ANNEXE 1:

Quelques jeux de paramètres:

```
%Paramètres du modèle "regular spiking"

a = 0.02;
b = 0.2 ;
c=-65;
d= 8;
a1=0.04;
a2=5;
C=140;
Ts=3000;
N=10000;
Te=Ts/N;
I=8;
```

(par la suite, quand I n'est pas précisé, on prendra I=8 par défaut)

Paramètres du modèle "intrinsically bursting"

a = 0.02;
b = 0.2 ;
c=-55;
d= 4;
I=10;

Paramètres du modèle "chattering"

a = 0.02;
b = 0.2 ;
c=-50;
d= 2;
I=10;

ANNEXE 2

schéma simulink du neurone:
On utilise la méthode d'intégration Runge-Kutta sinon on obtient des erreurs

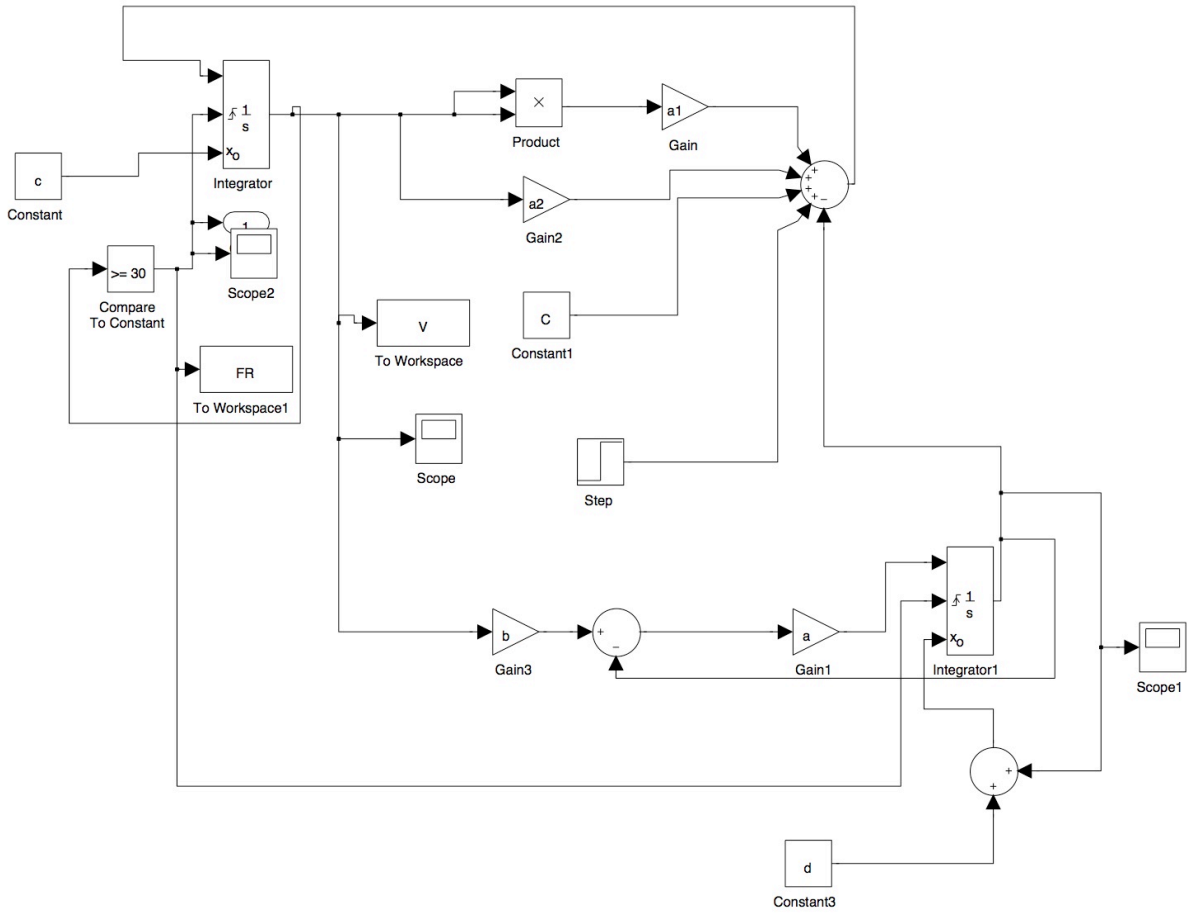
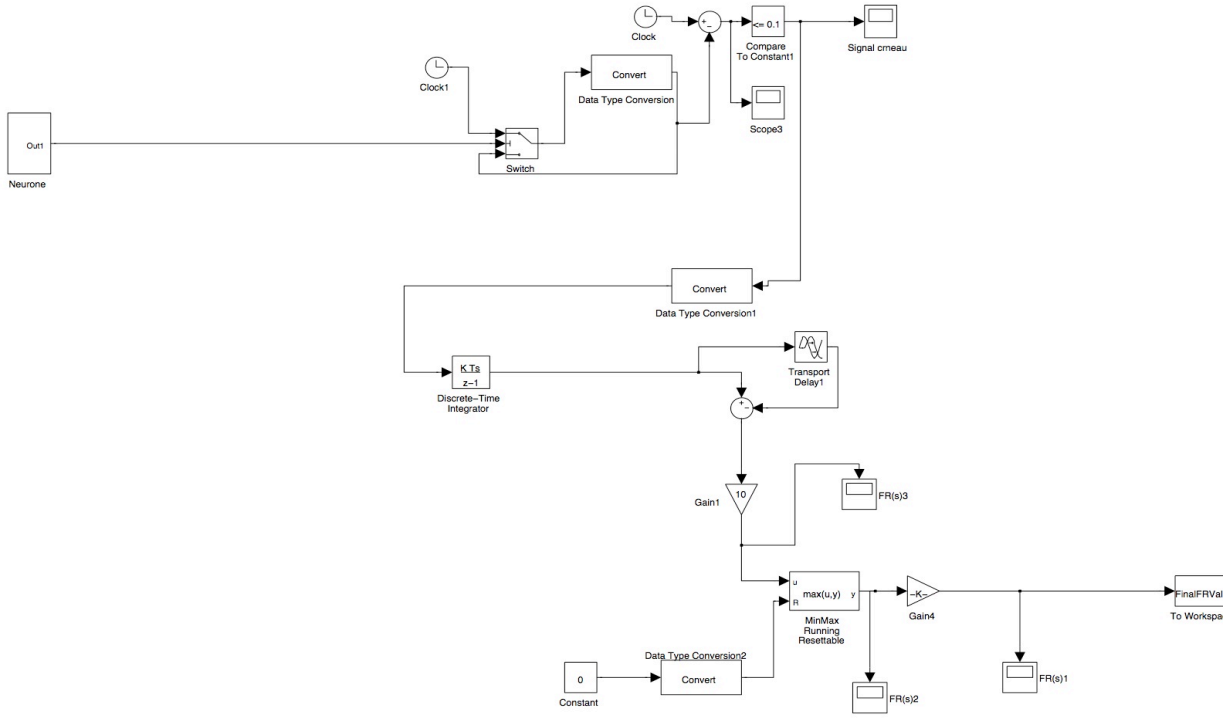


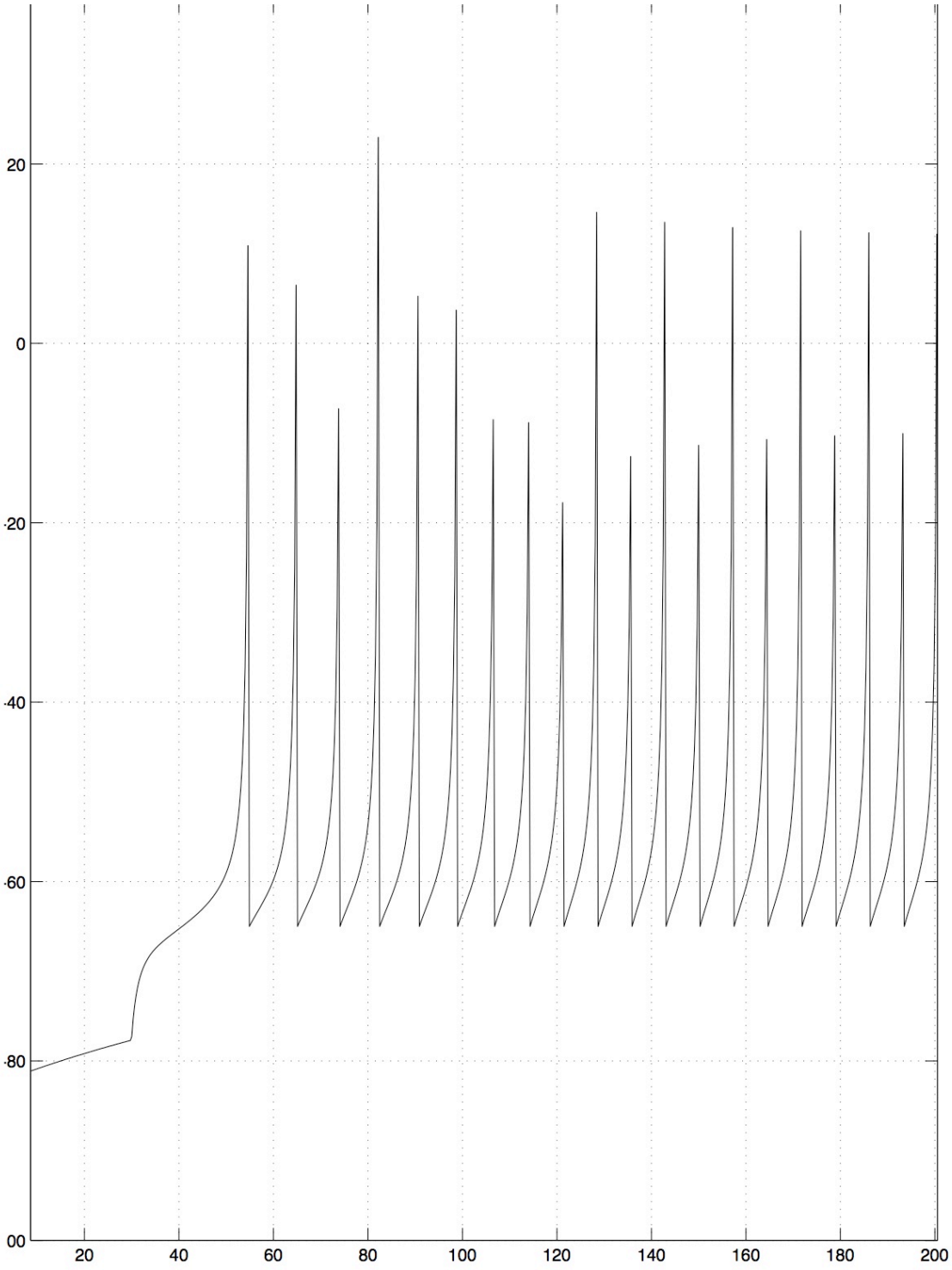
schéma simulink du dispositif de comptage:

On utilise un système qui crée un créneau très court à chaque pic. En comptant (en intégrant) le nombre de créneaux, on retrouve le Firing Rate

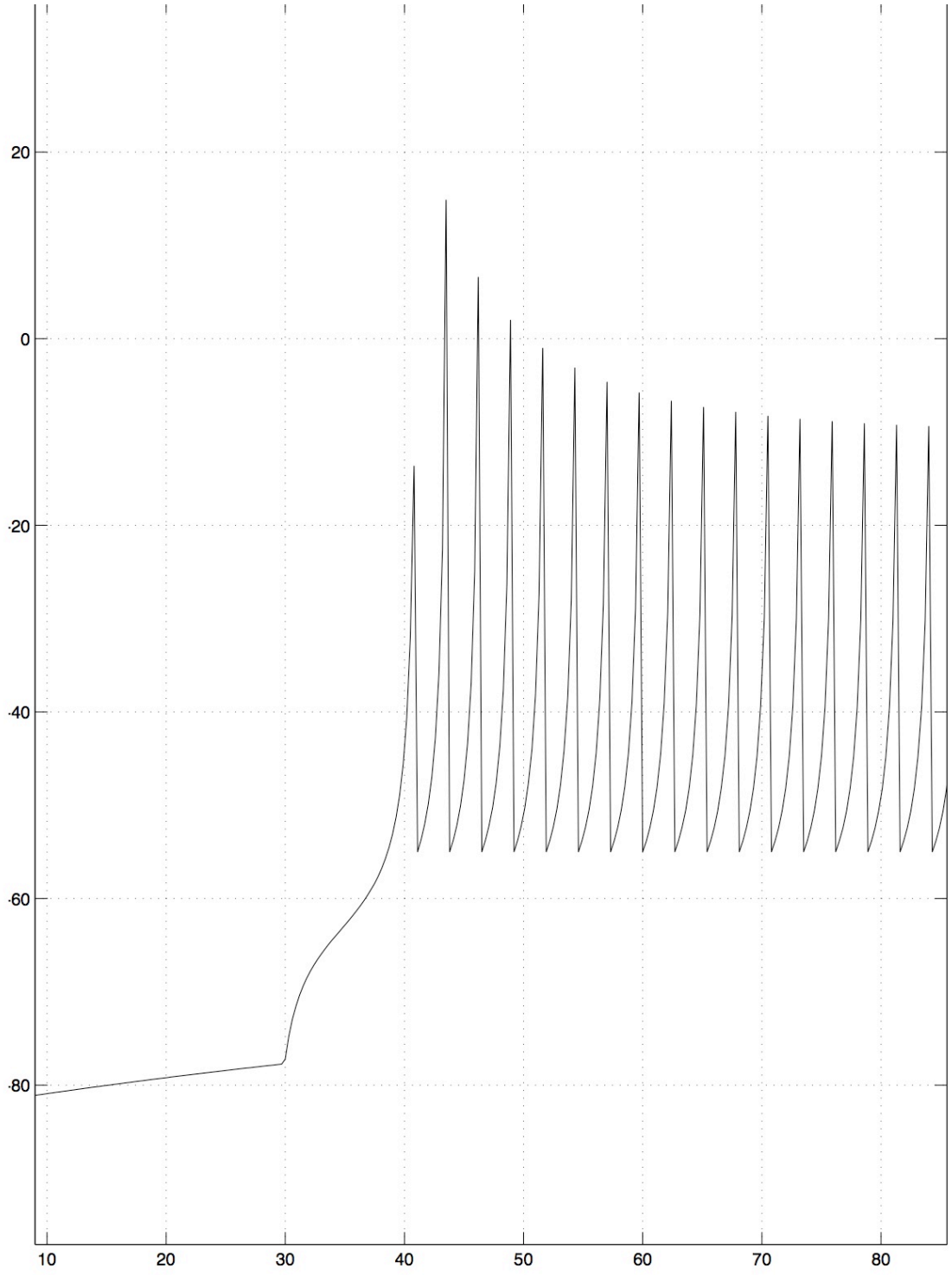


courbes obtenues pour chacun des modèles: V=f(t), V en mV, t en ms

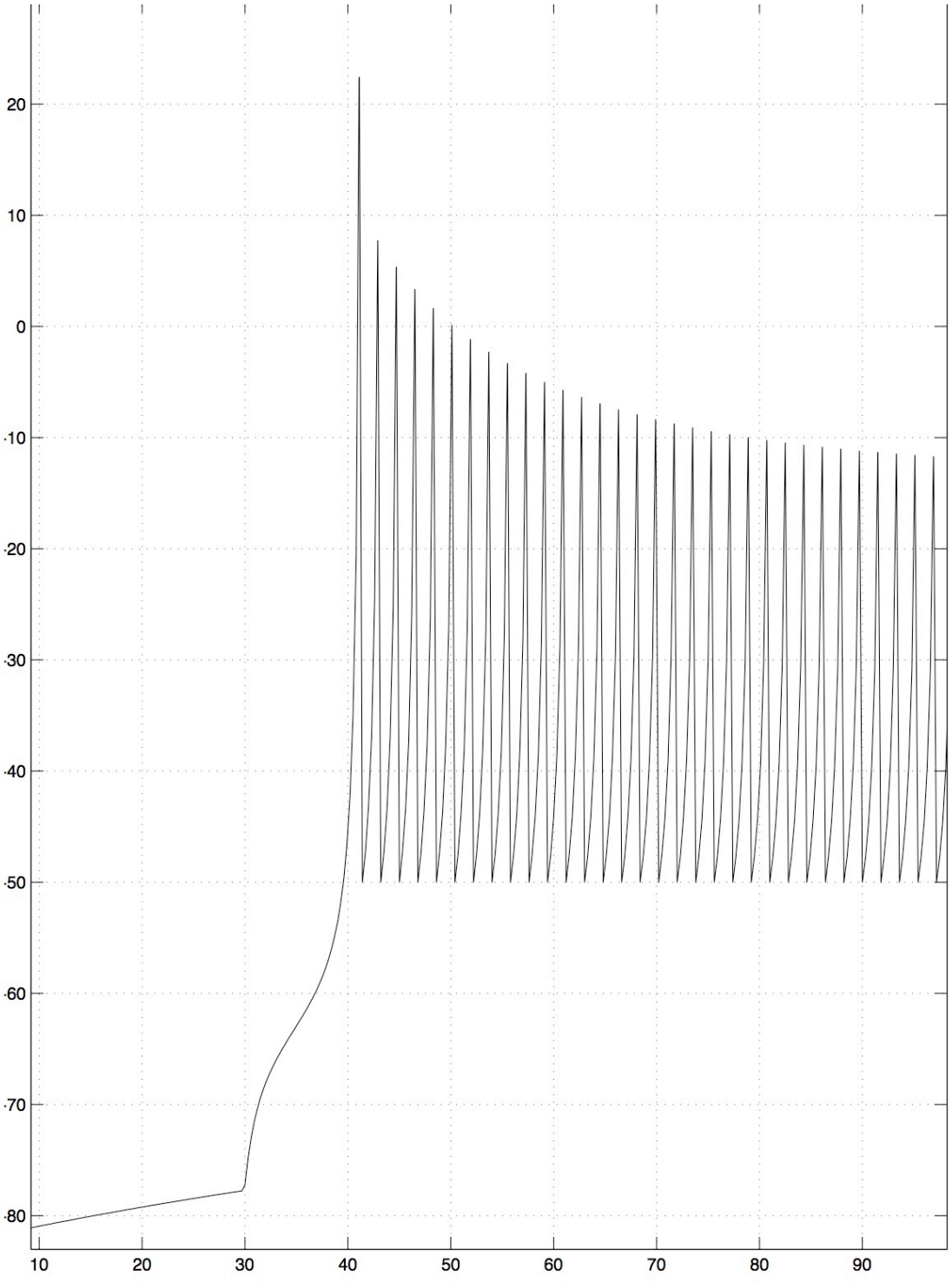
temps de départ de l'échelon de I =30 ms
10000points;
T simulation =150ms



regular spiking



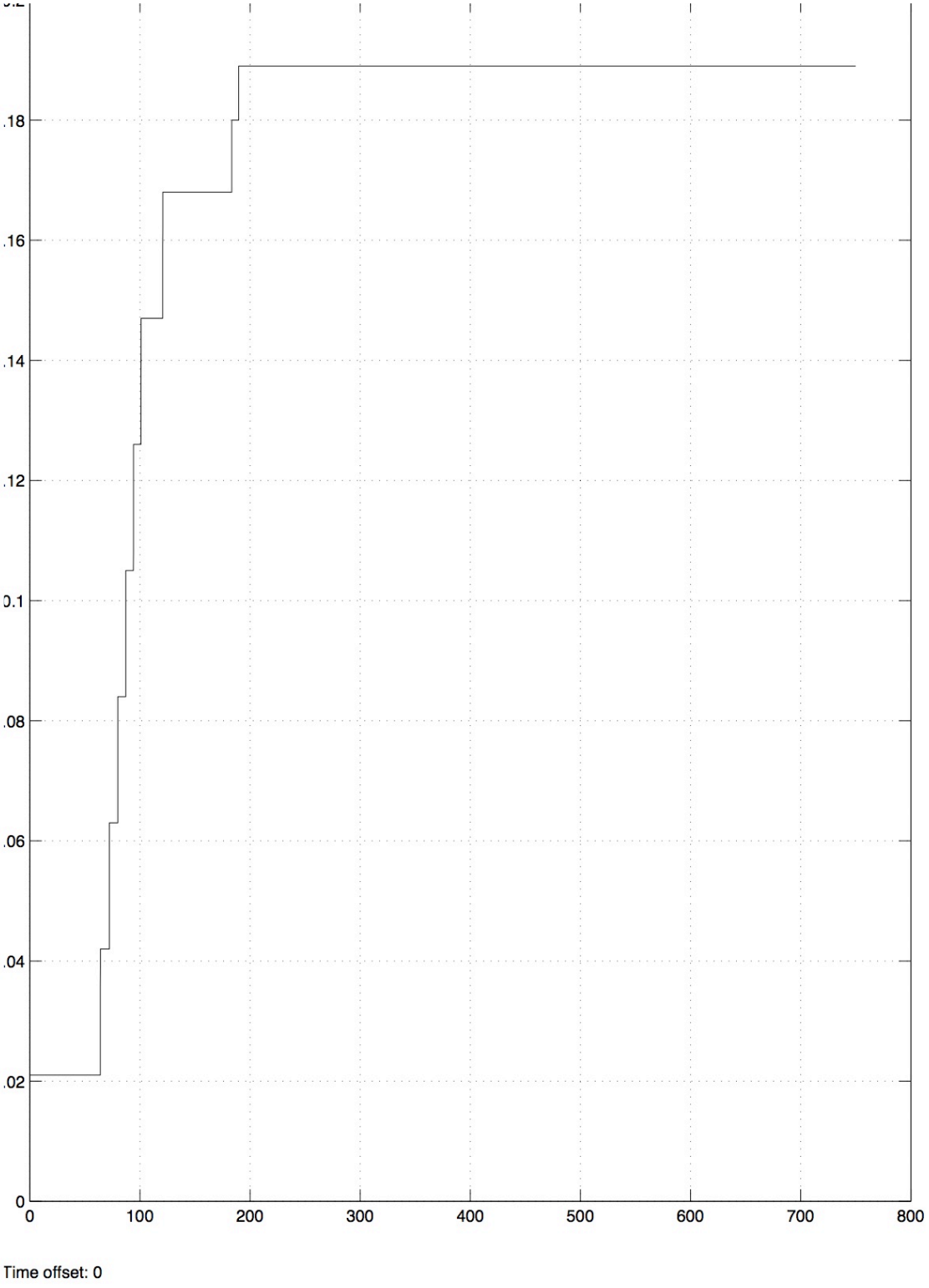
intrinsically bursting


20
10
0
-10
-20
-30
-40
-50
-60
-70
-80
10
20
30
40
50
60
70
80
90
Time offset: 0

Chattering

résultat du Firing Rate

pour une simulation de 50 000pts, 150ms, fenêtre 50ms, avec le max en temps réel du FR, pour le regular spiking: La forme est cohérente, car les spikes sont discontinus.

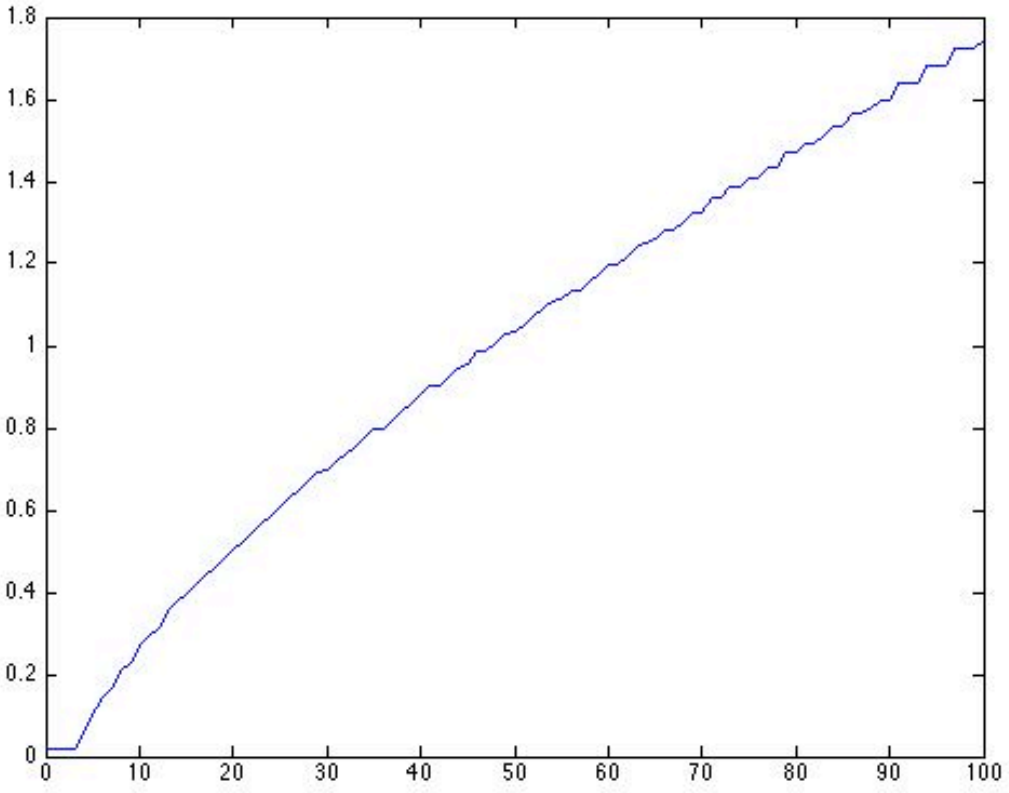


ANNEXE 3:

```
%Obtention de la sigmoïde:
On trace les Firing Rates finaux (donc maximaux ici), en fonction de I
```

```
VI=(0:1:100);
VFR=eye(1,length(VI));
for i = 1 : length(VI)
    I=i;
    sim('projet_neurones_simulation1');
    VFR(i)=FinalFRValue(length(FinalFRValue));
end
plot(VI,VFR)
```

courbe obtenue: Firing Rate = f(I), I en mA et Firing Rate en pics/milisecondes pour des simulations de 150ms, fenêtre de 50ms, courbe finale de 101 points (101 simulations au total) --> La courbe est croissante de pente de moins en moins élevée.



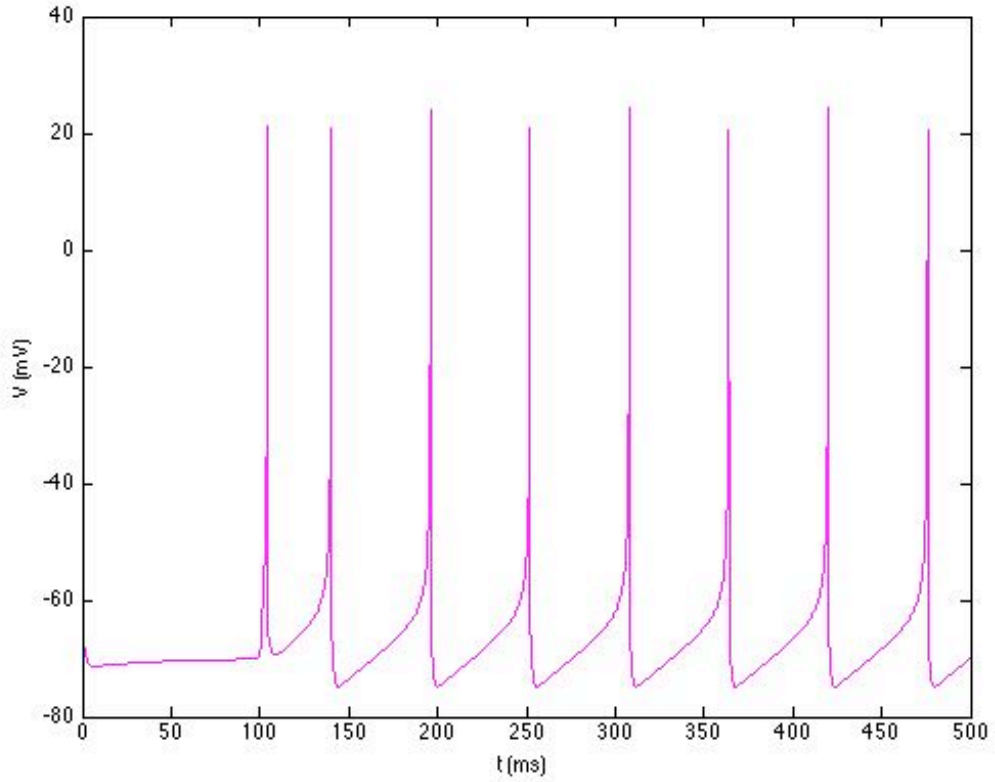
ANNEXE 4:

Programme Matlab de calcul de V:
L'équation différentielle est résolue par la méthode d'Euler et on injecte un échelon qui commence à tstep dans le neurone

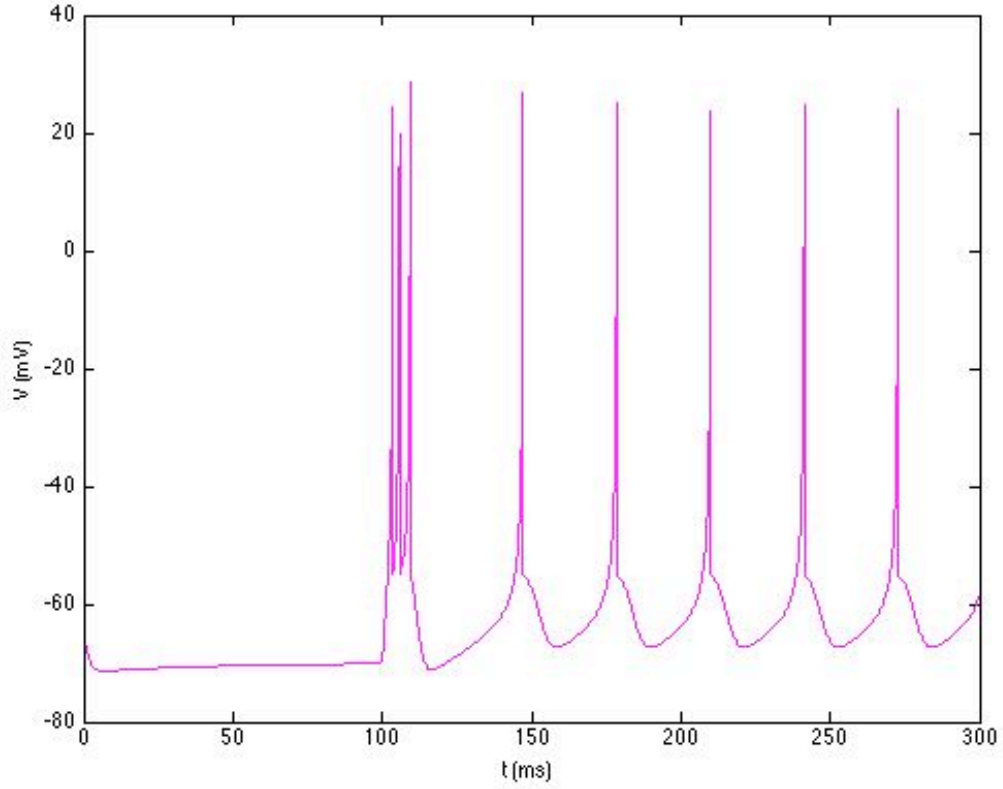
```
function [V,A]  = ModeleNeuronalIntensiteEchelon(a,b,c,d,i,tstep,Te,Ts)
N=Ts/Te;
A=zeros(1,N);
T=[0:Te:(N-1)*Te];
I=[];
Tstep=Te*floor(tstep/Te);
for k=1:Tstep/Te
    I(k)=0;
end
for k=Tstep/Te+1:N
    I(k)=i;
end
V=[-65];
U=[b*V(1)];
for k=1:N-1
    V(k+1)=V(k)+Te*(0.04*V(k)^2+5*V(k)+140-U(k)+I(k));
    U(k+1)=U(k)+Te*a*(b*V(k)-U(k));

    if V(k+1)>=30
        V(k+1)=c;
        U(k+1)=U(k+1)+d;
        A(k+1)=1;
    end

end

% plot(T,V,'magenta')
% hold on
% plot(T,U,'red')
```

Résultats pour les 3 modèles

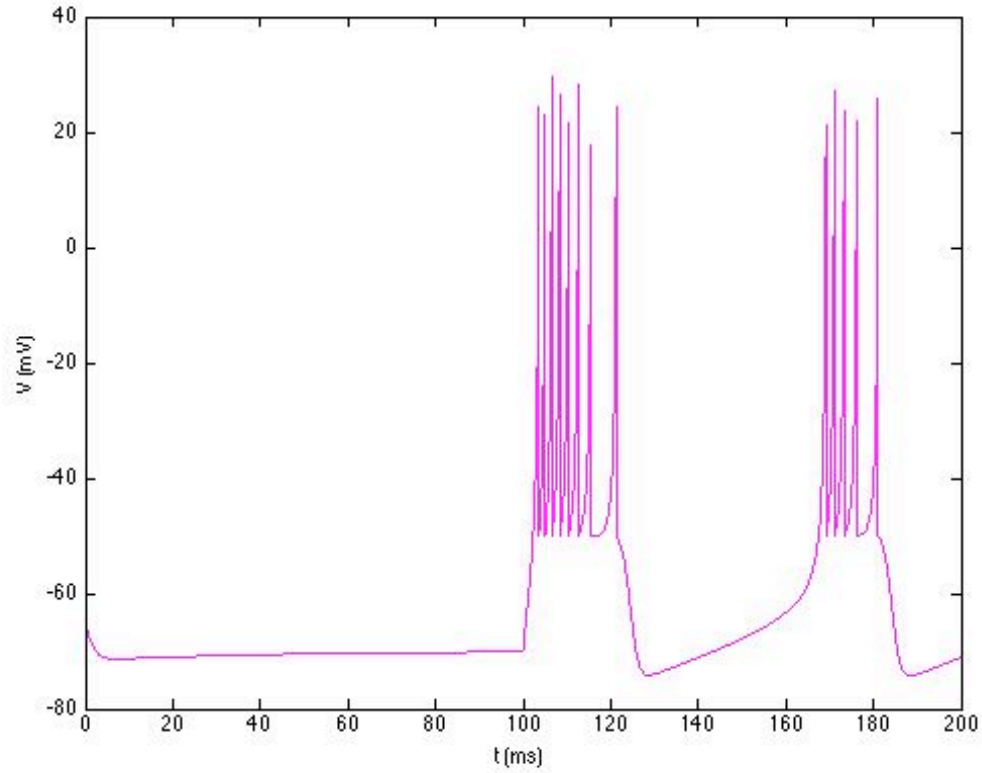
Ts variable selon les courbes
Te=0.05ms
temps de début de l'échelon de I=100ms



regular spiking



intrinsically bursting

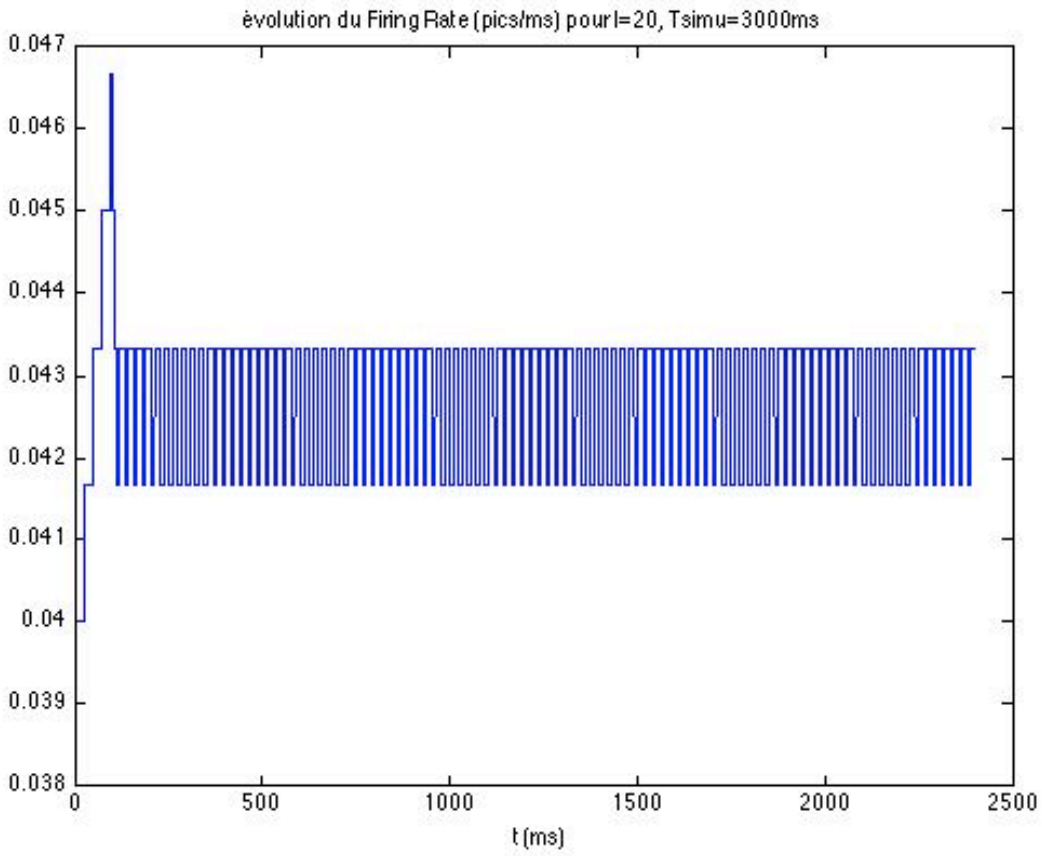


chattering: On obtient des séquences de burst

Programmes pour le calcul du Firing Rate

```
%Calcul du Firing Rate d'un neurone

function FR=FiringRatePourUnNeurone(A,Te,Ts,Tf)

%A: vecteur avec des 1 ‡ chaque spike
%Tf: temps de la fenÍtre; celle-ci se dÈplace et mesure le nombre de spikes
%qu'elle contient
N=Ts/Te;
T=[0:Te:(N-1)*Te];
L=length(T);
FR=zeros(1,L-floor(Tf/Te));
for n=1:L-floor(Tf/Te)
    q=0;
    for i=n:n+floor(Tf/Te)
        if (A(i)==1)
            q=q+1;
        end
    end
FR(n)=q/(Tf);
end;
```

courbe de FR=f(t) pour Tsimulation = 3000ms, Te=0.05ms, et fenêtre: 500ms, (résultat pour regular spiking)



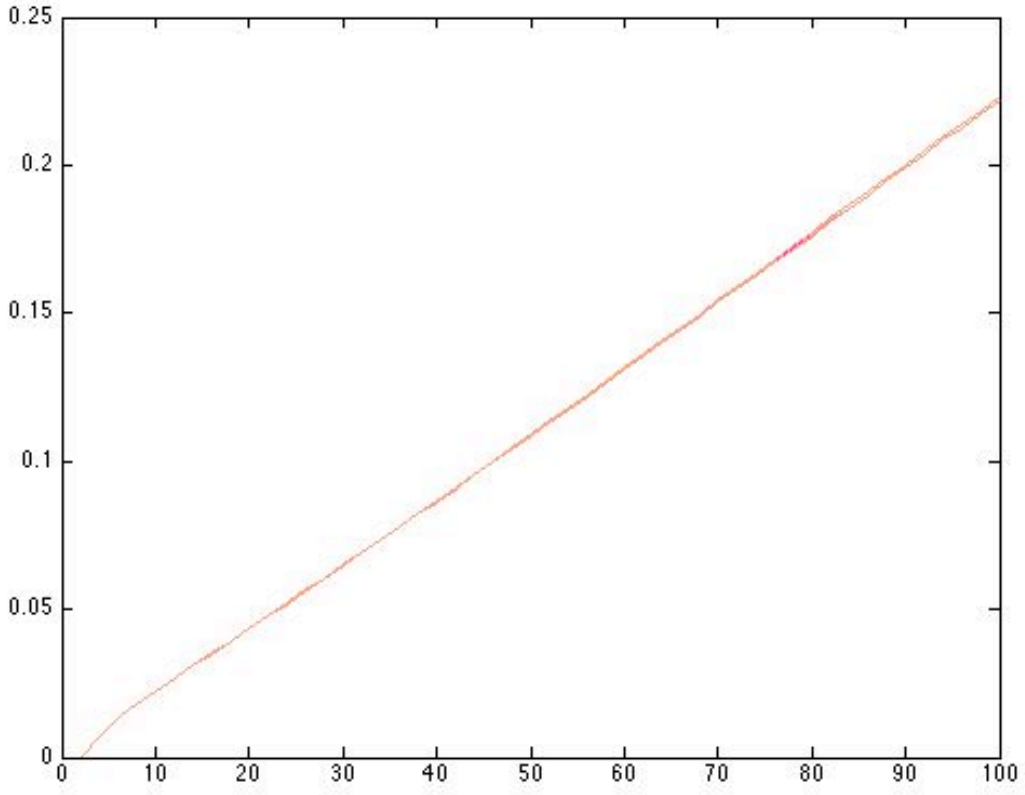
différentes courbes FR=f(I) selon la taille de la fenêtre

On prendra la moyenne des valeurs pour la valeur de Firing Rate finale:

```
%dessin de diffÈrentes courbes FR=f(I) selon la taille de la fenÍtre

B=[100:100:600];
InitRegularSpiking;

for k=1:length(B)

VI=(0:2:100);
VFR=eye(1,length(VI));

for i = 1 : length(VI)
    I=VI(i);

T=[0:Te:(N-1)*Te];
```

```
L=length(T);
Tf=B(k);
[V,A]= ModeleNeuronalIntensiteEchelon(a,b,c,d,I,100,Te,Ts);
FR=FiringRatePourUnNeurone(A,Te,Ts,Tf);
T1=[0:Te:((L-floor(Tf/Te))-1)*Te];
VFR(i)=sum(FR(floor(length(FR)/2):length(FR)))/(length(FR)-
floor(length(FR)/2));%moyennage des derniËres valeurs de FR pour annuler
les oscillations

end

plot(VI,VFR,'Color',[1 1-1/sqrt(k) 1/sqrt(k)])
hold on
end
```

Résultats pour le modèle de Regular Spiking en fonction de la taille de la fenêtre (51 points pour chaque courbe) cf analyse des réslutats



Remarque importante: La courbe trouvée dans le rapport préliminaire à cet endroit était fausse: nous avions alors posé le Firing Rate asymptotique comme sa valeur maximale, ce qui est faux.

ANNEXE 5: INTERCONNEXION:

création de la matrice d'interconnection: exemple pour l'aléatoire:

```
M=random('bino',1,1/2,6,6);
for i=1:6
    if M(i,i)==1
        M(i,i)=0;
```

```
    end
end
```

Programme de Calcul des V pour plusieurs neurones interconnectés:

On voit que l'on rajoute un input prenant en compte les sorties de tous les neurones reliés.

```
function [V,A]  = NeuronePourInterconnection(a,b,c,d,j,tstep,Te,Ts,M,R )

%Il s'agit ici d'un Èchelon
%j: intensitÈ de l'Èchelon
%tstep: temps de l'Èchelon
%Remarque: l'Èchelon dure jusqu'‡ la fin
%M:matrice d'interconnection
%R: rÈsistance d'interconnection



n=length(M); %nombre de neurones
N=Ts/Te; %nombre de points calculÈs
A=zeros(n,N);
T=[0:Te:(N-1)*Te];
I=[];
Tstep=Te*floor(tstep/Te); %construction de l'Èchelon I
for k=1:Tstep/Te
    I(k)=0;
end
for k=Tstep/Te+1:N
    I(k)=j;
end
v0=-65;             %crÈation des vecteurs initiaux
u0=b*v0;            %
V=zeros(n,N);       %
U=zeros(n,N);       %
V(:,1)=v0;          %
U(:,1)=u0;          %
% I1=zeros(n,N);%crÈation d'une matrice I d'impulsion
% I1(1,:)=I;

for k=1:N-1
    for i=1:n
        S=sum(V(:,k).*M(:,i)/R);
        V(i,k+1)=V(i,k)+Te*(0.04*V(i,k)^2+5*V(i,k)+140-U(i,k)+I(k)+S);
        U(i,k+1)=U(i,k)+Te*a*(b*V(i,k)-U(i,k));

        if V(i,k+1)>=30
            V(i,k+1)=c;
            U(i,k+1)=U(i,k+1)+d;
            A(i,k+1)=1;
        end
    end

end
for i=1:n
    subplot(2,3,i)
    plot(T,V(i,:),'magenta')
 end
% hold on
% plot(T,U,'red')
```

Calcul du Firing Rate en fonction du temps pour une population neuronale:

Il s'agit de la somme des Firing Rates individuels divisée par le nombre de neurones.

```
% Firing Rate pour une population de neurones

%A(i,j): matrice des vecteurs A (indicateurs de spikes) calculÈe dans
NeuronePourInterconnection

function FR=FiringRatePopulation(A,Te,Ts,Tf)
n=length(A(:,1));
N=Ts/Te;
FiringRatePartiel=zeros(n,N-floor(Tf/Te));

for k=1:n
    FiringRatePartiel(k,:)=FiringRatePourUnNeurone(A(k,:),Te,Ts,Tf); %
calcul de chaque Firing Rate
end

FR=zeros(1,N-floor(Tf/Te));
for t=1:N-floor(Tf/Te)
    FR(t)=sum(FiringRatePartiel(:,t))/n; %calcul du FR moyen de la
population
end

T1=[0:Te:((N-floor(Tf/Te))-1)*Te];%vecteur temps adaptÈ en taille
%plot(T1,FR) %tracÈ de la courbe
```

Tracé de la courbe FR=f(I):

Il s'agit de la courbe dont chaque point a comme ordonnée le Firing Rate asymptotique pour une intensité I en abscisse.

```
%attention aux arguents dans NeuronePourInterconnection: Il est dit qu'on
%utilise un regular spiking


function [B,FRfinal]=SigmoideInterconnection(B,R,M,tstep,Te,Ts,Tf)
l=length(B);
FRfinal=zeros(1,l);
for k=1:l
   [V,A]=NeuronePourInterconnection(0.02,0.2,-65,8,B(k),tstep,Te,Ts,M,R);
   FR=FiringRatePopulation(A,Te,Ts,Tf);
   FRfinal(k)=sum(FR(floor(length(FR)/2):length(FR)))/(length(FR)-
floor(length(FR)/2));%moyennage des derniËres valeurs de FR pour annuler
les oscillations
end
```