Appendix

Basic Program for estimating the parameters of action potential using GA

```
ff='testfunction'; % objective function
npar=100; % number of optimization variables
varhi=100; varlo=0; % variable limits
maxit=600; % max number of iterations
minCost=-9999999; % minimum

%_____________________________________________________
% III GA parameters
popsize=90; % set population size
mutrate=0.001; % set mutation rate
selection=2; % fraction of population kept
Nt=npar; % continuous parameter GA Nt=#variables
keep=floor(selection*popsize); % #population
% members that survive
nmut=ceil((popsize-1)*Nt*mutrate); % total number of
% mutations
M=ceil((popsize-keep)/2); % number of matings
%_____________________________________________________
% Create the initial population
iga=0; % generation counter
%initialized
par=(varhi-varlo)*rand(popsize,npar)+varlo; % random
simulationtime=5;
dt1=0.01;t1=0:dt1:simulationtime;
changetime=[0];currentlevel=[20];
I(1:500)=currentlevel;
I(501:2000)=0;
I(2001:numel(t1))=currentlevel;
Cost=0;
b=0.125*exp(Cost/80);
a=0.01*(Cost+10)/(exp((Cost+10)/10)-1);
```

```
b1=4*exp(-Cost/18);
a1=0.1*(-Cost+25)/(exp((-Cost+25)/10)-1);
b2=1/(exp((Cost+30)/10)+1);a2=0.07* exp(Cost/20);
m1(1)=a/(a+b);
n(1)=a1/(a1+b1);
h(1)=a2/(a2+b2);
 for i=1:numel(t1)-1;
       b(i)=0.125*exp(-Cost(i)/80);
a(i)=0.01*(-Cost(i)+10)/(exp((-Cost(i)+10)/10)-1);
b1(i)=4*exp(-Cost(i)/18);
a1(i)=0.1*(-Cost(i)+25)/(exp((-Cost(i)+25)/10)-1);
b2(i)=1/(exp((-Cost(i)+30)/10)+1);a2(1,i)=0.07* exp(-Cost(i)/20);
ENa = 115;
EK = -12;
El =10.6;
gNamax = 120; %max conductances
gKmax = 35;
glmax = 0.3;
gNa = gNamax*m1(i).^3*h(i);
  gK = gKmax*n(i).^4;
    gl = glmax;
    INa = (((gNa*(Cost(i)-ENa))));
    IK = gK*(Cost(i)-EK);
    Il = gl*(Cost(i)-El);
    Cost(i+1) = (dt1*(I(i)-( INa + IK + Il)))+Cost(i);
    m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i);
 end
Cost=Cost-20;
 % calculates population Cost
% using ff
[Cost,ind]=sort(Cost); % min Cost in element 1
 % sort continuous
```

```
minc(1)=min(Cost); % minc contains min of
meanc(1)=mean(Cost); % meanc contains mean of
%population
%population
% Iterate through generations
while iga<maxit
iga=iga+1; % increments generation counter
%pair and mate
M=ceil((popsize-keep)/2); % number of matings
prob=flipud([1:keep]/sum([1:keep])); % weights
% chromosomes
odds=[0 cumsum(prob(1:keep))]; % probability
% distribution
% function
pick1=rand(1,M); % mate #1
pick2=rand(1,M);
end% mate #2
% ma and pa contain the indicies of the chromosomes
% that will mate
ic=1;
while ic<=M
for id=2:keep+1
if pick1(ic)<=odds(id) && pick1(ic)>odds(id-1)
ma(ic)=id-1;
end
if pick2(ic)<=odds(id) && pick2(ic)>odds(id-1)
pa(ic)=id-1;
end
end
ic=ic+1;
end
% Performs mating using single point crossover
ix=1:2:keep; % index of mate #1
xp=ceil(rand(1,M)*Nt); % crossover point
```

```
r=rand(1,M); % mixing parameter
for ic=1:M
xy=par(ma(ic),xp(ic))-par(pa(ic),xp(ic)); % ma and pa
% mate
par(keep+ix(ic),:)=par(ma(ic),:); % 1st offspring
par(keep+ix(ic)+1,:)=par(pa(ic),:); % 2nd offspring
par(keep+ix(ic),xp(ic))=par(ma(ic),xp(ic))-r(ic).*xy;
% 1st
par(keep+ix(ic)+1,xp(ic))=par(pa(ic),xp(ic))+r(ic).*xy;
% 2nd
if xp(ic)<npar % crossover when last variable not
%selected
par(keep+ix(ic),:)=[par(keep+ix(ic),1:xp(ic))
par(keep+ix(ic)+1,xp(ic)+1:npar)];
par(keep+ix(ic)+1,:)=[par(keep+ix(ic)+1,1:xp(ic))
par(keep+ix(ic),xp(ic)+1:npar)];
end % if
end
% Mutate the population
mrow=sort(ceil(rand(1,nmut)*(popsize-1))+1);
mcol=ceil(rand(1,nmut)*Nt);
% ma and pa contain the indicies of the chromosomes
% that will mate
ic=1;
while ic<=M
for id=2:keep+1
if pick1(ic)<=odds(id) && pick1(ic)>odds(id-1)
ma(ic)=id-1;
end
if pick2(ic)<=odds(id) && pick2(ic)>odds(id-1)
pa(ic)=id-1;
end
end
ic=ic+1;
```

```
end
% The new offspring and mutated chromosomes are
% evaluated
simulationtime=5;
dt1=0.01;t1=0:dt1:simulationtime;
changetime=[0];currentlevel=[20];
I(1:500)=currentlevel;
I(501:2000)=0;
I(2001:numel(t1))=currentlevel;
Cost=0;
b=0.125*exp(Cost/80);
a=0.01*(Cost+10)/(exp((Cost+10)/10)-1);
b1=4*exp(-Cost/18);
a1=0.1*(-Cost+25)/(exp((-Cost+25)/10)-1);
b2=1/(exp((Cost+30)/10)+1);a2=0.07* exp(Cost/20);
m1(1)=a/(a+b);
n(1)=a1/(a1+b1);
h(1)=a2/(a2+b2);
   for i=1:numel(t1)-1;
     b(i)=0.125*exp(-Cost(i)/80);
a(i)=0.01*(-Cost(i)+10)/(exp((-Cost(i)+10)/10)-1);
b1(i)=4*exp(-Cost(i)/18);
a1(i)=0.1*(-Cost(i)+25)/(exp((-Cost(i)+25)/10)-1);
b2(i)=1/(exp((-Cost(i)+30)/10)+1);a2(1,i)=0.07* exp(-Cost(i)/20);
ENa = 115;
EK = -12;
El =10.6;
gNamax = 120; %max conductances
gKmax = 35;
glmax = 0.3;
gNa = gNamax*m1(i).^3*h(i);
   gK = gKmax*n(i).^4;
   gl = glmax;
   INa = (((gNa*(Cost(i)-ENa))));
```

```
    IK = gK*(Cost(i)-EK);
    Il = gl*(Cost(i)-El);
    Cost(i+1) = (dt1*(I(i)-( INa + IK + Il)))+Cost(i);
    m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i);
end
Cost=Cost-20;
% Sort the Costs and associated parameters
% [Cost,ind]=sort(Cost);
%_______________________________________________________
% Do statistics for a single nonaveraging run
minc(iga+1)=min(Cost);
meanc(iga+1)=mean(Cost);
%_______________________________________________________
% Stopping criteria
if iga>maxit || Cost(1)<minCost
break
end
[iga Cost(1)];
 %iga
num2str(Cost(1))
disp(['best solution'])
disp([num2str(par(1,:))])
disp('continuous genetic algorithm')
simulationtime=5;
dt1=0.01;t1=0:dt1:simulationtime;
changetime=[0];currentlevel=[20];
I(1:500)=currentlevel;
I(501:2000)=0;
I(2001:numel(t1))=currentlevel;
V=0;
b=0.125*exp(V/80);
a=0.01*(V+10)/(exp((V+10)/10)-1);
```

```
b1=4*exp(-V/18);
a1=0.1*(-V+25)/(exp((-V+25)/10)-1);
b2=1/(exp((V+30)/10)+1);a2=0.07* exp(V/20);
m1(1)=a/(a+b);
n(1)=a1/(a1+b1);
h(1)=a2/(a2+b2);
   for i=1:numel(t1)-1;
      b(i)=0.125*exp(-V(i)/80);
a(i)=0.01*(-V(i)+10)/(exp((-V(i)+10)/10)-1);
b1(i)=4*exp(-V(i)/18);
a1(i)=0.1*(-V(i)+25)/(exp((-V(i)+25)/10)-1);
b2(i)=1/(exp((-V(i)+30)/10)+1);a2(1,i)=0.07* exp(-V(i)/20);
ENa = 115;
EK = -12;
El =10.6;
gNamax = 120; %max conductances
gKmax = 36;
glmax = 0.3;
gNa = gNamax*m1(i).^3*h(i);
   gK = gKmax*n(i).^4;
   gl = glmax;
   INa = (((gNa*(V(i)-ENa))));
   IK = gK*(V(i)-EK);
   Il = gl*(V(i)-El);
   V(i+1) = (dt1*(I(i)-( INa + IK + Il)))+V(i);
   m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i);
end
V=V-20;
plot(t1,Cost,t1,V)
legend('Estimated','Reference Signal')
xlabel( 'Time in milliseconds');ylabel('Action Potential(mV)')
```

Basic Program for estimating the parameters of action potential using PSO

```
clear
ff = 'testfunction'; % Objective Function
% Initializing variables
popsize = 3000; % Size of the swarm
npar = 5; % Dimension of the problem
maxit = 600; % Maximum number of iterations
c1 = 1; % cognitive parameter
c2 = 4-c1; % social parameter
C=1; % constriction factor
% Initializing swarm and velocities
par=rand(popsize,npar); % random population of
% continuous values
vel = rand(popsize,npar); % random velocities
% Evaluate initial population
simulationtime=5;
dt1=0.01;t1=0:dt1:simulationtime;
changetime=[0];currentlevel=[20];
I(1:500)=currentlevel;
I(501:2000)=0;
I(2001:numel(t1))=currentlevel;
Cost=0;
b=0.125*exp(Cost/80);
a=0.01*(Cost+10)/(exp((Cost+10)/10)-1);
b1=4*exp(-Cost/18);
a1=0.1*(-Cost+25)/(exp((-Cost+25)/10)-1);
b2=1/(exp((Cost+30)/10)+1);a2=0.07* exp(Cost/20);
m1(1)=a/(a+b);
n(1)=a1/(a1+b1);
h(1)=a2/(a2+b2);
   for i=1:numel(t1)-1;
      b(i)=0.125*exp(-Cost(i)/80);
```

```
a(i)=0.01*(-Cost(i)+10)/(exp((-Cost(i)+10)/10)-1);
b1(i)=4*exp(-Cost(i)/18);
a1(i)=0.1*(-Cost(i)+25)/(exp((-Cost(i)+25)/10)-1);
b2(i)=1/(exp((-Cost(i)+30)/10)+1);a2(1,i)=0.07* exp(-Cost(i)/20);
ENa = 115;
EK = -12;
El =10.6;
gNamax = 110; %max conductances
gKmax = 35;
glmax = 0.3;
gNa = gNamax*m1(i).^3*h(i);
   gK = gKmax*n(i).^4;
   gl = glmax;
   INa = (((gNa*(Cost(i)-ENa))));
   IK = gK*(Cost(i)-EK);
   Il = gl*(Cost(i)-El);
   Cost(i+1) = (dt1*(I(i)-( INa + IK + Il)))+Cost(i);
   m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i);
end
Cost=Cost-20;
 % calculates population Cost using
% ff
minc(1)=min(Cost); % min Cost
meanc(1)=mean(Cost); % mean Cost
globalmin=minc(1); % initialize global minimum
% Initialize local minimum for each particle
localpar = par; % location of local minima
localCost = Cost; % Cost of local minima
% Finding best particle in initial population
[globalCost,indx] = min(Cost);
globalpar=par(indx,:);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%
% Start iterations
iter = 0; % counter
while iter < maxit
iter = iter + 1;
% update velocity = vel
w=(maxit-iter)/maxit; %inertia weiindxht
r1 = rand(popsize,npar); % random numbers
r2 = rand(popsize,npar); % random numbers
vel = C*(w*vel + c1 *r1.*(localpar-par) +c2*r2.*(ones(popsize,1)*globalpar-par));
% update particle positions
par = par + vel; % updates particle position
overlimit=par<=1;
underlimit=par>=0;
par=par.*overlimit+not(overlimit);
par=par.*underlimit;
% Evaluate the new swarm
simulationtime=5;
dt1=0.01;t1=0:dt1:simulationtime;
changetime=[0];currentlevel=[20];
I(1:500)=currentlevel;
I(501:2000)=0;
I(2001:numel(t1))=currentlevel;
Cost=0;
b=0.125*exp(Cost/80);
a=0.01*(Cost+10)/(exp((Cost+10)/10)-1);
b1=4*exp(-Cost/18);
a1=0.1*(-Cost+25)/(exp((-Cost+25)/10)-1);
b2=1/(exp((Cost+30)/10)+1);a2=0.07* exp(Cost/20);
m1(1)=a/(a+b);
n(1)=a1/(a1+b1);
h(1)=a2/(a2+b2);
   for i=1:numel(t1)-1;
```

```
    b(i)=0.125*exp(-Cost(i)/80);
a(i)=0.01*(-Cost(i)+10)/(exp((-Cost(i)+10)/10)-1);
b1(i)=4*exp(-Cost(i)/18);
a1(i)=0.1*(-Cost(i)+25)/(exp((-Cost(i)+25)/10)-1);
b2(i)=1/(exp((-Cost(i)+30)/10)+1);a2(1,i)=0.07* exp(-Cost(i)/20);
ENa = 115;
EK = -12;
El =10.6;
gNamax = 110; %max conductances
gKmax = 35;
glmax = 0.3;
gNa = gNamax*m1(i).^3*h(i);
  gK = gKmax*n(i).^4;
  gl = glmax;
  INa = (((gNa*(Cost(i)-ENa))));
  IK = gK*(Cost(i)-EK);
  Il = gl*(Cost(i)-El);
  Cost(i+1) = (dt1*(I(i)-( INa + IK + Il)))+Cost(i);
  m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i)
end
Cost=Cost-20;
% calculating alphas abd betas for the initial membrane voltage
 % evaluates Cost of swarm
% Updating the best local position for each particle
betterCost = Cost < localCost;
localCost = localCost.*not(betterCost) +Cost.*betterCost;
localpar((betterCost),:) =par((betterCost),:);
% Updating index g
[temp, t] = min(localCost);
if temp<globalCost
globalpar=par(t,:); indx=t; globalCost=temp;
end
```

```
[iter globalpar globalCost]; % print output each
% iteration
minc(iter+1)=min(Cost); % min for this
% iteration
globalmin(iter+1)=globalCost; % best min so far
meanc(iter+1)=mean(Cost); % avg. Cost for
% this iteration
end% while
iters=0:length(minc)-1;
simulationtime=5;
dt1=0.01;t1=0:dt1:simulationtime;
changetime=[0];currentlevel=[20];
I(1:500)=currentlevel;
I(501:2000)=0;
I(2001:numel(t1))=currentlevel;
V=0;
b=0.125*exp(V/80);
a=0.01*(V+10)/(exp((V+10)/10)-1);
b1=4*exp(-V/18);
a1=0.1*(-V+25)/(exp((-V+25)/10)-1);
b2=1/(exp((V+30)/10)+1);a2=0.07* exp(V/20);
m1(1)=a/(a+b);
n(1)=a1/(a1+b1);
h(1)=a2/(a2+b2);
   for i=1:numel(t1)-1;
      b(i)=0.125*exp(-V(i)/80);
a(i)=0.01*(-V(i)+10)/(exp((-V(i)+10)/10)-1);
b1(i)=4*exp(-V(i)/18);
a1(i)=0.1*(-V(i)+25)/(exp((-V(i)+25)/10)-1);
b2(i)=1/(exp((-V(i)+30)/10)+1);a2(1,i)=0.07* exp(-V(i)/20);
ENa = 115;
EK = -12;
El =10.6;
gNamax = 120; %max conductances
```

```
gKmax = 36;
glmax = 0.3;
gNa = gNamax*m1(i).^3*h(i);
   gK = gKmax*n(i).^4;
   gl = glmax;
   INa = (((gNa*(V(i)-ENa))));
   IK = gK*(V(i)-EK);
   Il = gl*(V(i)-El);
   V(i+1) = (dt1*(I(i)-( INa + IK + Il)))+V(i);
   m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i);
end
V=V-20;
plot(t1,Cost,t1,V)
legend('Estimated','Reference Signal')
xlabel( 'Time in milliseconds');ylabel('-V(mV)')
```

Basic Program for estimating parameters for action potential using FA

```
clc;

clear;

close all;

%% Problem Definition

simulationtime=5;

dt1=0.01;t1=0:dt1:simulationtime;

changetime=[0];currentlevel=[20];

I(1:500)=currentlevel;

I(501:2000)=0;

I(2001:numel(t1))=currentlevel;
```

```
Cost=0;

b=0.125*exp(Cost/80);

a=0.01*(Cost+10)/(exp((Cost+10)/10)-1);

b1=4*exp(-Cost/18);

a1=0.1*(-Cost+25)/(exp((-Cost+25)/10)-1);

b2=1/(exp((Cost+30)/10)+1);a2=0.07* exp(Cost/20);

m1(1)=a/(a+b);

n(1)=a1/(a1+b1);

h(1)=a2/(a2+b2);

   for i=1:numel(t1)-1;

      b(i)=0.125*exp(-Cost(i)/80);

a(i)=0.01*(-Cost(i)+10)/(exp((-Cost(i)+10)/10)-1);

b1(i)=4*exp(-Cost(i)/18);

a1(i)=0.1*(-Cost(i)+25)/(exp((-Cost(i)+25)/10)-1);

b2(i)=1/(exp((-Cost(i)+30)/10)+1);a2(1,i)=0.07* exp(-Cost(i)/20);

ENa = 115;

EK = -12;

El =10.6;

gNamax = 120; %max conductances

gKmax = 35;

glmax = 0.3;

gNa = gNamax*m1(i).^3*h(i);

   gK = gKmax*n(i).^4;

   gl = glmax;
   INa = (((gNa*(Cost(i)-ENa))));
   IK = gK*(Cost(i)-EK);
   Il = gl*(Cost(i)-El);
 Cost(i+1) = (dt1*(I(i)-( INa + IK + Il)))+Cost(i);
   m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i);
end
Cost=Cost-20;
  % Cost Function
nVar=5;                % Number of Decision Variables
VarSize=[1 nVar];      % Decision Variables Matrix Size
VarMin=-10;            % Decision Variables Lower Bound
VarMax= 10;            % Decision Variables Upper Bound
%% Firefly Algorithm Parameters
MaxIt=200;        % Maximum Number of Iterations
nPop=25;          % Number of Fireflies (Swarm Size)
gamma=1;          % Light Absorption Coefficient
beta0=2;          % Attraction Coefficient Base Value
alpha=0.2;        % Mutation Coefficient
alpha_damp=0.98;    % Mutation Coefficient Damping Ratio
delta=0.05*(VarMax-VarMin);     % Uniform Mutation Range
m=2;
if isscalar(VarMin) && isscalar(VarMax)
```

```
    dmax = (VarMax-VarMin)*sqrt(nVar);
else
    dmax = norm(VarMax-VarMin);
end
%% Initialization
% Empty Firefly Structure
firefly.Position=[];
firefly.Cost=[];
% Initialize Population Array
pop=repmat(firefly,nPop,1);
% Initialize Best Solution Ever Found
BestSol.Cost=inf;
% Create Initial Fireflies
for i=1:nPop
   pop(i).Position=unifrnd(VarMin,VarMax,VarSize);
  % pop(i).Cost=CostFunction(pop(i).Position);
   if pop(i).Cost<=BestSol.Cost
       BestSol=pop(i);
   end
end
% Array to Hold Best Cost Values
BestCost=zeros(MaxIt,1);
%% Firefly Algorithm Main Loop
for it=1:MaxIt
```

```
newpop=repmat(firefly,nPop,1);
for i=1:nPop
   newpop(i).Cost = inf;
   for j=1:nPop
      if pop(j).Cost < pop(i).Cost
         rij=norm(pop(i).Position-pop(j).Position)/dmax;
         beta=beta0*exp(-gamma*rij^m);
         e=delta*unifrnd(-1,+1,VarSize);
         %e=delta*randn(VarSize);
         newsol.Position = pop(i).Position ...
                     + beta*rand(VarSize).*(pop(j).Position-pop(i).Position) ..
         newsol.Position=max(newsol.Position,VarMin);
         newsol.Position=min(newsol.Position,VarMax);
         newsol.Cost=CostFunction(newsol.Position);
         if newsol.Cost <= newpop(i).Cost
            newpop(i) = newsol;
            if newpop(i).Cost<=BestSol.Cost
               BestSol=newpop(i)
             end
         end
      end
   end
end
% Merge
```

```
    pop=[pop newpop];
    % Sort
    % Truncate
    pop=pop(1:nPop);
    % Store Best Cost Ever Found
    % Show Iteration Information
    disp(['Iteration ' num2str(it) ': Best Cost = ' num2str(BestCost(it))]);
    % Damp Mutation Coefficient
    alpha = alpha*alpha_damp;
end
simulationtime=5;
dt1=0.01;t1=0:dt1:simulationtime;
changetime=[0];currentlevel=[20];
I(1:500)=currentlevel;
I(501:2000)=0;
I(2001:numel(t1))=currentlevel;
V=0;
b=0.125*exp(V/80);
a=0.01*(V+10)/(exp((V+10)/10)-1);
b1=4*exp(-V/18);
a1=0.1*(-V+25)/(exp((-V+25)/10)-1);
b2=1/(exp((V+30)/10)+1);a2=0.07* exp(V/20);
m1(1)=a/(a+b);
n(1)=a1/(a1+b1);
```

```
h(1)=a2/(a2+b2);
   for i=1:numel(t1)-1;
      b(i)=0.125*exp(-V(i)/80);
a(i)=0.01*(-V(i)+10)/(exp((-V(i)+10)/10)-1);
b1(i)=4*exp(-V(i)/18);
a1(i)=0.1*(-V(i)+25)/(exp((-V(i)+25)/10)-1);
b2(i)=1/(exp((-V(i)+30)/10)+1);a2(1,i)=0.07* exp(-V(i)/20);
ENa = 115;
EK = -12;
El =10.6;
gNamax = 120; %max conductances
gKmax = 36;
glmax = 0.3;
gNa = gNamax*m1(i).^3*h(i);
   gK = gKmax*n(i).^4;
   gl = glmax;
   INa = (((gNa*(V(i)-ENa))));
   IK = gK*(V(i)-EK);
   Il = gl*(V(i)-El);
   V(i+1) = (dt1*(I(i)-( INa + IK + Il)))+V(i);
   m1(i+1)=((a1(i).*(1-m1(i)))-(b1(1,i).*m1(i))).*dt1+m1(i);
n(i+1)=((a(1,i).*(1-n(i)))-(b(1,i).*n(i))).*dt1+n(i);
h(i+1)=((a2(1,i).*(1-h(i)))-(b2(1,i).*h(i))).*dt1+h(i);
end
```

```
V=V-20;

plot(t1,Cost,t1,V)

legend('Estimated','Reference Signal')

xlabel( 'Time in milliseconds');

ylabel('-V(mV)')
```