Université de Montréal

PHY 3075 – Modélisation numérique en physique

Projet 1 - Le modèle de Hodgkin-Huxley

par :

Patrice Béchard

20019173

6 février 2017

1 Introduction

Les neurones sont des cellules pour lesquelles le milieu intérieur est conducteur et la membrane est une couche isolante possédant des canaux ioniques, contrôlant le passage d'électricité entre l'intérieur et l'extérieur de la cellule. Le modèle de Hodgkin-Huxley, permet de décrire ce phénomène mathématiquement par quatre équations différentielles ordinaires (EDO) présentées ci-dessous :

$$\frac{d}{dt}\begin{pmatrix} V \\ n \\ m \\ h \end{pmatrix} = \begin{pmatrix} \frac{1}{c_m}(I_a - g_K n^4(V - V_K) - g_{Na} m^3 h(V - V_{Na}) - g_L(V - V_L)) \\ \alpha_n(V)(1-n) - \beta_n(V)n \\ \alpha_m(V)(1-m) - \beta_m(V)m \\ \alpha_h(V)(1-h) - \beta_h(V)h \end{pmatrix} \tag{1}$$

où V est le potentiel transmembranaire et n, m et h sont des fonctions barrières adimentionnelles. Les valeurs des différents potentiels de repos $V_{\{K,Na,L\}}$ et coefficients de conductance $g_{\{K,Na,L\}}$ sont présentées dans les notes de cours au tableau 1.2 [1]. Les fonctions $\alpha_x(V)$ et $\beta_x(V)$ sont aussi présentées dans les notes de cours. On remarque que toutes ces équations sont fortement reliées entre eux, signe que ce système est très non-linéaire.

Pour résoudre ce système d'EDO, la méthode de Runge-Kutta sera utilisée. Cette méthode utilise quatre estimés de pente pour itérer dans le temps et ainsi trouver le comportement des EDO. La méthode de Runge-Kutta peut cependant s'avérer imprécise lorsque le système est très fortement non-linéaire, comme c'est le cas ici. Pour remédier à ce problème, cette méthode sera combinée avec un pas adaptif, comparant deux solutions (une obtenue avec un pas entier, puis l'autre avec deux demi-pas), et adaptera la longueur du pas en fonction de la différence entre ces deux résultats.

Une vérification du code se fera tout d'abord avec des résultats connus dans la section 2 (voir notes de cours[1]), puis suivra une exploration numérique de divers phénomènes sur le modèle de Hodgkin-Huxley dans la section 3. Une version du code ainsi que figures sont disponibles en couleur ainsi qu'en pleine résolution à l'adresse suivante : `goo.gl/DxpB6g`.

2 Validation du code

Pour s'assurer de la validité du code, une reproduction des figures 1.14 à 1.17 des notes de cours a été faite avec programme qui sera utilisé pour faire l'exploration numérique subséquente. La figure 1 présente les valeurs à l'équilibre ainsi que le temps de relaxation des fonctions barrières n, m et h en fonction du potentiel transmembranaire, reproduisant la figure 1.14 des notes de cours.

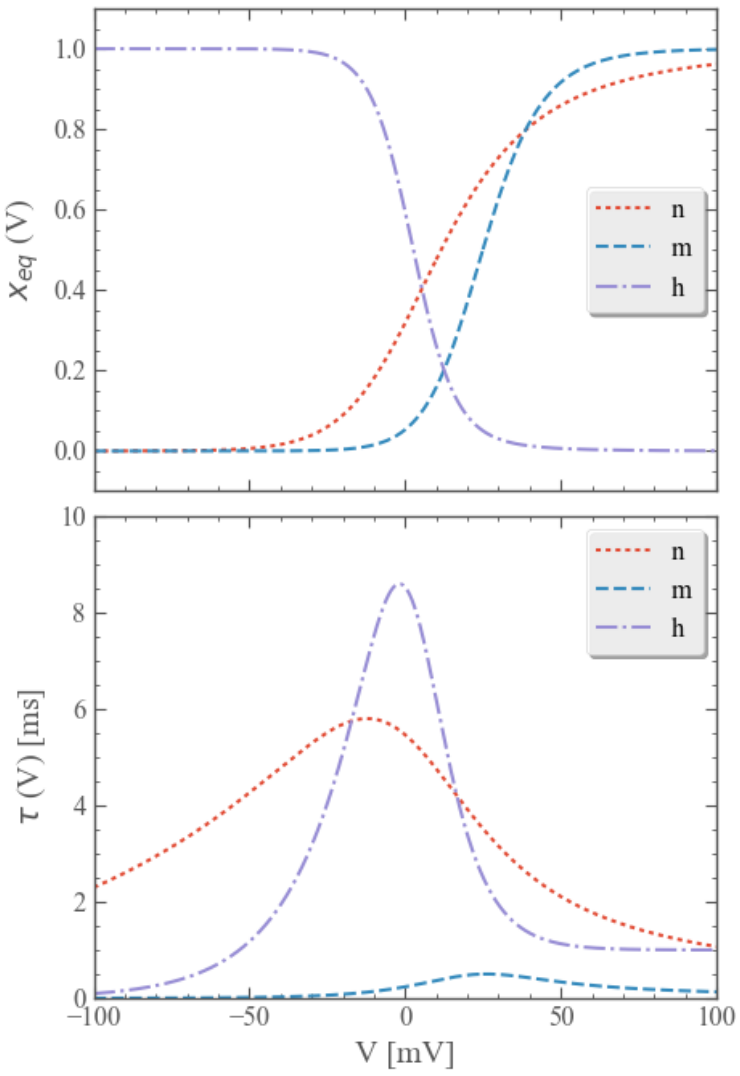


FIGURE 1 – Valeurs à l'équilibre et temps de relaxation des fonctions barrières n, m et h du modèle de Hodgkin-Huxley en fonction du potentiel transmembranaire V. Les valeurs à l'équilibre de n, m et h à V=0 sont utilisés dans le modèle de Hodgkin-Huxley comme condition initiale.

Chacune des courbes tracées semble exactement répliquer celles présentées à la figure 1.14 des notes de cours, signe que le programme développé fait bel et bien le travail voulu en ce qui a trait aux valeurs à l'équilibre et aux temps de relaxation des fonctions barrière.

La prochaine vérification à faire est au niveau de la propagation du modèle dans le temps. La figure 2, analogue à la figure 1.15 des notes de cours, présente la progression dans le temps des trois fonctions barrières ainsi que du potentiel transmembranaire V, où un courant de $7\mu A/cm^2$ est appliqué pendant une milliseconde au début de la simulation (représenté par la région grise). Au départ, le potentiel transmembranaire ainsi que les fonctions barrières se trouvent à leur valeur d'équilibre, selon la figure 1. Le potentiel croît linéairement lors de l'application du courant externe, et les fonctions barrières dévient de leur valeur à l'équilibre légèrement. Après que ce courant ait été appliqué, le potentiel transmembranaire continue à croître lentement et croît ensuite très rapidement après $5ms$. Ce pic est appelé *potentiel d'action*. Le potentiel transmembranaire décroît ensuite très rapidement jusqu'à un potentiel sous sa valeur d'équilibre, avant de revenir lentement vers celle-ci. La forme de la courbe empruntée par le potentiel transmembranaire, ainsi que celle empruntée par les différentes fonctions barrières est la même que celle obtenue à la figure 1.15 des notes de cours, signe que le programme utilisé est correct. Le point quadruple aux alentours de 5ms et 40mV est aussi répété dans les deux figures, nous indiquant que l'algorithme utilisé est équivalent pour les deux figures en terme de résultats.

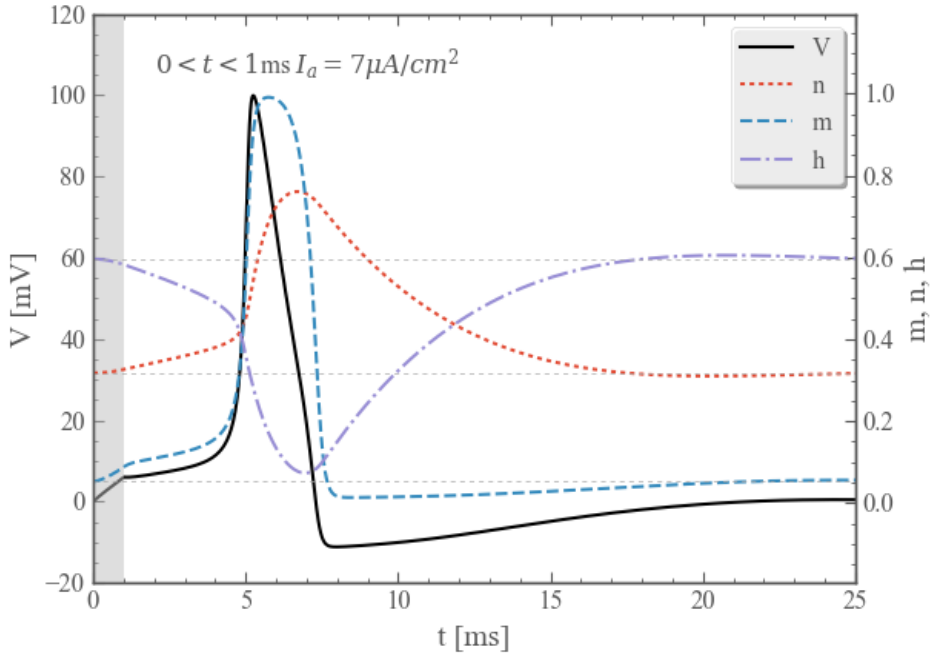


FIGURE 2 – Évolution du potentiel transmembranaire V et des fonctions barrières n, m et h en fonction du temps écoulé après avoir implanté un courant de $7\mu A/cm^2$ pendant 1ms. Le potentiel et les fonctions barrières se trouvent initialement à l'équilibre.

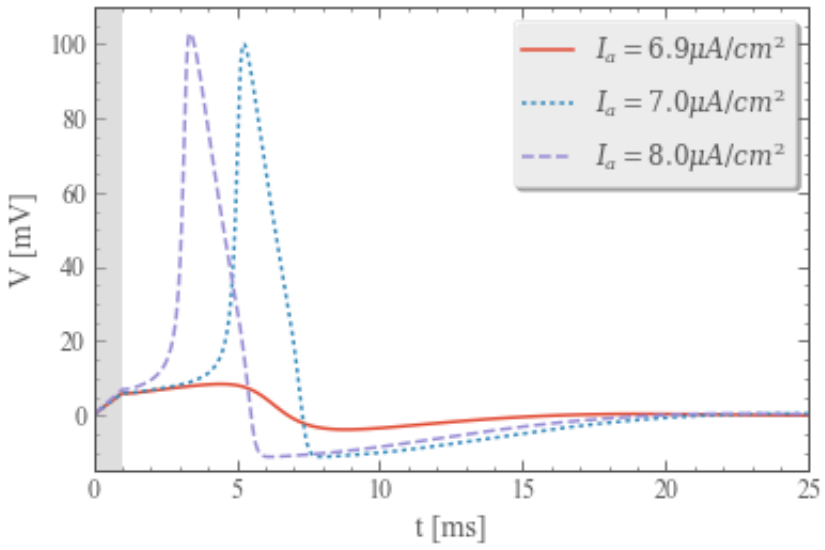


FIGURE 3 – Évolution du potentiel transmembranaire V en fonction du temps écoulé après avoir implanté pendant 1ms différentes valeurs de courant. Le potentiel initial est fixé à sa valeur d'équilibre, soit 0, et les fonctions barrières sont initialement établies à leur valeur d'équilibre correspondantes. Le seuil de courant à dépasser pour observer un potentiel d'action se trouve entre 6.9 et 7.0 $\mu A/cm^2$.

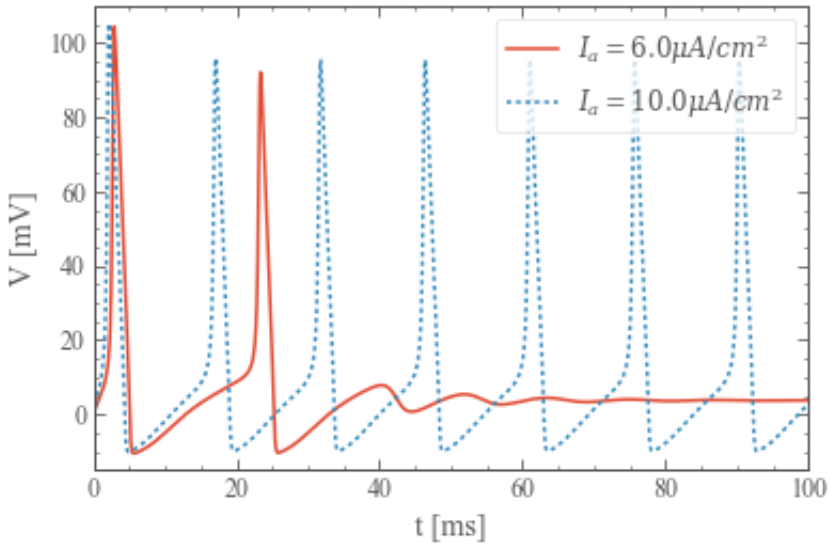


FIGURE 4 – Évolution du potentiel transmembranaire V en fonction du temps écoulé en implantant un courant constant tout au long de la simulation pour deux valeurs différentes de courant. Le potentiel initial est fixé à sa valeur d'équilibre, soit 0, et les fonctions barrières sont initialement établies à leur valeur d'équilibre correspondantes. Le signal est périodique dans le cas du plus grand courant, alors qu'il s'affaisse rapidement dans l'autre.

La figure 3 présente l'évolution du potentiel transmembranaire V après une impulsion initiale durant 1ms de courant de $8\mu A/cm^2$, $7\mu A/cm^2$ et $6.9\mu A/cm^2$, respectivement. Le potentiel d'action est déclenché plus tôt lorsque l'impulsion initiale est de plus grande norme ($8\mu A/cm^2$, dans notre cas), et il existe un seuil minimal en dessous duquel il est impossible de déclencher ce potentiel d'action. Cette caractéristique est observable en comparant les courbes à $7\mu A/cm^2$ et $6.9\mu A/cm^2$ deux valeurs de courant très rapprochées, mais qui influencent le système de façon très différentes l'une de l'autre, puisque le courant de $6.9\mu A/cm^2$ ne déclenche jamais de croissance rapide du potentiel transmembranaire. Cette figure est encore une fois une réplique presque parfaite de la figure 1.16 des notes de cours, signe que le programme utilisé fonctionne de la façon voulue.

La figure 4 est la dernière figure étudiée pour faire la vérification du code. Celle-ci présente encore une fois l'évolution du potentiel transmembranaire dans le temps, mais cette fois-ci sous l'action d'un courant constant tout au long de la simulation. Deux valeurs de courant ont été étudiées, soit à $6.0\mu A/cm^2$ et à $10.0\mu A/cm^2$. La principale différence entre les deux courbes tracées est que celle pour un courant de $10.0\mu A/cm^2$ produit un signal périodique se répétant indéfiniment avec la même amplitude ainsi que la même fréquence, tandis que l'autre courbe n'est plus en mesure de produire des potentiels d'actions après le deuxième pic, et ce, malgré qu'il y ait un courant constant pour toute la simulation. Une analyse plus détaillée du phénomène sera présentée à la section 3. La figure présentée est identique à la figure 1.17 des notes de cours, terminant une fois pour toute la validation de notre code, qui semble produire les résultats

voulus pour les différents aspects du modèle de Hodgkin-Huxley étudiés.

3 Résultats et discussion

Une analyse détaillée du potentiel transmembranaire en fonction de l'intensité du courant appliqué en continu a été menée pour des intensités de courant allant de 0 à 50 $\mu A/cm^2$. Le but était d'observer une relation entre le courant appliqué et la fréquence des pics répétés ainsi que leur amplitude. Il a été aussi possible de découvrir plusieurs régimes de courant pour lesquels la réponse du potentiel transmembranaire était très différente. Ceux-ci sont présentés au tableau 1.

Plage de courant $[\mu A/cm^2]$ (± 0.0005)	Nombre de pics observés
$[0, 2.2410]$	0
$[2.2410, 5.9726]$	1
$[5.9726, 6.1716]$	2
$[6.1716, 6.2171]$	3
$[6.2171, 6.2355]$	4
$[6.2355, 6.2640]$	Nombre croissant
$[6.2640, 50.0]$	Tous

TABLE 1 – Nombre de maxima observés en fonction de l'intensité du courant continu appliqué.

Il est observé qu'avant un courant seuil, soit $2.2410 \pm 0.0005 \mu A/cm^2$, aucun potentiel d'action n'est déclenché par le courant continu. Pour les valeurs de courant entre 2.2410 et 6.2640 $\mu A/cm^2$, le nombre de pics observés est fini et dépend de l'intensité du courant. Après le courant seuil de $6.2640 \pm 0.0005 muA/cm^2$, le signal devient périodique indéfiniment et les potentiels d'action sont toujours entraînés. L'allure de l'amplitude moyenne ainsi que la fréquence à laquelle les potentiels d'action surviennent en fonction du courant sont présentés à la figure 5. Le graphique ne montre que la portion d'intérêt, juste après le courant seuil.

La distribution de l'amplitude des maxima semble atteindre un maximum aux alentours de $8 \mu A/cm^2$, après quoi elle diminue en augmentant le courant. Cette tendance semble être linéaire à partir de $10 \mu A/cm^2$ et continue pour toutes les valeurs mesurées, soit jusqu'à $50 \mu A/cm^2$. La tendance empruntée par la fréquence est aussi linéaire à partir de $10 \mu A/cm^2$. Le fait que la fréquence grandit en fonction du courant appliqué est bien visible sur a figure 4, où les pics du courant plus grand devancent ceux du courant plus petit dès le début. Le système aura donc tendance à posséder des pulses de potentiels d'action de plus en plus rapprochés, mais de plus en plus faibles en augmentant la norme du courant appliqué.

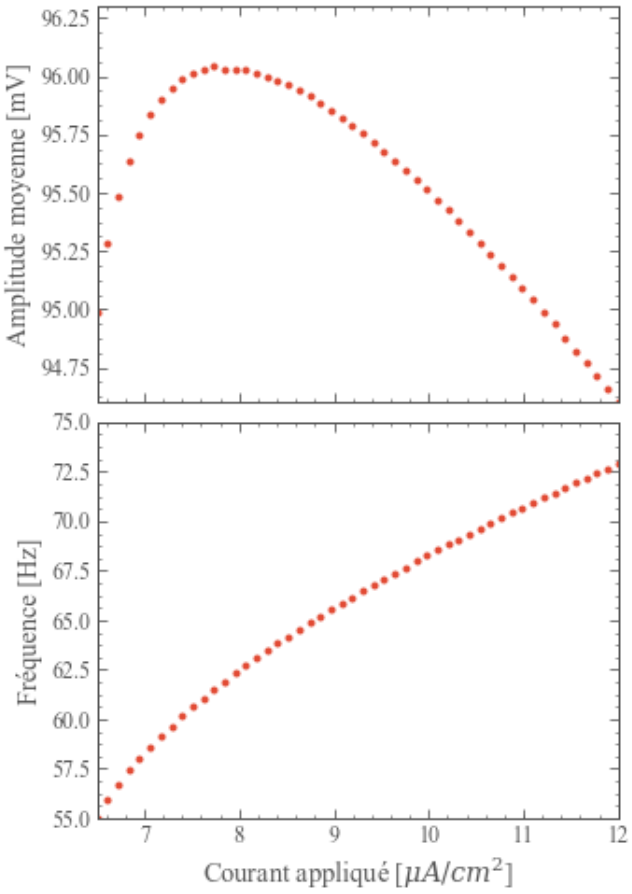
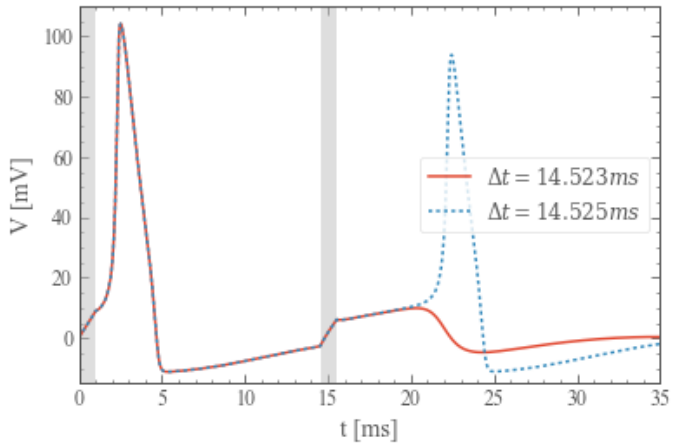


FIGURE 5 – Amplitude moyenne et fréquence des potentiels d'action engendrés par un courant continu en fonction de celui-ci.

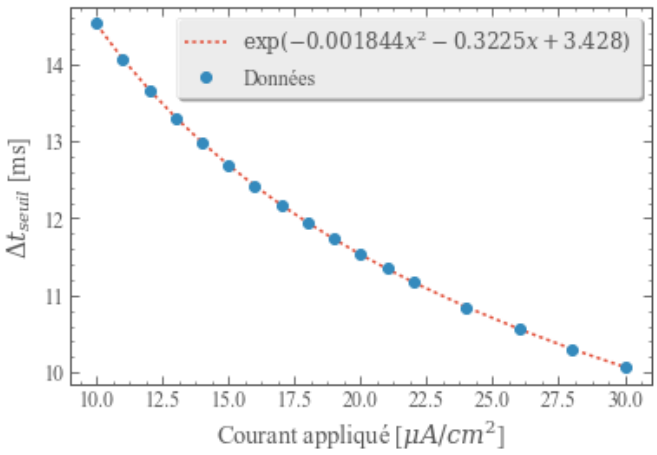
L'effet de deux pulses de 1ms sur le système a aussi été étudié pour trouver après quelle intervalle de temps ces deux pulses peuvent engendrer deux potentiels d'action. En utilisant deux pulses de $10\mu A/cm^2$, il a été observé que le début de chacun des pulses devait être séparé de $14.5240 \pm 0.0010ms$ avant de pouvoir engendrer deux potentiels d'action. La figure 6a présente l'impact sur le système de deux pulses séparés de 14.523ms et 14.525ms du pulse original. Le courant déclenché seulement 0.002ms plus tard fait apparaître un second potentiel d'action.

D'autres tests ont été effectués avec des grandeurs de courant de 11 à 30 $\mu A/cm^2$ pour le second pulse. L'écart temporel minimal pour chacune de ces grandeurs de courant a été mis en graphique à la figure 6b et une régression exponentielle a été tracée en fonction des données obtenues. L'équation de cette régression exponentielle est présentée dans la figure. En théorie, en suivant cette courbe, lorsque la norme du courant appliqué tendra vers l'infini, un potentiel d'action pourra être déclenché de plus en plus tôt en tendant vers 0. Cependant, en pratique, il est impossible que le potentiel d'action devance le premier étant créé, ce qui veut dire qu'un Δt_{seuil} d'au moins 5ms doit exister. De plus, lorsque le courant est très

grand, celui-ci semble engendrer à lui-seul le pic de potentiel transmembranaire, signe que le potentiel d'action observé n'en est pas un.



(a) Évolution du potentiel transmembranaire en fonction du temps lors d'une implantation de deux pulses de courant de $10\mu A/cm^2$, le premier était à t=0ms, et le second était à 14.523ms et à 14.525 pour les deux cas différents étudiés. Le potentiel initial est fixé à 0 et les fonctions barrières sont initialement à leurs valeurs d'équilibre correspondantes.

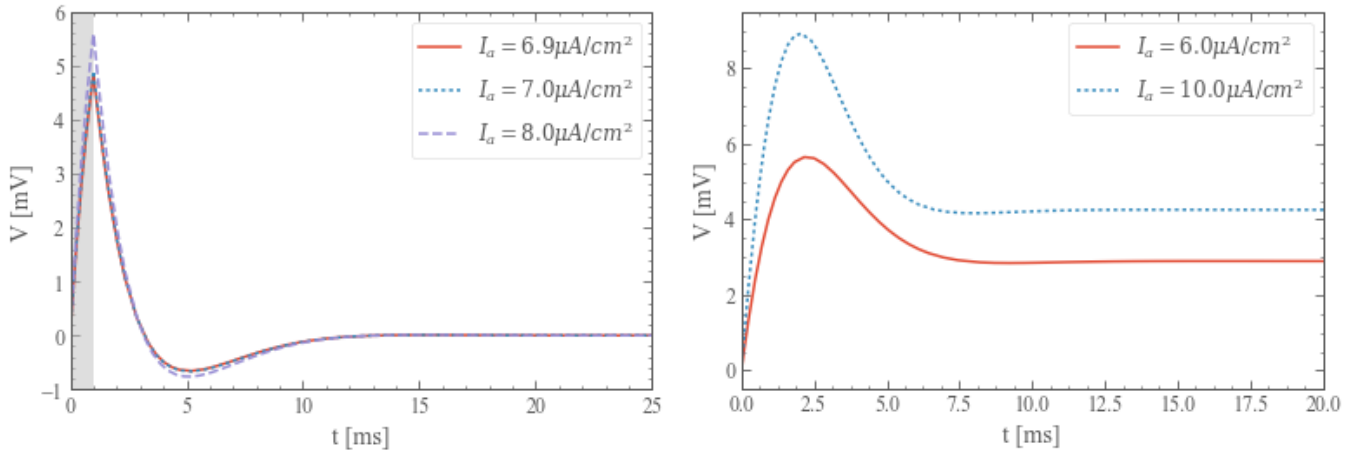


(b) Δt_{seuil} minimal en fonction de la norme du courant implanté pendant le second pulse. La courbe décrite par les données propose une solution exponentielle décroissante, signe qu'il n'y a pas de temps minimal à attendre avant de déclencher un second potentiel d'action.

FIGURE 6 – Réponse du potentiel transmembranaire à une succession de deux pulses pour lesquelles l'intensité du second pulse a varié en fonction des simulations.

Le dernier aspect exploré par-rapport au modèle de Hodgkin-Huxley est le fait que le temps caractéristique de la fonction barrière m est beaucoup plus court que le temps caractéristique des deux autres fonctions barrières (n et h), comme on peut le voir sur la figure 1. Il est possible de voir si la fonction barrière m a une influence sur le modèle de Hodglin-Huxley en l'enlevant tout simplement du système d'EDO présenté à l'équation 1 et en simulant le modèle numériquement. En comparant le résultat obtenu avec les résultats obtenus à la section 2, il est possible d'observer les changements qu'engendrent l'omission l'EDO reliée à cette fonction barrière, présentés à la figure 7.

Le premier cas étudié est lorsqu'une impulsion de courant de 1ms est appliquée au système au début de la simulation. Tout comme pour la figure 3, des courants de 6.9, 7 et 8 $\mu A/cm^2$ ont été appliqués sur le système initialement à l'équilibre. La figure 7a présente le cas mentionné. Dans cette situation, aucun potentiel d'action n'est créé suite au pulse de courant, et ce, peu importe l'intensité du courant, de sorte que les trois courbes simulées se superposent presque parfaitement. De plus, puisqu'aucun potentiel d'action n'est créé, l'amplitude du potentiel transmembranaire reste très faible et le retour à l'équilibre



(a) Évolution du potentiel transmembranaire après l'application d'un courant pendant 1ms au début de la simulation. La fonction barrière m du modèle de Hudgkin-Huxley est gardée constante pour cette simulation. Le potentiel transmembranaire est initialement fixé à 0 et les fonctions barrières n et h sont initialement fixées à leur valeur d'équilibre respective.

(b) Évolution du potentiel transmembranaire lors d'une application continue d'un courant électrique lors de la simulation. La fonction barrière m du modèle de Hudgkin-Huxley est gardée constante pour cette simulation. Le potentiel transmembranaire est initialement fixé à 0 et les fonctions barrières n et h sont initialement fixées à leur valeur d'équilibre respective.

FIGURE 7 – Réponse du potentiel transmembranaire lorsque la fonction barrière m est gardée constante.

arrive vers 10 ms, comparativement au cas où l'EDO dépendant de la fonction barrière m n'était pas omise, alors que le retour a l'équilibre arrivait vers 20ms.

L'autre cas étudié est lorsque le courant appliqué est constant tout au long de la simulation numérique. Cette situation était présentée à la figure 4 dans le cas où l'EDO reliée à la fonction barrière m n'était pas négligée, et ce, pour des courants de 6 et 10 $\mu A/cm^2$. La même situation a été simulée sans l'EDO modulant la fonction barrière m. Le résultat de la simulation est présenté à la figure 7b. Plusieurs caractéristiques de la simulation originale ne sont plus présentes dans la simulation sans variation de la fonction barrière. Tout d'abord, aucun potentiel d'action n'est déclenché peu importe la grandeur du courant. Le caractère périodique du signal est lui aussi absent de cette simulation, ne possédant qu'un maximum au début sans avoir d'autre pics par la suite. Finalement, la valeur du potentiel transmembranaire s'équilibre à une valeur non-nulle pour le reste de la simulation, phénomène n'ayant pas été observé auparavant dans aucune situation.

Le fait d'enlever l'EDO responsable de la modulation de la fonction barrière m change donc complètement le modèle de Hodgkin-Huxley étudié plus tôt, puisque suite à son omission, aucune caractéristique présente dans les simulations précédentes n'est observée. Il n'est pas possible non plus de négliger la modulation des autres fonctions barrières, puisque leur omission mène à des conclusions semblables. Cela dit,

puisque le temps caractéristique des deux autres fonctions barrières était plus important, cette conclusion pouvait facilement être prédite.

4 Programme

```
# -*- coding: utf-8 -*-
"""
Title : Hudgkin-Huxley Model resolution with Runge-Kutta Method (ODE)

Author : Patrice Bechard

Date : February 5th, 2017

Program that computes the Hudgkin-Huxley model for the transmission
of electric signal in the brain locally.

We use the 4th degree Runge-Kutta method with adaptive step to solve
ordinary differential equations.

Electric potential is calculated in mV, time in ms, capacitance in Î¼F cm^-2,
Electric current in Î¼A cm^-2.
"""
############## MODULES

import numpy as np                          #for arrays and mathematical functions
import matplotlib.pyplot as plt             #for plots
import sys                                  #for debugging purposes
import time

plt.style.use('patrice')                    #custom graphics package

############## FONCTIONS

def find_equilibrium():
    """Function that finds the values of the N,H,M functions
    and characteristic time for a given V.

    Using equations 1.65 and 1.66 to calculate xN, tauM, etc.
    """
    V=np.linspace(-100,100,10001)   #list of values for potential from -100 to 100
    for i in range(len(V)):

        xN=alphaN(V[i])/(alphaN(V[i])+betaN(V[i]))
        xM=alphaM(V[i])/(alphaM(V[i])+betaM(V[i]))
        xH=alphaH(V[i])/(alphaH(V[i])+betaH(V[i]))
        tauN=1/(alphaN(V[i])+betaN(V[i]))
        tauM=1/(alphaM(V[i])+betaM(V[i]))
        tauH=1/(alphaH(V[i])+betaH(V[i]))
```

```
        if i==0:                            #set the first part of array
            x=np.array([[xN,xM,xH]])
            tau=np.array([[tauN,tauM,tauH]])
        else:                               #dynamically add values to array
            x=np.concatenate((x,[[xN,xM,xH]]),axis=0)
            tau=np.concatenate((tau,[[tauN,tauM,tauH]]),axis=0)

    return V,x,tau

def adaptive_step(nMax,tFin):
    """Function that performs an evolution of the H-H model in time
    for one situation using an adaptive step.

    Calls the Runke-Kutta function to solve the ODEs
    """
    #initiate default step and iterator
    eps=1.e-5                   #tolerance for RK adaptive step
    h=0.1                       #initial step
    nn=0                        #iterator for number of steps

    #initiate arrays to store time and position with equilibrium values
    t=np.array([0])
    u=np.array([[0.,x[index,0],x[index,1],x[index,2]]])
    dim=len(u[0])               #number of parameters to analyse

    while t[-1]<tFin and nn<nMax-1:
        u1=rungeKutta(t[-1],u[-1],h)        #full step
        u2a=rungeKutta(t[-1],u[-1],h/2.)    #first half-step for adaptive step method
        u2=rungeKutta(t[-1],u2a,h/2.)       #second half-step
        delta=max(abs(u2[0]-u1[0]),abs(u2[1]-u1[1]),abs(u2[2]-u1[2]),abs(u2[3]-u1[3]))#eqn 1.42
        if delta>eps:
            h/=1.5
        else:
            nn+=1
            t=np.append(t,t[-1]+h)
            u=np.append(u,u2)               #add new values to array
            u=u.reshape((nn+1,dim))         #and reshape it as a nn+1 by "dim" matrix
            if delta<=eps/2.:
                h*=1.5

    localmax=[]                             #empty array where we store max
    period=[]                               #same for when the max happen
    for i in range(1,len(t)-2):             #in case of IndexError
        if (u[i-1][0]<u[i][0]>=u[i+1][0]):  #else value is not a max
                localmax.append(u[i,0])
                period.append(t[i])
    localmax=localmax[1:]                   #first max is not good
    period=period[1:]
    meanmax=np.mean(localmax)               #value of mean max
```

```
    freq=[(1000/(period[i+1]-period[i])) for i in range(len(period)-1)] #and frequency
    meanfreq=np.mean(freq)                  #mean value of frequency

    return [meanmax,meanfreq]

def slope(t0,uu):
    """Function that contains right side of the differential equations to solve """

    #constants for the problem (see table 1.2)
    VK,VNA,VL = -12., 115., 10.6  #resting potentials (constants)
    GK,GNA,GL = 36., 120., 0.3    #conductance coefficients (constants)
    CM=1                          #conductance membranaire (constant)

    global ione, itwo, second     #variables from main() can be used here

    # Current applied to the membrane
    if second<0:                  #current is constant fot the whole time
        courantA=ione
    elif second==0:               #current applied for 1ms and never again
        courantA=ione if t0<1 else 0
    else:                         #two pulses happening at t=0 and t=second
        if t0<=1:
            courantA=ione
        elif second<=t0<=second+1:
            courantA=itwo
        else:
            courantA=0

    ###slopes (eq. 1.68)
    gV=(1./CM)*(courantA-GK*(uu[1]**4)*(uu[0]-VK)-GNA*(uu[2]**3)*uu[3]*(uu[0]-VNA)-GL*(uu[0]-VL))
    gn=alphaN(uu[0])*(1-uu[1])-betaN(uu[0])*uu[1]
    gm=0                          #comment following line for 6)
    gm=alphaM(uu[0])*(1-uu[2])-betaM(uu[0])*uu[2]
    gh=alphaH(uu[0])*(1-uu[3])-betaH(uu[0])*uu[3]

    return np.array([gV,gn,gm,gh])

def rungeKutta(t0,uu,h):
    """Function that performs the Runge-Kutta method for solving ODEs"""

    g1=slope(t0,uu)                 #slope1
    g2=slope(t0+h/2,uu+(h/2)*g1)    #slope2
    g3=slope(t0+h/2,uu+(h/2)*g2)    #slope3
    g4=slope(t0+h,uu+h*g3)          #slope4
    return uu+(h/6)*(g1+2*g2+2*g3+g4)

# Alpha and Beta functions (1.62, 1.63, 1.64)

def alphaH(V): return 0.07*np.exp(-V/20.)
def alphaM(V): return (2.5-0.1*V)/(np.exp(2.5-0.1*V)-1.)
```

```
def alphaN(V): return (0.1-0.01*V)/(np.exp(1.-0.1*V)-1.)
def betaH(V):  return 1./(np.exp(3.-0.1*V)+1.)
def betaM(V):  return 4.*np.exp(-V/18.)
def betaN(V):  return 0.125*np.exp(-V/80.)

############### MAIN

t_init=time.time()          #we start the time when the program begins

V,x,tau=find_equilibrium()          #we find the equilibrium values
index=int(np.where(V==0.)[0])       #index where is V=0 for
"""
#Two graphs with same x axis
f, axarr = plt.subplots(2, sharex=True,figsize=(5,8))

#first graph is for x_eq
axarr[0].plot(V, x[:,0],':',label='n')
axarr[0].plot(V, x[:,1],'--',label='m')
axarr[0].plot(V, x[:,2],'-.',label='h')
axarr[0].set_ylabel(r"$x_{eq}$ (V)")
axarr[0].axis([-100,100,-0.1,1.1])
axarr[0].legend(loc=5,fancybox=True,shadow=True)

#second graph is for tau
axarr[1].plot(V,tau[:,0],':',label='n')
axarr[1].plot(V,tau[:,1],'--',label='m')
axarr[1].plot(V,tau[:,2],'-.',label='h')
axarr[1].set_ylabel(r"$\tau$ (V) [ms]")
axarr[1].set_xlabel("V [mV]")
axarr[1].axis([-100,100,0,10])
axarr[1].legend(fancybox=True,shadow=True)

f.subplots_adjust(hspace=0.05)          #reduce space in between
f.savefig("verif_equilibre.png")        #save figure
"""

# Initialize loop contraints

nMax=10000                           #max number of steps
tFin=100.                            #max time
courantA=np.array([6.0,10])
itwo=10                             #intensity of second pulse (#3)
#time of second pulse (<1 if current is constant, 0 if only one pulse)
secondpulse=np.array([-1])
frequency=[]                        #empty array for mean frequency
maximum=[]                          #and mean max for one simulation

for ione in courantA:               #comment if we test on second pulses
    second=secondpulse[0]
#for second in secondpulse:         #comment if we test on multiple currents
```

```
#    ione=courantA[0]

    results=adaptive_step(nMax,tFin)

    maximum.append(results[0])
    frequency.append(results[1])

"""
rangeI=np.linspace(10,20,11)          #values found for min time vs current
rangeI=np.append(rangeI,np.array([21,22,24,26,28,30]))
data=np.array([14.524,14.06,13.66,13.304,12.981,12.689,12.420,12.173,\
11.944,11.731,11.531,11.345,11.169,10.848,10.560,10.303,10.069])
plt.loglog(rangeI,data,'*')

logx = np.log(rangeI)                    #for the fit
logy = np.log(data)
coeffs = np.polyfit(logx,logy,deg=2)
poly = np.poly1d(coeffs)
print(poly)

x=np.linspace(0,10000,10001)
yfit = lambda rangeI: np.exp(poly(np.log(rangeI)))
plt.loglog(rangeI,yfit(rangeI))
plt.show()
plt.plot(rangeI,yfit(rangeI),':',label=r'$\exp{(-0.001844x^2-0.3225x+3.428)}$')
plt.plot(rangeI,data,'o',label='DonnÃ©es')

plt.xlabel(r"Courant appliquÃ© [$\mu A/cm^2$]")
plt.ylabel(r"$\Delta t_{seuil}$ [ms]")
plt.legend(fancybox=True,shadow=True)
plt.show()
"""

"""
#graph with different currents plotted
plt.axis([0,tFin,-20,120])
#plt.axis([0,tFin,-25,150])
plt.fill_between([0,1],-20,120,facecolor="0.75",alpha=0.5)
plt.plot([0,tFin],[0,0],':',c="0.66",linewidth=0.5)
plt.legend(fancybox=True,shadow=True)
plt.xlabel("V [mV]")
plt.ylabel("t [ms]")
plt.savefig("verif_diff_courant.png")
#plt.savefig("verif_progr_courant_cst.png")
"""

"""
#graph with 2 different y axis
f, ax1=plt.subplots()
ax2=ax1.twinx()
```

```
#plot data
ax1.plot(t,u[:,0],'k')
ax2.plot([0],[0],'k',label='V') #to solve the legend problem
ax2.plot(t,u[:,1],':',label='n')
ax2.plot(t,u[:,2],'--',label='m')
ax2.plot(t,u[:,3],'-.',label='h')
#grey space at beginning
ax2.fill_between([0,1],-0.2,1.2,facecolor="0.75",alpha=0.5)

#plot horizontal lines as in fig 1.15
for i in [1,2,3]:
    ax2.plot([0,25],[u[0,i],u[0,i]],':',c="0.66",linewidth=0.5)

#set plot parameters
ax1.set_xlabel("t [ms]")
ax1.set_ylabel("V [mV]")
ax2.set_ylabel("m, n, h")
ax1.axis([0,25,-20,120])
ax2.axis([0,25,-0.195,1.195])
ax2.legend(fancybox=True,shadow=True)
ax2.text(2,1.05,r"$0<t<1$ms $I_a=7\mu A/cm^2$")

f.savefig("verif_V_and_x.png")
"""
print("ELAPSED : ",time.time()-t_init)
```

Références

[1] Charbonneau, P., *PHY3075 - Modélisation numérique en physique, Chapitre 1 : Équations différentielles ordinaires*, Université de Montréal, Montréal, 2017, 30p.