract inside

Optimization terminated:
the current x satisfies the termination criteria using OPTIONS.TolX of 1.000000e-04
and F(X) satisfies the convergence criteria using OPTIONS.TolFun of 1.000000e-04



(B.) fminunc minimization

```
% fminunc to minimize the error function E(c,b) for FitzHugh-Nagumo
% equations
clear
clc

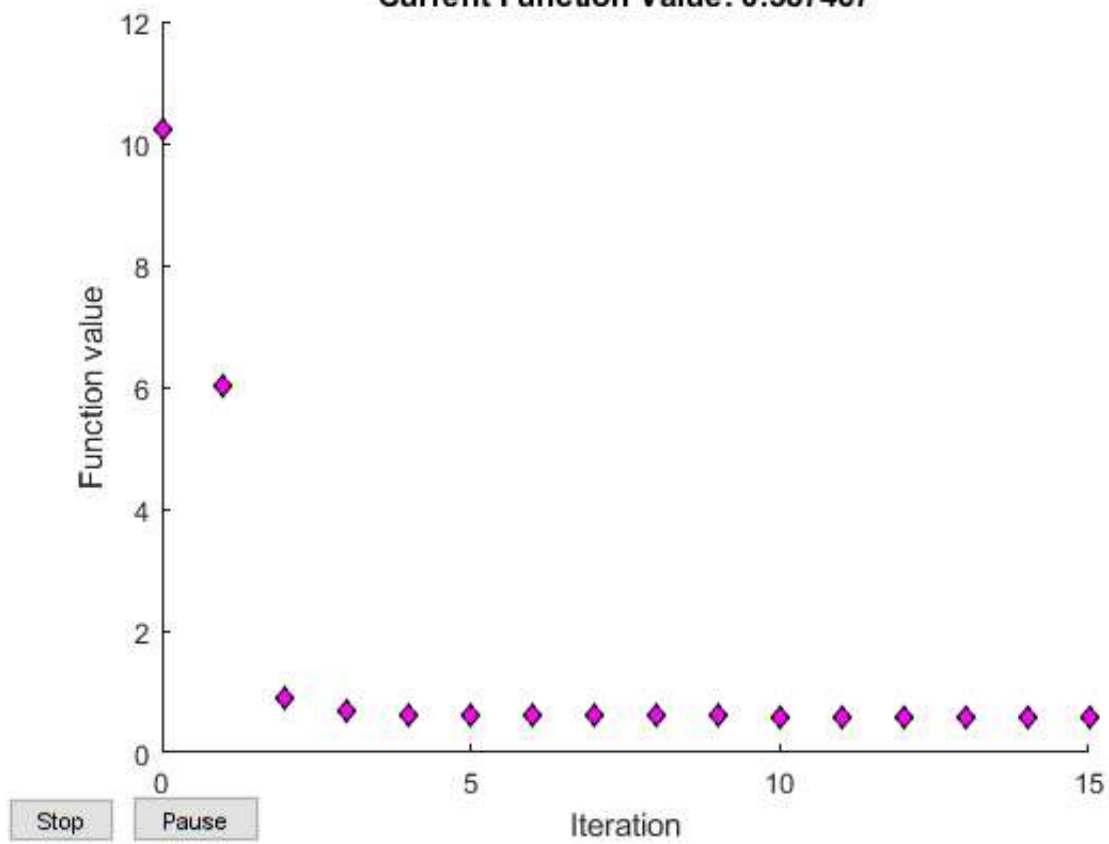
load('synthetic.mat')
x0 = [3 0.5];
options = optimset('Display','iter','PlotFcns',@optimplotfval);
fun = @(t) (errorFunction2(t,dataav,dataw,T,M));
[x,fval,exitflag,output] = fminunc(fun,x0,options);
```

Iteration	Func-count	f(x)	Step-size	First-order optimality
0	3	10.2268		152
1	9	6.01588	0.00112005	94
2	12	0.895076	1	22.8
3	15	0.684901	1	14.9
4	18	0.62015	1	2.94
5	21	0.616523	1	0.833
6	24	0.616013	1	0.624
7	27	0.613016	1	2.1
8	30	0.608419	1	3.41
9	33	0.601601	1	4.17
10	36	0.594411	1	3.37
11	39	0.588362	1	1.07
12	42	0.587524	1	0.17
13	45	0.587468	1	0.0272
14	48	0.587467	1	0.000408
15	51	0.587467	1	2.29e-05

Local minimum found.

Optimization completed because the size of the gradient is less than the default value of the optimality tolerance.

Current Function Value: 0.587467



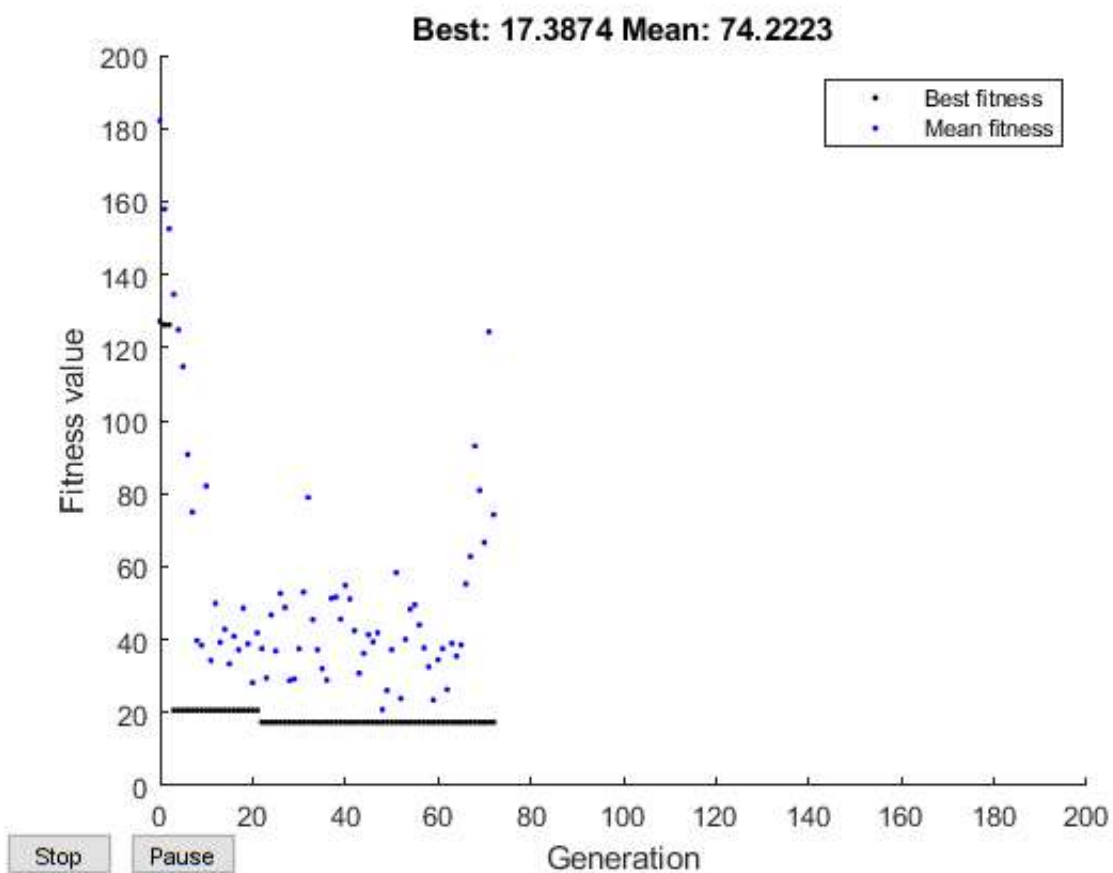
(C.) ga minimization

```

% ga to minimize the error function E(c,b) for FitzHugh-Nagumo
% equations
clear
clc
load('synthetic.mat')
x0 = [3 0.5];
options = optimoptions('ga','PlotFcn',@gaplotbestf);
fun = @(t) (errorFunction2(t,dataav,dataaw,T,M));
nvars = 2;
lb = [];
ub = [];
% [x,fval,exitflag,output] = ga(fun,nvars,lb,ub,options);
[x,fval,exitflag,output] = ga(fun,nvars,options);

```

Optimization terminated: average change in the fitness value less than options.FunctionTolerance.



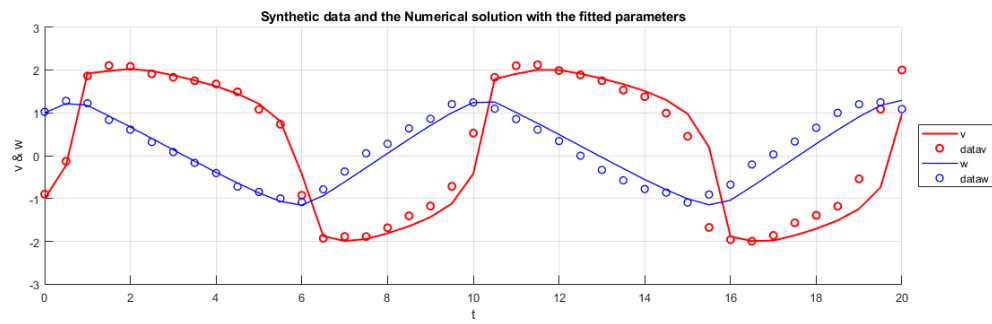


Figure 3: Data and the numerical solution with the fitted parameters