

Sorbonne Université  
Faculté des sciences et ingénierie  
Rapport de stage de troisième année licence

---

## Modélisation d'un neurone artificiel sous Matlab

---

**Réalisé par :**  
Yousri Aboudaoud, Marc Zhan  
**Orientation :**  
Electronique Energie Automatique CMI ÉLECTRONIQUE  
**Entité d'accueil :**  
Laboratoire de génie électrique de Paris-GEEPS  
**Responsables du stage :**  
M. Aziz BenLarbi-DELAI  
M. Pietro Ferreira

2021/2022

## Remerciements

Nous souhaitons remercier Monsieur Aziz Ben Larbi et Monsieur Pietro Marris qui nous ont accompagné durant le stage avec passion et panache ; nous les remercions pour leurs précieux conseils.

<b>Résumé</b>	<b>4</b>
<b>Summary</b>	<b>5</b>
<b>Introduction</b>	<b>6</b>
<b>Présentation de l'entité d'accueil</b>	<b>7</b>
2.1 Le laboratoire Geeps	7
2.2 Le pôle électronique	8
<b>Le Neurone biologique et sa modélisation</b>	<b>9</b>
3.1 Fonctionnement du neurone biologique	9
3.1.1 Le neurone	9
3.1.2 L'influx nerveux	10
3.1.2 Codage en fréquence	11
3.1.3 Le neurone en circuit électronique	12
3.2 Modèles de neurones	13
3.2.1 La plausibilité biologique	13
3.2.2 Modèle Hodgkin-Huxley	13
3.2.3 Modèle LIF (Leaky Integrate Fire)	15
<b>La modélisation sous Matlab</b>	<b>17</b>
4.1 Simulation LIF	17
4.2 Interpolation des données de Cadence sous Matlab	20
4.3 Estimation de l'erreur	23
4.4 Comparaison avec les simulations sous cadence	25
<b>5. Conclusion</b>	<b>28</b>
<b>6. Compétences acquises</b>	<b>29</b>
<b>Références</b>	<b>30</b>

## Résumé

Le monde tel que nous le connaissons aujourd’hui n’a jamais été aussi connecté. Cela induit une très grosse consommation d’énergie électrique. Une solution s’impose, dans ce stage ce sera une solution neuro inspirée qui sera modélisée. Notre cerveau est un dispositif capable d’analyser des scènes complexes en un temps record tout en consommant une faible puissance. Le neurone, composant élémentaire du cerveau, est un véritable dispositif électronique, il reçoit, traite et transmet de l’information (sous forme d’oscillation que l’on nomme potentiel d’actions) suite à une excitation ; il utilise les courants ioniques afin d’effectuer les fonctions précédemment énumérées. Aussi, le secret de sa faible consommation réside dans sa capacité à coder en fréquence une information qui lui provient sous forme de courant.

L’on peut donc reproduire ce fonctionnement par le biais de circuits électroniques en employant divers modèles, ils vont du moins complexe mais le moins bio-inspiré jusqu’au plus complexe (en termes d’équations différentielles à coefficients couplés) mais le plus bio-inspiré comme le modèle de Hodgkin-Huxley. Un autre modèle, LIF (Leaky integrate and fire), offrant un bon compromis entre la bio-inspiration et la complexité du modèle ; c’est le modèle qui nous a été demandé de simuler.

Pour réaliser la simulation, il faut passer par des logiciels de conception de circuit électronique comme Cadence, ce dernier étant précis mais chronophage afin de simuler un neurone artificiel. L’objet de stage est de proposer une alternative plus rapide à savoir Matlab.

La modélisation sous Matlab consiste à mettre en point un algorithme qui simule approximativement les résultats fournis par Cadence. Puis, il nous revient à prendre en compte le courant d’excitation  $I_{ex}$  en interpolant les données données fournies par Cadence (données qui nécessitent toute une durée d’une journée pour être produites), puis nous verrons l’effet de lex sur la fréquence de nos potentiels d’action (Spike). Après avoir réussi cela nous comparant nos résultats avec ceux de Cadence en utilisant une figure de mérite (l’erreur moyenne quadratique).

## Summary

The world as we know it today has never been so connected. This induces a very high consumption of electrical energy. A solution is needed, in this internship it will be a neuro-inspired solution that will be modeled. Our brain is a device capable of analyzing complex scenes in record time while consuming low power. The neuron, an elementary component of the brain, is a real electronic device, it receives, processes and transmits information (in the form of oscillation called action potential) following an excitation; it uses ionic currents in order to carry out the functions previously listed. Also, the secret of its low consumption lies in its ability to code in frequency an information which comes to it in the form of a current.

We can therefore reproduce this functioning through electronic circuits using various models, from the least complex but least bio-inspired to the most complex (in terms of differential equations with coupled coefficients) but most bio-inspired as the Hodgkin-Huxley model. Another model, LIF (Leaky integrate and fire), offers a good compromise between bio-inspiration and model complexity; this is the model we were asked to simulate.

To realize the simulation, it is necessary to use electronic circuit design software such as Cadence, which is accurate but time consuming to simulate an artificial neuron. The purpose of this internship is to propose a faster alternative, namely Matlab.

The modeling under Matlab consists in developing an algorithm which simulates approximately the results provided by Cadence. Then, we have to take into account the excitation current  $I_{ex}$  by interpolating the data provided by Cadence (data that require a whole day to be produced), then we will see the effect of  $I_{ex}$  on the frequency of our action potentials (Spike). After having done this, we will compare our results with those of Cadence using a figure of merit (root mean square error).

# 1. Introduction

Le monde tel que nous le connaissons n'a jamais été aussi connecté qu'aujourd'hui. En effet, les IoT (Internet of Things) désigne un nombre croissant d'objets connectés à Internet permettant ainsi une communication entre nos biens dits physiques et leurs existences numériques. Ces formes de connexions permettent de rassembler de nouvelles masses de données sur le réseau et donc, de nouvelles connaissances et formes de savoirs [1].

La réception de consignes et l'envoi d'information qui s'effectue entre l'opérateur et l'objet connecté s'effectue via le CLOUD computing (ou l'informatique en nuage, est un ensemble de matériels, de raccordements réseau et de logiciels) [2].

L'objet connecté envoie des informations relatives à l'environnement dans lequel il est déployé (Températures, humidité...) au CLOUD (Cf. Figure 1), dépendamment de la programmation qu'on lui implémente ainsi que l'information envoyée par l'objet connecté, il enverra une consigne (baisser la température, action le déshumidificateur...) à l'IoT afin qu'il l'exécute. Tout cet échange engendre une très grande consommation d'énergie, on estime que 80 % de la consommation énergétique des objets connectés est dédiée au maintien de leur connectivité au réseau [3]. Nous sommes contraints de trouver une autre alternative afin de réduire cette consommation, que ce soit pour une nécessité écologique ou financière.

Une solution est proposée : Imiter l'aptitude du cerveau humain à exécuter des tâches complexes tout en consommant une faible puissance. D'où l'objectif de ce stage, modéliser le fonctionnement électrique du neurone biologique, afin de concevoir des circuits électroniques neuromorphiques, donc des systèmes dont l'expression de la fonction de transfert qui s'adapte. La modélisation s'effectue sous Matlab puis les résultats de la simulation seront comparés avec ceux du laboratoire GEEPS (simulés sous Cadence).

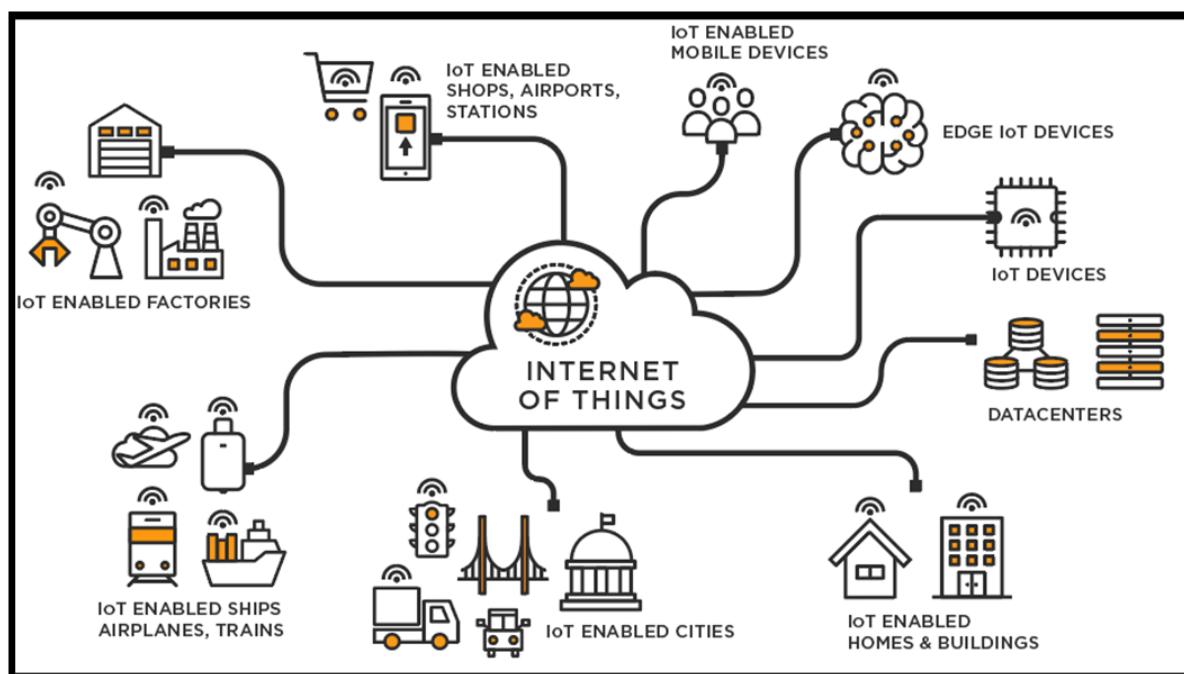


Figure.1 : Transmission d'informations via CLOUD [4]

## 2. Présentation de l'entité d'accueil

### 2.1 Le laboratoire Geeps

Le laboratoire est une unité mixte **CNRS**, **CentraleSupélec**, **Université Paris-Saclay** et **Sorbonne Université**. Créé en **2015**, il est le fruit de la fusion du **LGEP** (Laboratoire de Génie Électrique de Paris, unité mixte d'origine) avec une partie de l'équipe d'accueil d'**ex-Supélec** et l'équipe d'accueil **L2E** (Laboratoire d'Electronique et d'Electromagnétisme) de **Sorbonne Université**. Il est installé sur le campus de **CentraleSupélec** de l'**Université Paris-Saclay** à Gif-sur-Yvette et sur le campus Pierre et Marie Curie de **Sorbonne Université** à Paris [1].

Les travaux de recherche réalisés au sein de l'unité combinent une triple approche : théorie-modélisation numérique - caractérisation et validation expérimentale. Ils sont répartis sur **3 pôles** (Cf. Figure 1) qui permettent de mener des activités sur un continuum qui s'étend des matériaux aux systèmes électroniques ou de conversion d'énergie. Deux centres d'expertise transversaux viennent en appui. Le premier qui capitalise les travaux liés à une compétence historique du laboratoire sur la modélisation numérique des systèmes électromagnétiques avec une orientation vers les problèmes multiphysiques, couplés. Le second regroupe les nombreuses plateformes expérimentales du laboratoire avec pour objectif premier la mutualisation des compétences en matière d'instrumentation ainsi que le partage des savoir-faire et des moyens [1].

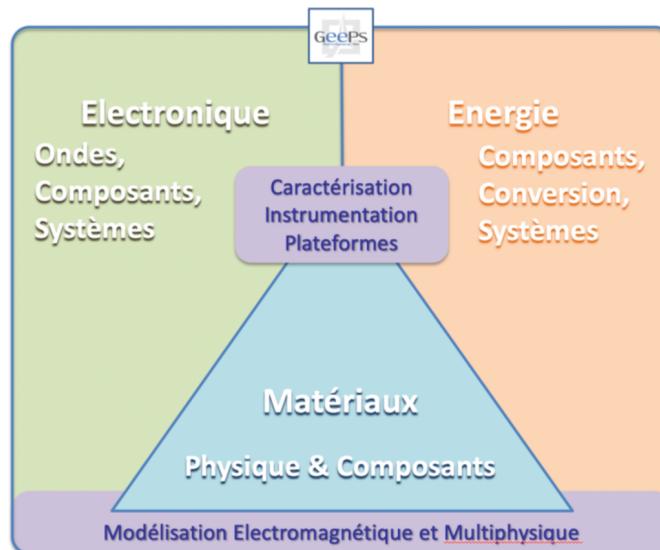


Figure. 1 : Les trois pôles du laboratoire GEEPS [1]

Durant notre stage nous étions affiliés au pôle Électronique, l'on avait donc deux lieux de travail, un situé sur le campus Pierre et Marie Curie de la faculté sciences et engineering de Sorbonne Université, l'autre site était situé sur le campus de CentraleSupélec de l'Université Paris-Saclay à Gif-sur-Yvette.

## 2.2 Le pôle électronique

Le pôle électronique se propose de répondre aux enjeux sociétaux définis dans le projet scientifique de **GeePs**, en offrant un éventail de solutions et de méthodes dans le domaine du traitement matériel de l'information.

Il développe pour cela une stratégie promouvant la transversalité et les études aux interfaces, et conduit une recherche en lien étroit avec le tissu industriel et le monde académique sur le plan national et international. Il focalise ses activités (Cf. Figure 2) autour :

- De la faible consommation, la miniaturisation et la fiabilité des circuits et systèmes intégrés (autonomie des objets connectés, bio-capteur)
- De la maîtrise de la génération, de la propagation et de la détection des ondes électromagnétiques (détection, localisation et focalisation de l'énergie et de l'information)
- Du couplage multiphysique de matériaux fonctionnels (micro sources d'énergie)
- Du contrôle électromagnétique des milieux complexes et CEM (modélisation et problème inverse)

Le pôle Électronique, structuré autour de CINQ thèmes dont TROIS transverses, dispose de compétences fortes dans le domaine de l'électronique intégrée, des ondes électromagnétiques et des capteurs. Il s'appuie, pour l'ensemble de ces activités, sur d'importantes ressources matérielles et logicielles incarnées par les deux centres d'expertise de **GeePs** (plateforme expérimentale de [caractérisation et d'instrumentation](#) et ressources informatiques et de calcul pour la [modélisation multiphysique et électromagnétique](#)).

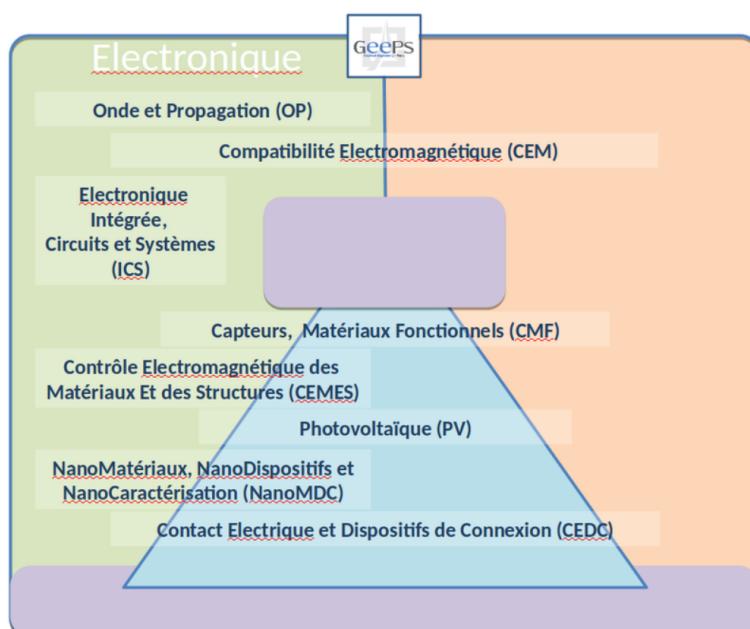


Figure. 2 : Le pôle électronique du laboratoire GEEPS [1]

## 3. Le Neurone biologique et sa modélisation

### 3.1 Fonctionnement du neurone biologique

#### 3.1.1 Le neurone

Le cerveau humain contient  $10^{11}$  neurones et  $10^{14}$  à  $10^{15}$  connexions [1]. Ces neurones peuvent différer selon leur rôle dans le cerveau et selon la structure qui les compose [2], mais tous sont basés sur la même structure [1]. Le fonctionnement du neurone repose sur plusieurs propriétés fonctionnelles [1] :

- Le neurone est capable de traiter les signaux en provenance de l'environnement externe au cerveau
- C'est un système d'intégration capable de traiter et de manipuler les signaux
- C'est un système capable de transférer des informations à distances

C'est la structure du neurone qui lui permet d'avoir ces propriétés. Le neurone est une cellule eucaryote (organisme dont le noyau cellulaire est séparé du cytoplasme par une membrane [3]), formé d'un corps cellulaire (souvent appelé soma) contenant un noyau et l'essentiel du cytoplasme [4]. Les neurones sont capables de recevoir, traiter, et de propager une information. Les structures permettant la transmission sont appelées "neurites" c'est -à-dire des prolongements par lesquels circule l'influx nerveux [1], ce dernier étant l'activité électrochimique transmise. Cet influx nerveux est transmis le long de deux types de neurites : l'axone et les dendrites. L'axone transmet l'influx nerveux en sortie du soma jusqu'au neurone suivant ; les dendrites quant à elle supporte l'information entrant dans le neurone [1]. L'axone est unique à chaque neurone, le nombre des dendrites s'élève à plus  $10^3$  environ pour chaque neurone [1] (Cf. Figure.1).

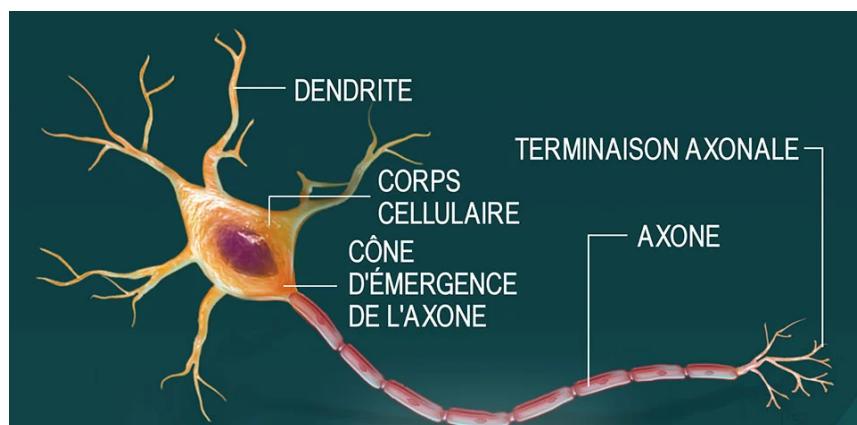


Figure. 1 : Structure d'un neurone [5]

### 3.1.2 L'influx nerveux

Dès la fin du XVIème siècle, Galvani et Volta montraient que l'information véhiculée le long des nerfs est en partie constituée de manifestations électriques [4], et ce grâce aux expériences de Galvani sur les grenouilles notamment. En effet, lesdites manifestations sont une conséquence de la dissolution des sels présents au sein de l'organisme, ces sels deviennent des ions, et sont donc porteurs de charges positives ou négatives (dans le système neuronale les principaux acteurs sont les ions de Sodium ( $\text{Na}^+$ ), Potassium ( $\text{K}^+$ ) et de chlorure ( $\text{Cl}^-$ )). La membrane cellulaire du neurone comporte différentes formes de "canaux à ions" (Cf. Figure 2), chaque canal est une sorte de nasse dont l'orientation dépend de l'ion qu'elle cible, privilégiant soit l'entrée soit la sortie du milieu cellulaire [1], ce fonctionnement est voltage-dépendant (i.e. La perméabilité à certains ions change suivant la différence de potentiel qui est appliquée sur les surfaces de la membrane).

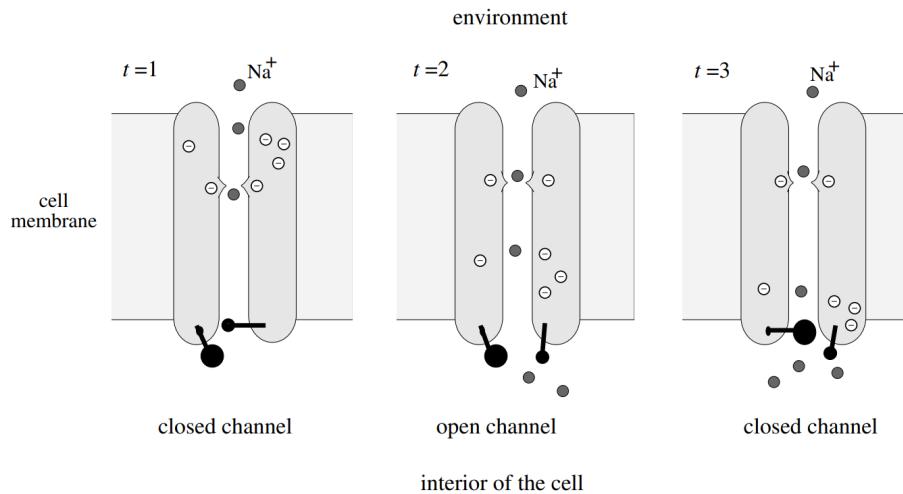


Figure. 2 : Canaux à ions [6]

Au repos (I.e. Absence d'excitation), la différence de potentiel au niveau de la membrane est d'environ -60 mV [4] (cette DDP varie d'une cellule à une autre). Tout se passe comme si les deux faces de la membrane correspondaient aux deux pôles d'un générateur de tension continue dont le pôle négatif serait situé à l'intérieur de la cellule et le pôle positif, côté extracellulaire [4].

Le corps cellulaire intègre l'influx nerveux reçu au niveau des dendrites, dès que l'influx nerveux entrant dépasse un certain seuil, le potentiel au niveau du neurone s'altère et cela produit des "potentiels d'action", ceux-là vont engendrer une vague de dépolarisation entraînant une hausse de la perméabilité aux ions de Potassium ( $\text{K}^+$ ) (dépolarisation) puis une hausse de la perméabilité aux ions de Sodium ( $\text{Na}^+$ ) (Cf. Figure 3) [1]. Ce potentiel d'action se propage le long de l'axone jusqu'à atteindre le neurone suivant et être récupéré par les dendrites afin d'être intégré à nouveau dans le neurone récepteur.

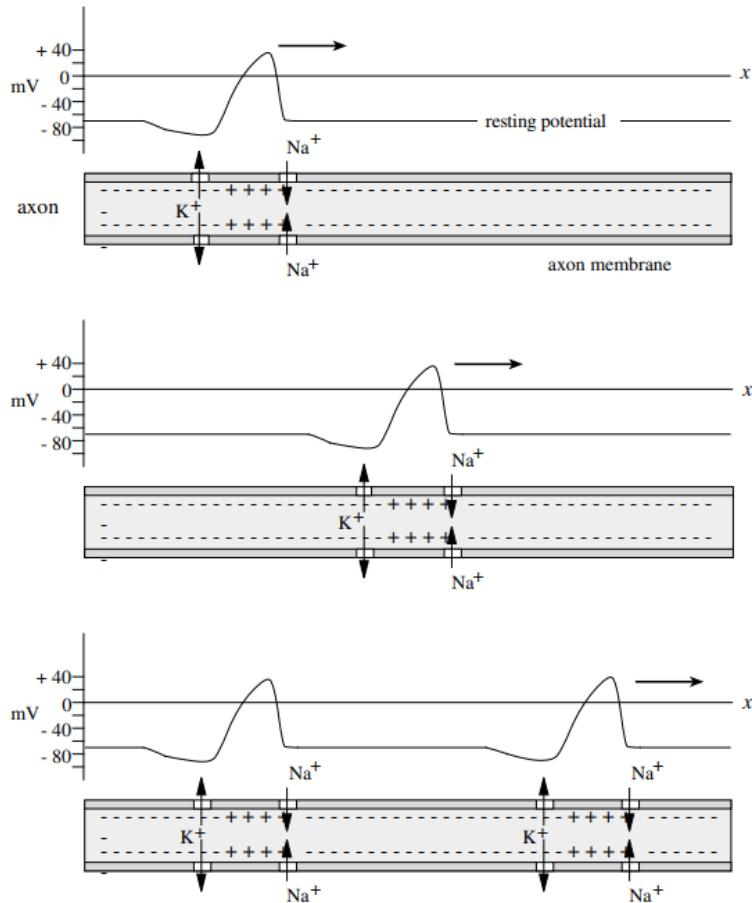


Figure. 3 : Transmission de l'influx nerveux [6]

### 3.1.2 Codage en fréquence

Le système nerveux fonctionne avec (entre autres) un codage en fréquence. C'est le nombre de potentiel d'action par seconde (fréquence) et les variations de fréquence (fréquence instantanée) qui code l'information. Un potentiel d'action isolé ne signifie rien [10]. Il est nécessaire de souligner que tous les potentiels d'action ont la même amplitude. On peut voir (Cf. Figure.4) comment l'angle de rotation du coude est fonction de la fréquence du signal ; les variations de fréquences codent alors la vitesse de rotation.

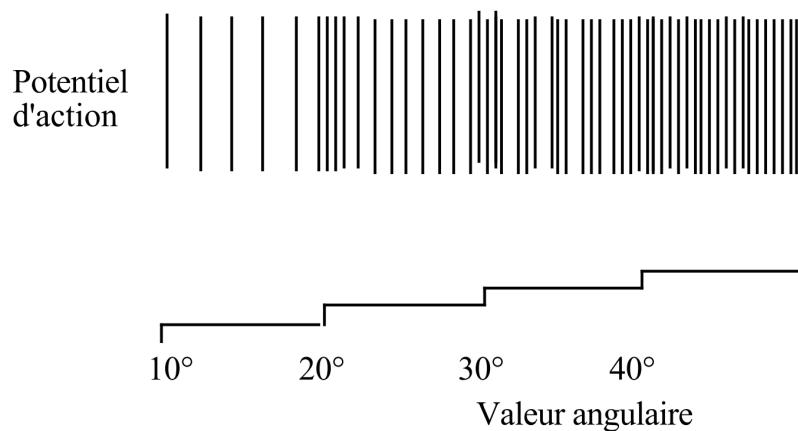


Figure. 4 : Codage en fréquence (mouvements d'une articulation telle que le coude). [10]

### 3.1.3 Le neurone en circuit électronique

L'on peut essayer de traduire le fonctionnement du neurone biologique en circuit électronique. La bicouche de phospholipides membranaires est constituée d'éléments non conducteurs, qui confèrent à la membrane des propriétés capacitives [4], la membrane sera donc représentée sous forme de capacité dans notre circuit. D'autre part, nous avons des protéines conductrices qui opposent une résistance au courant d'excitation [4]. Avec tous ces éléments pris en compte l'on peut avoir un circuit très simplifié modélisant le fonctionnement du neurone (Cf. Figure. 5), aussi appelé modèle LIF, que l'on verra plus en détail dans le chapitre 3.3 "Modèle LIF (Leaky Integrate Fire)".

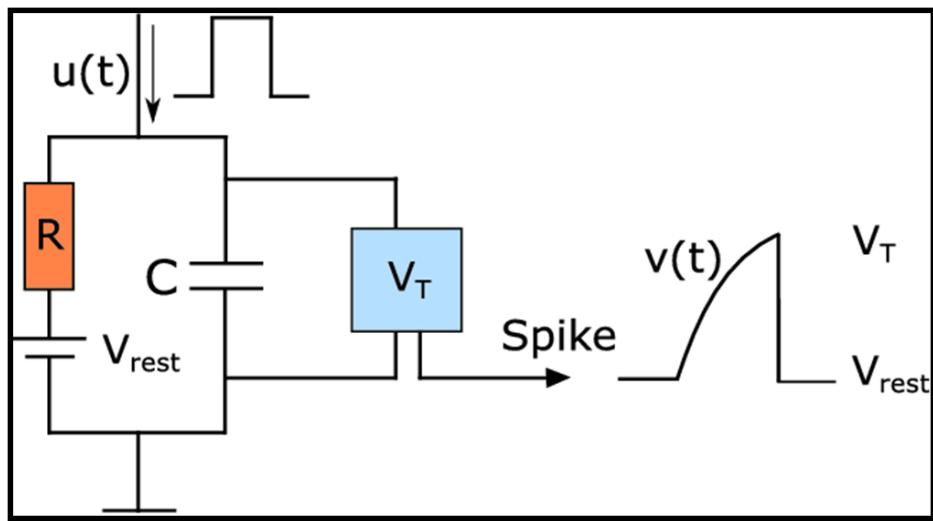


Figure. 5 : Schéma électrique reproduisant le fonctionnement du neurone biologique.  $R$  est la résistance engendrée par les protéines électriques. Les phospholipides éléments électriquement isolants représentent le condensateur  $C$ .  $V_{rest}$  est le potentiel de la membrane au repos.  $V_T$  un interrupteur électrique à seuil.  $U(t)$  un signal d'excitation.

$V(t)$  (ou Spike) est le potentiel d'action en sortie [7]

## 3.2 Modèles de neurones

Dans ce chapitre, nous verrons quelles sont les caractéristiques qu'il faut prendre en compte afin de juger l'efficacité d'un modèle neuronal. Puis nous allons étudier certains modèles, notamment les modèles de Hodgkin-Huxley et LIF, ce dernier étant celui que l'on nous a demandé de simuler sous Matlab.

### 3.2.1 La plausibilité biologique

Nous venons de voir une modélisation simplifiée du fonctionnement d'un neurone biologique ; ce modèle prenait en compte divers paramètres biologiques comme la résistance des protéines électriques, les propriétés capacitives de la membrane... On peut constater à priori qu'il manque certains éléments dans cette représentation compte tenu de ce qui a été dit dans le chapitre précédent, des paramètres manquants comme la faculté des dendrites à intégrer le flux nerveux entrant.

Cela introduit un paramètre "la plausibilité biologique" afin d'estimer si un modèle reflète correctement le fonctionnement biologique du neurone.

Compte tenu de la complexité du fonctionnement d'un neurone biologique et des paramètres qui entrent en jeu dans sa gestion des informations, la plausibilité biologique se retrouve en corrélation avec la complexité du modèle (complexité en terme mathématique).

Ce qui nous donne le graph de la figure 6 qui compare qualitativement la complexité mathématique des modèles en fonction de leur degré de bio-inspiration (i.e. la plausibilité biologique).

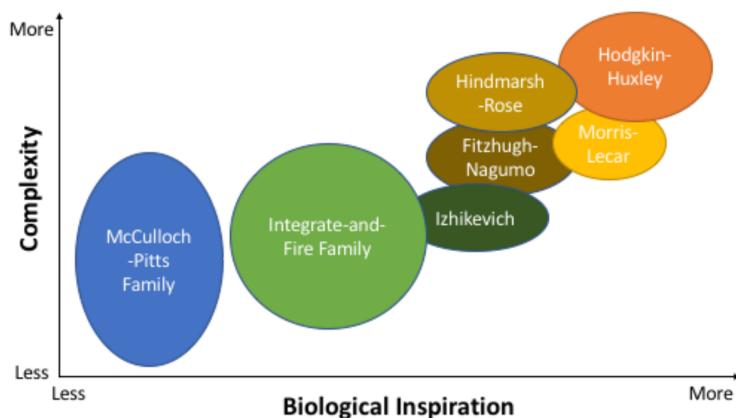


Figure. 6 : Comparaison qualitative des modèles de neurone [8]

### 3.2.2 Modèle Hodgkin-Huxley

Un des modèles les plus réputé en termes de plausibilité biologique est celui de Hodgkin-Huxley introduit en 1952, c'est un modèle relativement complexe, il nécessite des équations différentielles non-linéaire quadridimensionnelle, ce modèle permet de décrire le comportement du neurone en termes de transfert d'ions en entrée et en sortie du neurone [8]. Le circuit de Hodgkin-Huxley (Cf. Figure. 7), est constitué de quatre branches principales, une branche avec un condensateur  $C_M$  représentant la capacitance de la membrane, deux autres branches avec un générateur de courant ionique associé aux ions de Potassium ( $K^+$ ) (Respectivement les ions de Sodium ( $Na^+$ )) et des résistances (une résistance associée à chaque type d'ion). Ces résistances évoluent en fonction du temps et

du potentiel de la membrane, ce qui suggère que le changement dans la perméabilité est causé par l'effet qu'a le champ électrique sur la distribution ou l'orientation des molécules avec une charge ou un moment dipolaire [9]. Une autre branche est présente, celle des courants dits "de fuites", en fait les milieux intra et extracellulaire ne peuvent pas être considérés comme étant des courts-circuits, ils présentent une résistance vis-à-vis du passage d'un courant électrique [4] ; cela est dû à la présence d'ions de chlorure ( $\text{Cl}^-$ ) [9], étant donné que cette dernière est perméable à travers la membrane [11].

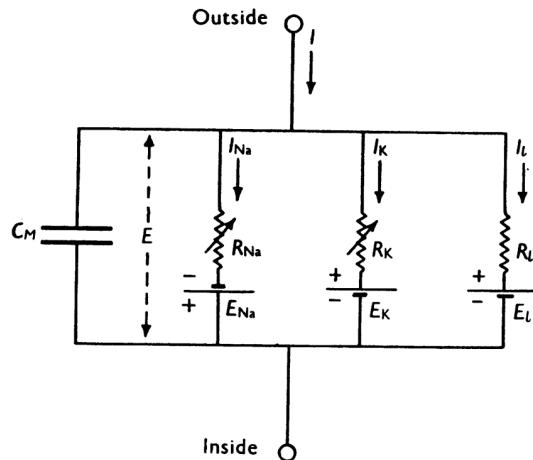


Figure. 7 : Circuit de HH modélisant la membrane [9]

En appliquant les lois de Kirchhoff sur le circuit de la figure. 7, nous obtenons l'équation suivante [9] :

$$I = C_M \frac{dV}{dt} + I_i \quad (1)$$

Avec

$I$  la densité de courant membranaire totale (courant entrant positif)

$I_i$  la densité de courant ionique (courant entrant positif)

$V$  le changement du potentiel membranaire depuis sa valeur au repos (dépolarisation négative)

$C_M$  la capacité de la membrane (que l'on considère constante)

$t$  le temps

Nous notons que :

$$I_i = I_{Na} + I_K + I_l \quad (2)$$

Mais aussi que

$$I_{Na} = g_{Na}(V - V_{Na}),$$

$$I_K = g_K(V - V_K),$$

$$I_l = \bar{g}_l(V - V_l),$$

Où

$$V = E - E_r,$$

$$V_{Na} = E_{Na} - E_r,$$

$$V_K = E_K - E_r,$$

$$V_l = E_l - E_r,$$

Avec

$\bar{g}_{Na}$ ,  $\bar{g}_K$ ,  $\bar{g}_l$  les conductances des ions qui exprime la perméabilité de la membrane  
( $\bar{g}_l$  est considérée comme constante)

$E_{Na}$ ,  $E_K$  sont les potentiels à l'équilibre de chaque ion

$E_r$  la valeur absolue du potentiel au repos de la membrane

$E_l$  le potentiel pour lequel le "courant de fuite" dû à la présence d'ions de chlorure ( $Cl^-$ ) et d'autres ions est nul

$V$ ,  $V_{Na}$ ,  $V_K$  et  $V_l$  peuvent être mesurés comme des "déplacements du potentiel au repos" [10]

En partant des équations (1) et (2), et en prenant en considération divers paramètres biologiques, notamment la dépendance des dimensions des fibres neuronales, dans [10] Hodgkin, A. L., et A. F. Huxley trouvent l'équation suivante :

$$\frac{a}{2R_2\theta^2} \frac{d^2V}{dt^2} = C_M \frac{dV}{dt} + \bar{g}_K n^4 (V - V_K) + \bar{g}_{Na} m^3 h (V - V_{Na}) + \bar{g}_l (V - V_l) \quad (3)$$

Avec

$\bar{g}_K$ ,  $\bar{g}_{Na}$ ,  $\bar{g}_l$  sont des conductances constantes

$n, m, h$  des quantité qui varient en fonction du temps suite à un changement dans le potentiel de la membrane

$a$  rayon de la fibre nerveuse

$\theta$  Vitesse de conduction de potentiel d'action

$R_2$  la résistance due à l'axoplasme, le liquide se trouvant à l'intérieur de l'axone

C'est une équation différentiel ordinaire dont la procédure à résoudre est compliquée car le paramètre  $\theta$  est inconnu, selon [10] il faudrait le deviner avant de commencer les calculs.

En conclusion, le modèle de Hodgkin-Huxley est fidèle au fonctionnement du neurone biologique, car comme vu précédemment il prend en considération toute une panoplie de paramètres biologiques impactant la transmission de données. Ceci étant, ce modèle devient fort complexe (que ce soit en compréhension théorique ou en application mathématique). Nous allons voir au chapitre suivant un modèle qui offre un bon compromis entre la complexité et le degré de bio inspiration.

### 3.2.3 Modèle LIF (Leaky Integrate Fire)

Le modèle LIF est un circuit offrant un bon compromis entre la complexité (complexité mathématique du modèle) et son degré de bio-inspiration. Ce modèle fait partie de la famille "Integrate and Fire Family" (Cf. Figure 6).

Le circuit qui représente le neurone biologique dans le modèle LIF est celui de la figure. 5 du chapitre 3.1.3 "Le neurone en circuit électronique" et peut être représenté par la figure. 8.

Le modèle LIF est une cellule RC, le condensateur dans ce circuit représente la membrane qui va continuellement se charger ou se décharger dépendamment de la valeur

de la tension à ses bornes (si cette tension dépasse celle dont la valeur sert de référence  $V_{th}$  le condensateur se décharge, en sortie du circuit nous aurons une exponentielle croissante. Dans le cas inverse, le condensateur se vide jusqu'à la valeur de repos). Les équations mathématiques qui régissent cette circuiterie seront développées davantage le chapitre suivant.

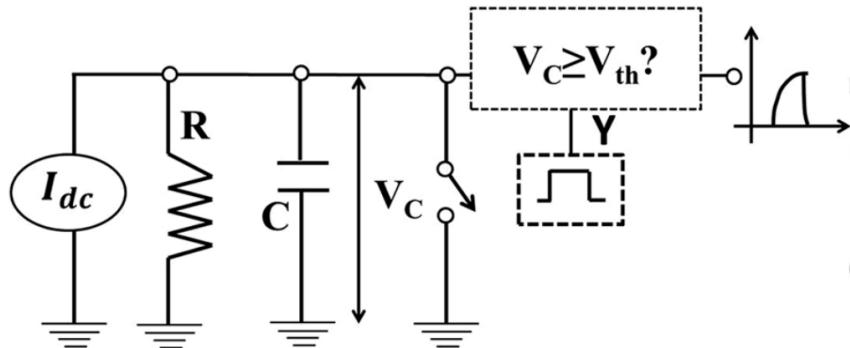


Figure. 8 : Circuit du modèle LIF [12]

L'équipe [13] propose le schéma de la figure. 9 afin de représenter le modèle LIF.

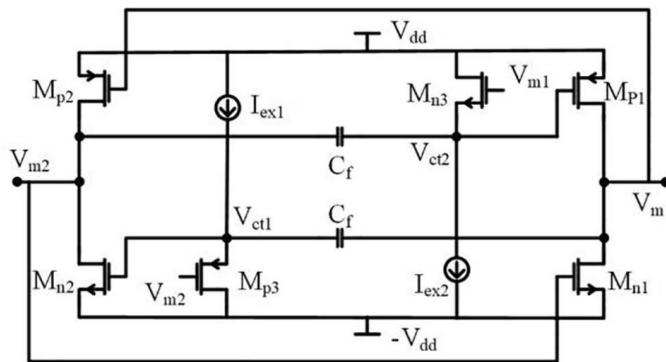


Figure. 9 : Circuit du modèle LIF amélioré en termes de consommation de puissance

Afin de simuler ce circuit l'équipe [13] utilise le logiciel Cadence, ce dernier est très précis néanmoins chronophage (Environ une journée pour simuler un seul neurone). Dans notre stage l'on nous demande de réaliser un programme sous Matlab capable de simuler le fonctionnement du circuit après avoir récupérer les données générées par la simulation sous Cadence. L'outil Matlab réduit considérablement le temps de simulation puisqu'il ne fait que décrire l'information produite par le logiciel Cadence. Cela facilitera l'analyse du circuit à nos encadrants durant leur recherche, lorsqu'ils étudieront des circuits à plusieurs neurones. Nous allons voir dans le chapitre suivant comment nous avons simulé le fonctionnement du neurone artificiel sous Matlab.

## 4. La modélisation sous Matlab

La modélisation sous Matlab consiste à mettre au point un algorithme qui simule approximativement les résultats fournis par Cadence. Puis, il nous revient à prendre en compte le courant d'excitation  $I_{ex}$  en interpolant les données fournies par Cadence, après une journée d'exécution, et l'effet de cet  $I_{ex}$  sur la fréquence de *Spike*.

### 4.1 Simulation LIF

La simulation du modèle LIF sous Matlab consiste à créer dans un premier temps un schéma bloc (Cf figure 1).

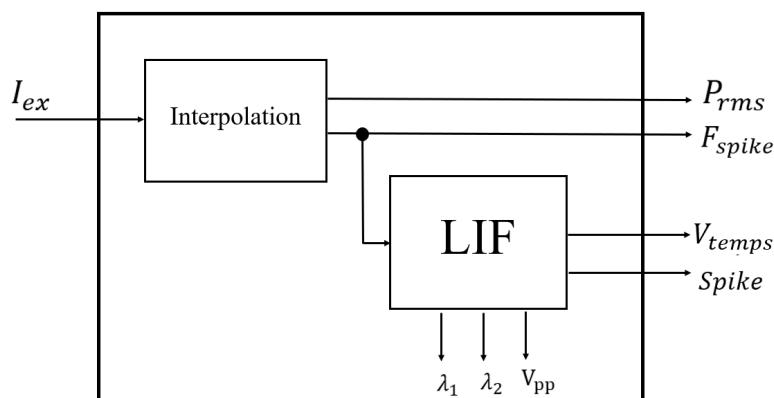


Figure. 1 : Schéma bloc du neurone LIF sous Matlab

Dans ce schéma bloc, on identifie :

- Deux blocs :
  - Un bloc d'interpolation (sera traité dans le chapitre 3.2)
  - Un bloc simulant le neurone LIF (sera traité dans le présent chapitre)
- En entrée :
  - Le courant d'excitation  $I_{ex}$ .
- En sortie :
  - $P_{rms}$  (la puissance consommée par un neurone)
  - $F_{spike}$  (la fréquence du potentiel d'action d'un neurone)
  - $V_{temps}$  (vecteur temps en ms)
  - *Spike* (potentiel d'action d'un neurone)

Par conséquent, il nous revient à modéliser dans un premier temps la fonction *Spike* pour une période comme qui suit :

$$Spike = e^{\lambda_1 \cdot t} \quad \text{pour } 0 \leq t \leq T_{on} \quad [1]$$

$$= e^{-\lambda_2 \cdot t} \quad \text{pour } T_{on} \leq t \leq T \quad (1)$$

Avec :

$T_{on}$  : temps de l'exponentielle croissante.

$T$  : période du potentiel d'action d'un neurone.

La valeur de  $T_{on}$  nous permet de positionner le pic à une position souhaitée dans l'intervalle  $T$ , la valeur de  $Vpp$  (constante) est l'amplitude du potentiel d'action, les valeurs de  $\lambda_1$  et  $\lambda_2$  sont obtenus à partir de la tension  $Vpp$  voulu. Ils sont définis par [1] :

$$\lambda_1 = \frac{\ln(Vpp)}{T_{on}}$$

$$\lambda_2 = \frac{\ln(Vpp)}{T-T_{on}}$$

Au final, nous obtenons :

$$\begin{aligned} Spike &= Vpp^{\frac{t}{T_{on}}} && \text{pour } 0 \leq t \leq T_{on} \\ &= Vpp^{-\frac{t}{T-T_{on}}} && \text{pour } T_{on} < t \leq T \end{aligned}$$

Après élaboration mathématique de ce modèle LIF, nous avons créé une fonction *LIF* sous Matlab pour le simuler. Cette dernière prend une entrée, issus de l'interpolation, et retourne plusieurs sorties comme le montre les figures 1-2.

ENTREE	NATURE	SORTIE	NATURE
$F_{spike}$	Fréquence du neurone LIF	$V_{temps}$	Vecteur temps en ms
		Spike	Vecteur de valeur du potentiel d'action en mV
		$\lambda_1$	Constante de l'exponentielle croissante en kHz
		$\lambda_2$	Constante de l'exponentielle décroissante kHz
		$V_{pp}$	Tension pic à pic en mV
		D'autre sorties si besoin	Par exemple période

Figure. 2 : Entrée et sorties de la fonction *LIF*.

Dans cette première fonction *LIF*, nous n'avons pas introduit le courant d'excitation  $I_{ex}$ . Mais, nous avons introduit la fréquence du neurone  $F_{spike}$  qui est indépendante de  $I_{ex}$ . Par conséquent, nous pouvons obtenir sous Matlab la courbe des *Spike* indépendante de l'excitation  $I_{ex}$  (Cf figure 4), après plusieurs difficultés. Ces difficultés auxquelles nous faisions face sont représentées par la figure 3.

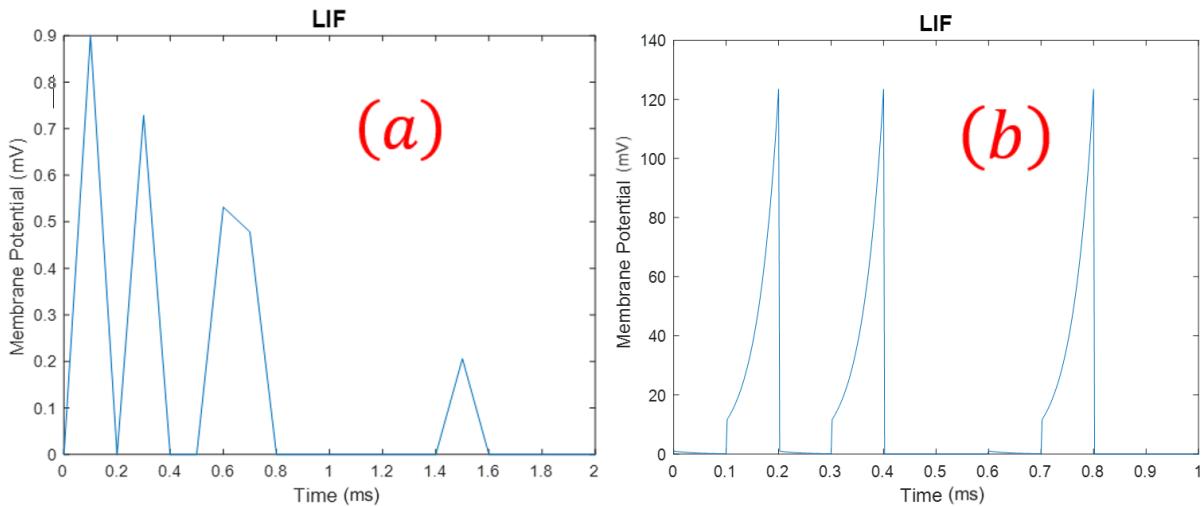


Figure. 3 : Difficultés rencontrées.

(a) Problème de diminution d'amplitude au cours du temps.

- Erreur à éviter :  $T_{on} = T_{on} + T$  (après chaque période)

(b) Variation de la période du signal

- Erreur à éviter :  $T = T + T$  (pour passer d'une période à l'autre)

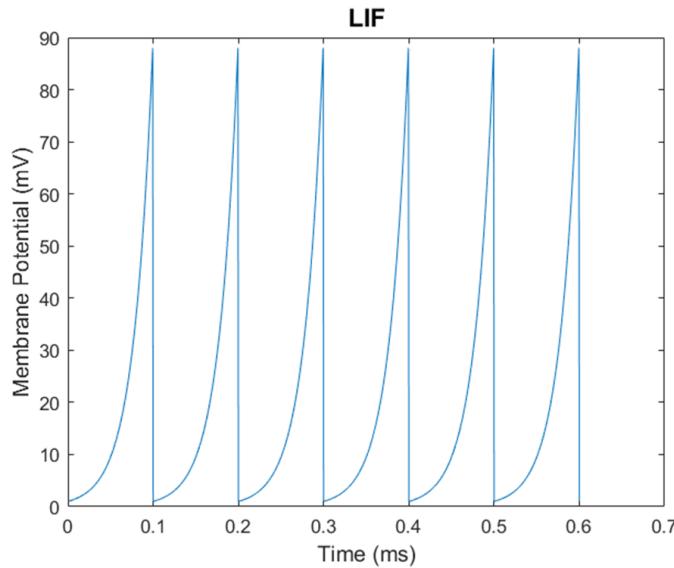


Figure. 4. Spike indépendante de  $I_{ex}$ .

Notre objectif étant d'être capable de générer ce même type de courbe (Cf figure 4) avec une dépendance entre la fréquence et le courant d'excitation  $I_{ex}$ , nous sommes contraints d'interpoler les données fournies par Cadence afin de créer cette dépendance dans notre programme.

## 4.2 Interpolation des données de Cadence sous Matlab

Maintenant que nous avons modélisé le neurone artificiel, il nous reste plus qu'à reproduire l'encodage en fréquence qu'effectue le neurone biologique, mais aussi la relation entre le courant d'excitation  $I_{ex}$  et la puissance ( $P_{RMS}$ ) consommée par le neurone artificiel (cette dernière permet aux chercheurs de déterminer le rendement d'un neurone artificiel).

Pour ce faire nous allons construire une courbe à partir des données d'un nombre fini de points, en l'occurrence ces points ont été déjà produits par la simulation via Cadence, sur la figure 5 nous pouvons voir les courbes résultant des simulations Cadence par nos encadrants [2].

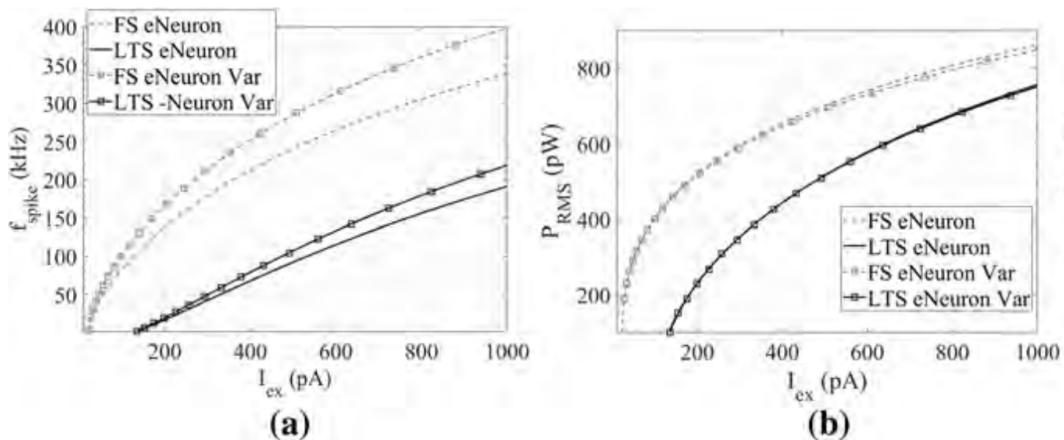


Figure. 5 : (a) Courbes de la fréquence  $F_{spike}$  en fonction du courant d'excitation  $I_{ex}$   
 (b) Courbes de la puissance  $P_{RMS}$  en fonction du courant d'excitation  $I_{ex}$   
 [2]

Notre objectif est de reproduire ces courbes et donc de déterminer l'expression mathématique qui lie la fréquence (respectivement la puissance) avec le courant d'excitation. Afin de réaliser cela, nous allons utiliser une commande *polyfit* (polynomial fitting, ajustement polynomial en français) de Matlab. D'abord, l'on va récupérer un tableau de trois colonnes et plus de trois cents échantillons et les stocker dans des vecteurs comme l'illustre la figure 6

```
lex= readtable('donnee.csv','Range','A3:A311');
lex=table2array(lex)
```

Figure. 6 : Récupération d'un tableau des valeurs de  $I_{ex}$  et reconversion en un vecteur

Nous lisons les valeurs provenant d'un fichier de type csv (dans la figure. 6 c'est le fichier donnée.csv), ce dernier est composé de trois colonnes (Cf. Figure 7) pour chaque grandeur, et de trois cent dix lignes.

Iex	Prms	fspike
17.58e-12	75.42e-12	1,00E+03
17.99e-12	75.72e-12	1,00E+03
18.41e-12	75.7e-12	1,00E+03
18.83e-12	76.05e-12	1,00E+03
19.26e-12	76.5e-12	1,00E+03
19.71e-12	76.7e-12	1,00E+03
20.16e-12	81.99e-12	2,00E+03
20.63e-12	86.94e-12	3,00E+03
21.1e-12	103.3e-12	5,00E+03
21.59e-12	110.4e-12	5,00E+03
22.09e-12	114.0e-12	6,00E+03
22.6e-12	124.1e-12	7,00E+03
23.12e-12	130.6e-12	8,00E+03
23.65e-12	144.3e-12	1,00E+04
24.2e-12	154.9e-12	1,20E+04
24.75e-12	162.0e-12	1,30E+04
25.33e-12	166.6e-12	1,40E+04

Figure. 7 : Tableau de valeurs produites par la simulation sous Cadence

Où :  $I_{ex}$  est en Ampère,  $P_{RMS}$  en Watt,  $F_{spike}$  en Hertz

Après avoir récupéré ses valeurs sous formes de vecteurs, nous allons procéder à l'interpolation. Nous utilisons la commande polyfit qui va nous renvoyer un vecteur ligne dont les composantes sont les coefficients d'un polynôme qui interpole le nuage de points de la figure 7. Polyfit utilise la méthode des moindres carrés, le logiciel Matlab essayera de dessiner une courbe traversant le maximum de points et ensuite fera converger le polynôme produit par son approximation (le faire converger vers les valeurs du nuage de points) au sens des moindres carrés, en minimisant l'énergie [3] :

$$S = \sum_0^N [y_i - f(x_i)]^2 \quad (2)$$

Avec :

$y_i$  Les valeurs mesurées (Dans notre cas, ils représentent les valeurs produites par Cadence)

$f(x_i)$  Les valeurs de notre fonction d'interpolation aux points  $x_i$

$N$  Le nombre d'échantillons à interpoler (dans notre cas N=310)

La commande Polyfit prend en argument l'ordre du polynôme que l'on souhaite avoir en retour, donc si nous avons une fonction qui suit une forme parfaitement quadratique, l'ordre à choisir est 2. Si notre nuage de points suit une fonction particulière (non polynomiale, une exponentielle par exemple) augmenter l'ordre en entrée augmentera la précision de notre interpolation. Nous pouvons voir une interpolation de l'ordre 2 du nuage de points sur la figure 8.

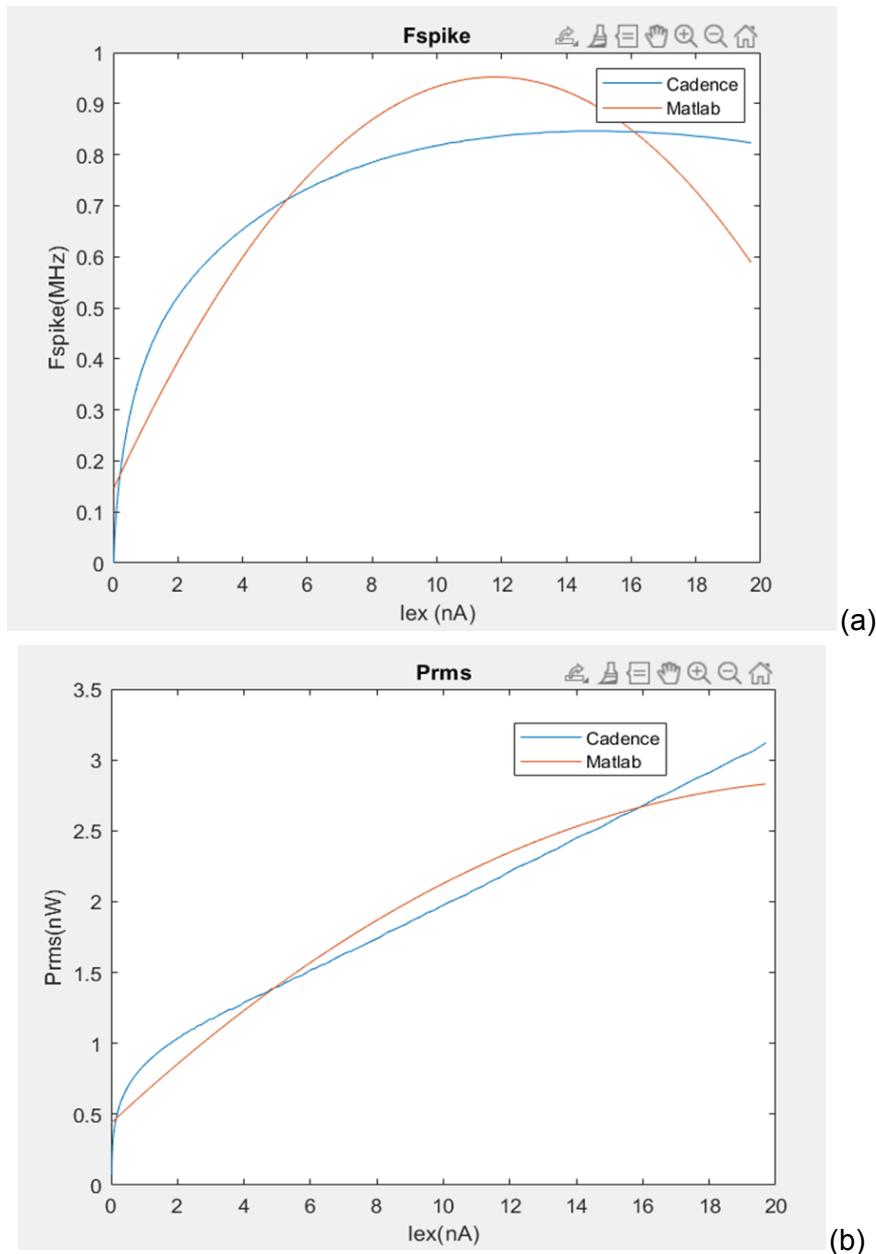


Figure. 8 : Comparaison interpolation/simulation Cadence

(a)  $F_{spike}$  en fonction de  $I_{ex}$

(b)  $P_{RMS}$  en fonction de  $I_{ex}$

Courbes rouges, courbes résultant de l'interpolation sous Matlab

Courbes bleus, courbes résultant de la simulation sous Cadence

On peut constater aisément que l'ordre deux ne convient pas comme ordre d'interpolation. Afin de déterminer le bon ordre il faut utiliser une figure de mérite afin d'estimer correctement l'erreur entre la courbe d'interpolation et celle de la simulation sous Cadence, et c'est ce que nous allons voir dans le chapitre suivant.

## 4.3 Estimation de l'erreur

Nous voyons qu'il existe un écart entre les courbes obtenues avec Cadence et celles obtenues avec Matlab. Cet écart, il nous revient de le réduire.

Pour réduire cet écart, nous avons plusieurs façons pour l'exprimer, par exemple :

- L'exprimer sous forme d'une erreur relative : erreur absolue (i.e. imprécision d'une mesure) par rapport à la taille de l'objet que l'on mesure. Il est utilisé pour mettre en perspective l'erreur et il est multiplié par 100 pour obtenir un pourcentage plus facile à lire et à comprendre [4].

$$\begin{aligned}
 \text{Absolute Error} &= \frac{\text{Measured Value} - \text{Expected Value}}{\text{Measured Value} - \text{Expected Value}} \\
 \text{Relative Error} &= \frac{\text{Absolute Error}}{\text{Actual Value}} \\
 &\downarrow \\
 \text{Relative Error} &= \frac{\text{Measured Value} - \text{Expected Value}}{\text{Actual Value}} \times 100
 \end{aligned}$$

Figure. 9. Erreur relative [4].

- L'exprimer sous forme d'un écart-type : une mesure de la dispersion des données dans un échantillon. Pour le calculer, que ce soit pour une série de données ou pour un échantillon issu de cette série, il nous faut en premier la moyenne et la variance de l'échantillon pour ensuite calculer l'écart-type. La variance est la mesure de la dispersion des données par rapport à la moyenne de l'échantillon. Ensuite, l'écart-type s'obtient en prenant la racine carrée de la variance de notre échantillon [5].
- L'exprimer sous forme d'une erreur quadratique moyenne (RMSE : root mean square) : une mesure utilisée pour estimer la précision du positionnement de deux courbes. RMSE est la racine carrée de la moyenne de l'ensemble des différences au carré entre les valeurs de coordonnées d'une base de données et les valeurs de coordonnées provenant d'une source indépendante de précision supérieure pour des points identiques [6].

$$RMSE = \sqrt{\sum_{i=1}^n \frac{(\hat{y}_i - y_i)^2}{n}}$$

Figure. 10. RMSE [7]

Notre encadrant nous a demandé de choisir d'exprimer l'écart mathématiquement sous forme d'une RMSE, voir figure 11, et sa version relative qu'on appelle RMSPE (Root Mean Square Percentage Error) comme suite :

$$\text{RMSPE} = \sqrt{\frac{100\%}{n} \cdot \sum_{i=1}^n \Delta X_{\text{rel},i}^2}$$

Avec

$$\Delta X_{\text{rel},i} = \frac{X_i}{T_i} - 1,$$

Figure. 11. RMSPE [8]

Enfin, il nous revient à calculer ces erreurs en fonction de l'ordre du polynôme qui nous est retourné par Polyfit. Nous obtenons au final :

Ordre de polynôme N	RMSE (Root Mean Square Error)		RMSPE (Root Mean Square Percentage Error)	
	Fspike	Prms	Fspike	Prms
1	0.1599	0.1907	31.9558	11.0328
2	0.0931	0.1560	21.1017	9.2490
3	0.0641	0.1260	15.6220	7.8207
4	0.0465	0.1021	12.0724	6.7031
5	0.0346	0.0845	9.5800	5.8475
6	0.0262	0.0719	7.7457	5.1931
8	0.0160	0.0553	5.3521	4.2478
10	0.0107	0.0448	3.9599	3.5804
12	0.0078	0.0371	3.0726	3.0659
14	0.0060	0.0315	2.4450	2.6622
16	0.0047	0.0274	1.9833	2.3397
18	0.0038	0.0240	1.6313	2.0667
20	0.0032	0.0213	1.3548	1.8347
23	0.0026	0.0184	1.0591	1.5804
26	0.0026	0.0196	1.0972	1.6123
27	0.0041	0.0214	0.9966	1.5581
28	0.0043	0.0244	1.0050	1.5793
29	0.0032	0.0626	0.9670	1.5507
30	NaN	NaN	NaN	NaN

Figure. 12. RMSE & RMSPE

Nous déduisons que ces erreurs diminuent lorsque l'ordre du polynôme augmente.

## 4.4 Comparaison avec les simulations sous cadence

Actuellement, nous avons pris en compte le courant d'excitation  $I_{ex}$  et nous sommes capables de gérer les erreurs quadratiques moyennes. Désormais, nous obtenons des courbes qui simulent le fonctionnement d'un neurone biologique excité, par exemple :

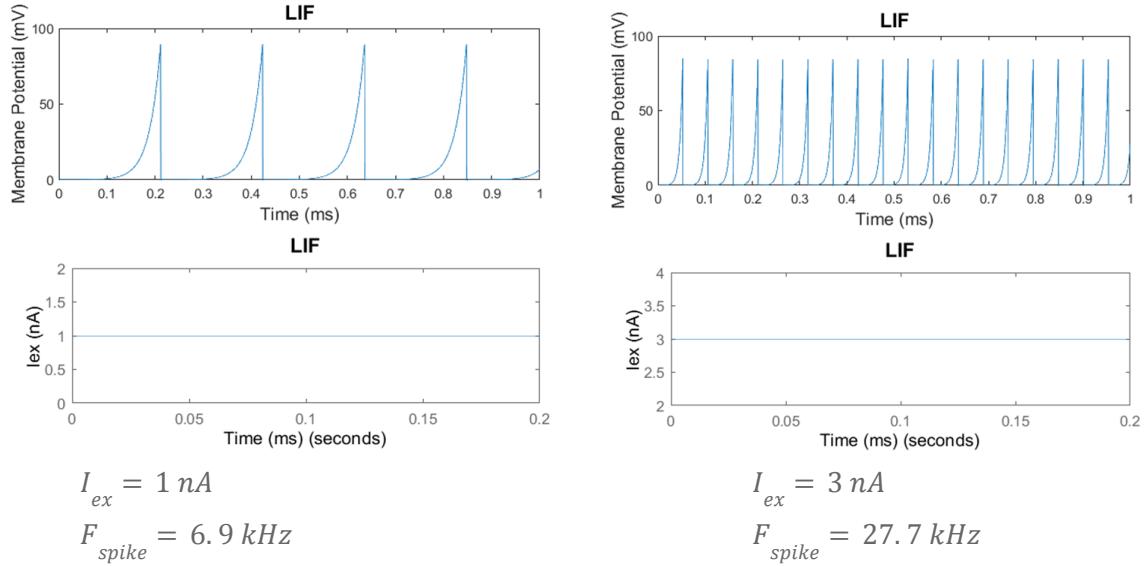


Figure. 13. Réponses de *Spike* suite à deux excitation  $I_{ex}$ .

Nous remarquons, dans la figure 13, que la fréquence de *Spike* augmente comme attendu avec le courant d'excitation  $I_{ex}$  : pour une excitation plus grande nous obtenons plus de *Spike*.

Enfin, il nous revient à comparer pour une excitation donnée, oscillations obtenues avec Cadence et celle obtenues avec Matlab (voir figure 14). Il y aura une bonne concordance lorsque l'erreur obtenue sera de moins de 10% (D'après nos recherches, cette condition lorsque l'ordre d'interpolation dépasse 7).

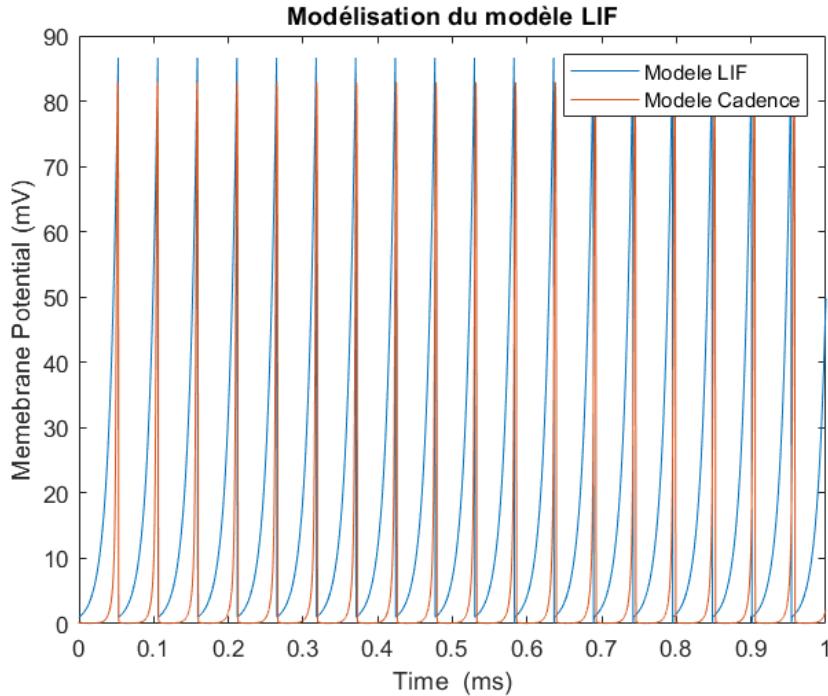


Figure. 14. Courbes de *Spike* obtenues suite à une excitation  $I_{ex} = 30\text{pA}$ .

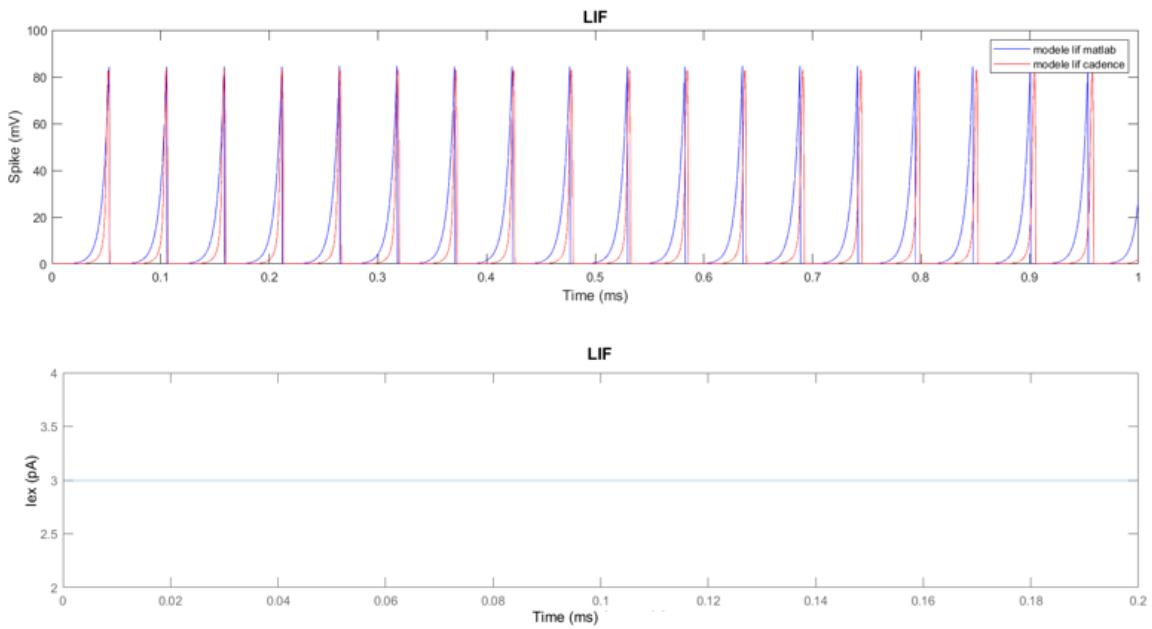
Actuellement, nous remarquons, dans la figure 14, que l'exponentielle décroissante reproduit assez bien la simulation sous Cadence (Cf figure 14), tandis que l'exponentielle croissante ne suit pas la simulation Cadence. Pour remédier à ce problème [1] a proposé le modèle amélioré suivant :

$$\begin{aligned} Spike &= A_1 \cdot e^{\lambda_1 \cdot t} + B_1 & 0 \leq t \leq T_{on} \\ &= A_2 \cdot e^{\lambda_2 \cdot t} + B_2 & T_{on} < t \leq T \end{aligned} \quad [1]$$

Avec :

$$\begin{aligned} \lambda_1 &= \frac{\ln(1 + \frac{V_{pp}}{A_1})}{T_{on}} & \text{pour} & & 0 < A_1 \\ \lambda_2 &= \frac{\ln(1 - \frac{V_{pp}}{A_2})}{T_{on-T}} & \text{pour} & & V_{pp} < A_2 \\ A_1 &= -B_1 & \text{et} & & A_2 = -B_2 \end{aligned}$$

Des nouvelles valeurs  $A_1$ ,  $A_2$ ,  $B_1$  et  $B_2$  de [1] sont ajoutées à la fonction *Spike*. Les valeurs  $A_1$  et  $A_2$  nous permettent de contrôler successivement l'allure de l'exponentielle croissante et décroissante ; les valeurs  $B_1$  et  $B_2$  font passer successivement l'exponentielle croissante à zéro lorsque  $t = 0$ , et à  $V_{pp}$  lors de l'exponentielle décroissante à  $t = T_{on}$ . Nous pouvons observer cette amélioration dans la figure 15.



Ordre de polynôme : 15      RMSPE : 9.2%  
Figure. 15. Oscillations obtenues suite à une excitation donnée

Nous remarquons, dans la figure 15, que pour une excitation donnée, il y a une bonne concordance entre les courbes d'oscillation obtenues par Cadence et Matlab. Puisque nous avons une erreur de moins de 10% lorsque le *Spike*, modélisé sous Matlab, est un polynôme d'ordre 15. Par conséquent, Matlab peut avantageusement remplacer Cadence lorsque nous aurons besoin de travailler sur un réseau de neurones.

## 5. Conclusion

L'expérience en laboratoire a été fort enrichissante que ce soit sur le plan intellectuel ou personnel. Le projet de modéliser le fonctionnement électrique d'un neurone biologique, (dans une perspective de délocalisation de l'intelligence artificielle au niveau des objets connectés) est challengeant et extrêmement intéressant.

Nous sommes sortis de notre zone de confort en se penchant vers un domaine complexe comme celui de la neurologie, nous permettant de découvrir une infime partie du fonctionnement de notre cerveau.

Cette infime partie nous l'avons modélisée, nous avons mis en oeuvre (avec l'aide précieuse de nos encadrants et de celle des enseignants qui ont répondu à nos questions lors de nos recherches) un programme capable de reproduire le mode de transmission d'informations du neurone sous Matlab tout en offrant à nos encadrants un algorithme ayant une figure de mérite leur permettant ainsi d'estimer la justesse de la simulation sous Matlab.

Après avoir réalisé cela, reste la prochaine étape. Nous souhaitons approfondir nos connaissances en neurologie et notamment des synapses (situé à la fin de l'axone, ce sont elles qui rendent l'humain capable de s'adapter), éventuellement mettre en œuvre des systèmes capables d'interagir avec les neurones et les synapses.

Tout cela nous oriente à choisir des projets impliquant des solutions neuro-inspirées, ainsi nous explorant davantage le fonctionnement neuronal, aussi être capable d'interagir avec ce dernier ou l'imiter.

## 6. Compétences acquises

Nous avons des connaissances sur le fonctionnement du neurone biologique, nous permettant ainsi de travailler sur de futurs projets portant sur les neurones (Machine Learning, interface homme machine comme des implants cérébraux d'interfaces neuronales directes).

Durant ce stage, nous avons effectué un certain nombre de recherches ce qui nous a amené à développer un esprit critique vis-à-vis des sources que l'on emploie, afin de comprendre le fonctionnement du neurone notamment. Par la même occasion nous effectuons des croisements entre toutes les sources que nous avions en notre possession afin de vérifier la justesse et la véracité d'une information donnée.

L'usage de Matlab afin de simuler le fonctionnement du neurone biologique a poussé nos connaissances en algorithmique. Aussi, nous avons affiné notre maniement de Matlab et nos aptitudes à résoudre des problématiques techniques.

Hormis les compétences d'ordre technique, nous avons développé les compétences relationnelles : la communication, l'écoute et l'objectivité lors des débats. Des compétences en termes d'organisation : planifier à l'avance via un agenda nos créneaux de travail avec des contraintes universitaires (Créneaux de cours qui parfois se retrouvent modifiés). Des compétences en termes de méthodologie de travail : rédiger un plan d'action regroupant nos aptitudes individuelles, ce que nous détenons et ce que l'on cherche à obtenir, avoir une ligne directrice à suivre durant le projet.

# Références

## 1. Introduction

[1] « Internet des objets ». Wikipédia, 12 novembre 2021. Wikipedia, [https://fr.wikipedia.org/w/index.php?title=Internet\\_des\\_objets&oldid=187951984](https://fr.wikipedia.org/w/index.php?title=Internet_des_objets&oldid=187951984).

[2] Judith Hurwitz, Robin Bloor, Marcia Kaufman et Fern Halper, Cloud Computing for Dummies, John Wiley & Sons, 2009 (ISBN 9780470484708).

[3] « More Data, Less Energy – Analysis ». IEA, <https://www.iea.org/reports/more-data-less-energy>. Consulté le 17 janvier 2022.

[4] « Qu'est ce que l'internet des objets (IoT) ». TIBCO Software, <https://www.tibco.com/fr/reference-center/what-is-the-internet-of-things-iot>. Consulté le 2 décembre 2021.

## 2. Présentation de l'entité d'accueil

[1] Accueil - [www.geeps.centralesupelec.fr](http://www.geeps.centralesupelec.fr). <https://www.geeps.centralesupelec.fr/index.php?page=accueil>. Consulté le 19 janvier 2022.

## 3. Le Neurone biologique et sa modélisation

[1] Rennard, Jean-Philippe. Réseaux neuronaux: une introduction accompagnée d'un modèle Java. Vuibert, 2006. P 11

[2] Mader, Sylvia S., et al. Biologie humaine. De Boeck, 2010. P 135

[3] Larousse, Éditions. Définitions : eucaryote - Dictionnaire de français Larousse.

<https://www.larousse.fr/dictionnaires/francais/eucaryote/31619>. Consulté le 17 janvier 2022.

[4] Richard, Daniel, et Didier Orsal. Neurophysiologie: organisation et fonctionnement du système nerveux. 3e éd, Dunod, 2007. P 11/17

[5] Body, Visible. Neurons. <https://www.visiblebody.com/fr/learn/nervous/neurons>. Consulté le 17 janvier 2022.

[6] Rojas, Raul. Neural Networks: A Systematic Introduction. Springer Science & Business Media, 2013.

[7] Chowdhury, Sayeed & Lee, Chankyu & Roy, Kaushik. (2020). Towards Understanding the Effect of Leak in Spiking Neural Networks, p.2.

[8] Schuman, Catherine D., et al. « A Survey of Neuromorphic Computing and Neural Networks in Hardware ». *arXiv:1705.06963 [cs]*, mai 2017. *arXiv.org*, <http://arxiv.org/abs/1705.06963>

[9] Hodgkin, A. L., et A. F. Huxley. « A Quantitative Description of Membrane Current and Its Application to Conduction and Excitation in Nerve ». *The Journal of Physiology*, vol. 117, no 4, 1952, p. 500-44. Wiley Online Library, <https://doi.org/10.1113/jphysiol.1952.sp004764>.

[10] TOUZET, Claude. les réseaux de neurones artificiels, introduction au connexionnisme. Ec2, 1992. P 15

[11] Alain, Faure. Classification et commande par Réseaux de neurones. 2006.

[12] Dutta, Sangya, et al. « Leaky Integrate and Fire Neuron by Charge-Discharge Dynamics in Floating-Body MOSFET ». *Scientific Reports*, vol. 7, no 1, août 2017, p. 8257. www.nature.com, <https://doi.org/10.1038/s41598-017-07418-y>.

[13] Daliri, Mojtaba, et al. « A Comparative Study Between E-Neurons Mathematical model and Circuit model ». *IET Circuits, Devices & Systems*, février 2021. HAL Archives Ouvertes, <https://doi.org/10.1049/cds2.12017>.

#### 4. La modélisation sous Matlab

[1] Geeps, R. H. (2021, July). Systèmes neuromorphiques pour les applications RF large bande. Riad Henider.

[2] Ferreira, Pietro M., et al. « Neuromorphic Analog Spiking-Modulator for Audio Signal Processing ». *Analog Integrated Circuits and Signal Processing*, vol. 106, no 1, janvier 2021, p. 261-76. DOI.org (Crossref), <https://doi.org/10.1007/s10470-020-01729-3>.

[3] Least Square Fit Method (Linear and non-linear). www.youtube.com, <https://www.youtube.com/watch?v=d5UC7wDZiQ8>. Consulté le 20 janvier 2022.

[4] Comment calculer l'erreur relative. wikiHow, <https://fr.wikihow.com/calculer-l%27erreur-relative>. Consulté le 20 janvier 2022.

[5] Comment calculer un écart type. wikiHow, <https://fr.wikihow.com/calculer-un-%C3%A9cart-type>. Consulté le 20 janvier 2022.

[6] "Root Mean Square Error - Definition - Learn CST -". Learn CST, <https://learncst.com/root-mean-square-error-rmse-definition/>. Consulté le 20 janvier 2022.

[7] Moody, James. « What Does RMSE Really Mean? » *Medium*, 6 septembre 2019, <https://towardsdatascience.com/what-does-rmse-really-mean-806b65f2e48e>.

[8] « rms - What is the correct definition of the root mean square percentage error (RMSPE)? » *Cross Validated*, [/413249/what-is-the-correct-definition-of-the-root-mean-square-percentage-error-rmspe](https://stats.stackexchange.com/questions/413249/what-is-the-correct-definition-of-the-root-mean-square-percentage-error-rmspe). Vu le 18 janvier 2022. <https://stats.stackexchange.com/questions/413249/what-is-the-correct-definition-of-the-root-mean-square-percentage-error-rmspe>