Neurones à impulsion pour les communications sans fil

Guillaume Marthe

N° d'ordre NNT : 2024ISAL0094

THESE de DOCTORAT DE L'INSA LYON,
membre de l'Université de Lyon

Ecole Doctorale 160
ELECTRONIQUE, ELECTROTECHNIQUE, AUTOMATIQUE DE LYON (EEA)

Spécialité/ discipline de doctorat :
Traitement du Signal et de l'Image

Soutenue publiquement/à huis clos le 12/11/2024, par :

Guillaume Marthe

Neurones à impulsion pour les communications sans fil

Devant le jury composé de :

GIRAU Bernard	Professeur des Universités	Université de Lorraine	Président
JULIEN-VERGONJANNE Anne	Professeur des Universités	Université de Limoges	Rapporteure
MARTINET Jean	Professeur des Universités	Université Côte d'Azur	Rapporteur
TEULIERE Céline	Maître de Conférences	Université Clermont Auvergne	Examinatrice
GOURSAUD Claire	Maître de conférence	INSA Lyon	Directrice
CLAVIER Laurent	Professeur	IMT Nord Europe	Co-encadrant

Département FEDORA – INSA Lyon - Ecoles Doctorales

SIGLE	ECOLE DOCTORALE	NOM ET COORDONNEES DU RESPONSABLE
ED 206 CHIMIE	**CHIMIE DE LYON** **https ://www.edchimie-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage **secretariat@edchimie-lyon.fr**	**M. Stéphane DANIELE** C2P2-CPE LYON-UMR 5265 Bâtiment F308, BP 2077 43 Boulevard du 11 novembre 1918 69616 Villeurbanne **directeur@edchimie-lyon.fr**
ED 341 E2M2	**ÉVOLUTION, ÉCOSYSTÈME, MICROBIOLOGIE, MODÉLISATION** **http ://e2m2.universite-lyon.fr** Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 **secretariat.e2m2@univ-lyon1.fr**	**Mme Sandrine CHARLES** Université Claude Bernard Lyon 1 UFR Biosciences Bâtiment Mendel 43, boulevard du 11 Novembre 1918 69622 Villeurbanne CEDEX **e2m2.codir@listes.univ-lyon1.fr**
ED 205 EDISS	**INTERDISCIPLINAIRE SCIENCES-SANTÉ** **http ://ediss.universite-lyon.fr** Sec. : Bénédicte LANZA Bât. Atrium, UCB Lyon 1 Tél : 04.72.44.83.62 **secretariat.ediss@univ-lyon1.fr**	**Mme Sylvie RICARD-BLUM** Laboratoire ICBMS - UMR 5246 CNRS - Université Lyon 1 Bâtiment Raulin - 2ème étage Nord 43 Boulevard du 11 novembre 1918 69622 Villeurbanne Cedex Tél : +33(0)4 72 44 82 32 **sylvie.ricard-blum@univ-lyon1.fr**
ED 34 EDML	**MATÉRIAUX DE LYON** **http ://ed34.universite-lyon.fr** Sec. : Yann DE ORDENANA Tél : 04.72.18.62.44 **yann.de-ordenana@ec-lyon.fr**	**M. Stéphane BENAYOUN** Ecole Centrale de Lyon Laboratoire LTDS 36 avenue Guy de Collongue 69134 Ecully CEDEX Tél : 04.72.18.64.37 **stephane.benayoun@ec-lyon.fr**
ED 160 EEA	**ÉLECTRONIQUE, ÉLECTROTECHNIQUE, AUTOMATIQUE** **https ://edeea.universite-lyon.fr** Sec. : Philomène TRECOURT Bâtiment Direction INSA Lyon Tél : 04.72.43.71.70 **secretariat.edeea@insa-lyon.fr**	**M. Philippe DELACHARTRE** INSA LYON Laboratoire CREATIS Bâtiment Blaise Pascal, 7 avenue Jean Capelle 69621 Villeurbanne CEDEX Tél : 04.72.43.88.63 **philippe.delachartre@insa-lyon.fr**
ED 512 INFOMATHS	**INFORMATIQUE ET MATHÉMATIQUES** **http ://edinfomaths.universite-lyon.fr** Sec. : Renée EL MELHEM Bât. Blaise PASCAL, 3e étage Tél : 04.72.43.80.46 **infomaths@univ-lyon1.fr**	**M. Hamamache KHEDDOUCI** Université Claude Bernard Lyon 1 Bât. Nautibus 43, Boulevard du 11 novembre 1918 69 622 Villeurbanne Cedex France Tél : 04.72.44.83.69 **direction.infomaths@listes.univ-lyon1.fr**
ED 162 MEGA	**MÉCANIQUE, ÉNERGÉTIQUE, GÉNIE CIVIL, ACOUSTIQUE** **http ://edmega.universite-lyon.fr** Sec. : Philomène TRECOURT Tél : 04.72.43.71.70 Bâtiment Direction INSA Lyon **mega@insa-lyon.fr**	**M. Etienne PARIZET** INSA Lyon Laboratoire LVA Bâtiment St. Exupéry 25 bis av. Jean Capelle 69621 Villeurbanne CEDEX **etienne.parizet@insa-lyon.fr**
ED 483 ScSo	**ScSo**[1] **https ://edsciencessociales.universite-lyon.fr** Sec. : Mélina FAVETON Tél : 04.78.69.77.79 **melina.faveton@univ-lyon2.fr**	**M. Bruno MILLY** (INSA : J.Y. TOUSSAINT) Univ. Lyon 2 Campus Berges du Rhône 18, quai Claude Bernard 69365 LYON CEDEX 07 Bureau BEL 319 **bruno.milly@univ-lyon2.fr**

1. ScSo : Histoire, Géographie, Aménagement, Urbanisme, Archéologie, Science politique, Sociologie, Anthropologie

Remerciements

Le document suivant ne serait pas ce qu'il est sans la présence d'un grand nombre de personnes, que je ne pourrais jamais assez remercier pour le rôle qu'elles ont joué dans ma vie ou mon parcours.

Je voudrais tout d'abord remercier M. Jean MARTINET et Mme Anne JULIEN-VERGONJANNE d'avoir accepté d'être les rapporteurs de mon manuscrit. Merci pour le temps passer à la relecture, et pour vos retours très utiles et constructifs. Un grand merci également à Bernard GIRAU et Céline TEULIERE d'avoir accepté de faire partie de mon jury de thèse, et d'avoir également relu mon manuscrit. Les discussions lors des questions ont été particulièrement enrichissantes.

Je remercie très sincèrement Claire et Laurent, mes encadrants, pour m'avoir fait confiance et m'avoir choisi comme doctorant sur ce projet qui me tenait à coeur. Claire, j'ai énormément apprécié ces trois années de collaboration au Citi, je serais éternellement reconnaissant de la chance que tu m'as donné et des libertés de recherche que tu m'as laissées tout au long de doctorat. Cette expérience aura été extrêmement enrichissante et tu y es pour beaucoup.

En toute logique, je remercie mes parents, pour leur soutien inconditionnel tout au long de ma vie et d'autant plus ces trois derniers années. Je ne pourrais jamais assez vous remercier de la chance que j'ai eu de vous avoir à mes cotés, d'avoir pu faire des études et en arriver là et de tout ce que vous avez fait pour moi ! Pour tout ça et bien plus, un éternel merci.

Camille, je ne pourrais jamais te remercier suffisamment pour le soutien que tu m'as apporté pendant ces trois années pas toujours simples. Merci d'avoir été là pour moi, je n'en serais sûrement pas là aujourd'hui sans toi. Merci de me rendre heureux comme tu le fais.

Pour rester dans la famille, un remerciement tout particulier à Cécile pour tout le temps passer à corriger mon manuscrit, j'espère qu'il restait quelque chose des stages de français. Merci également à Geneviève et Raymonde pour leur soutien sans faille. Merci aussi à Alex pour le temps à converser sur des passions, scientifiques ou non, depuis et pour longtemps. Enfin, merci au reste de la famille pour avoir toujours cru en moi et m'avoir poussé à aller toujours plus loin.

Un énorme merci aussi à Jordan pour sa présence et son soutien depuis toutes ces années, et je l'espère, pour encore beaucoup à venir.

Enfin, je voudrais remercier mes collègues du Citi pour toutes les interactions très enrichissantes que l'on a pu avoir tout au long de mon passage au laboratoire. Un remerciement tout particulier à Alix, Aurélien et Maxime pour avoir été à mes cotés dans toutes les épreuves du doctorat, celui ci n'aurait pas été le même sans vous.

Merci à tous.

Résumé

Dans le contexte de l'Internet des Objets, l'un des plus grands défis réside dans la gestion énergétique. Les radios à réveil (Wake-up Radio) permettent aux dispositifs de rester en veille tout en consommant très peu d'énergie, se réveillant uniquement lors de la réception de signaux spécifiques.

Dans cette thèse, nous proposons d'utiliser les réseaux de neurones à impulsions (SNNs) comme Wake-up Radio (WuR). Le rôle du réseau de neurones sera de reconnaître la séquence d'activation du noeud concerné dans un flux de bits, afin de le réveiller.

Nous présentons tout d'abord les hypothèses et modèles de neurones, de réseau et de signaux utilisés pour notre étude. La première contribution est de montrer la pertinence de ces réseaux.

Notre seconde contribution a été l'étude et la proposition du modèle Saturating Leaky Integrate and Fire pour le conception d'une WuR. Dans cette partie, nous proposons d'utiliser un phénomène bio-inspiré appelé Interaction Synaptique afin de produire un filtre temporel dépendant de l'Inter-Spike Timing. Nous étudions les paramètres de ce modèle afin de comprendre comment adapter cette plage d'Inter-Spike Timings. L'originalité de cette contribution est de proposer un nouveau moyen de reconnaître dans le domaine de l'analogique des séquences temporelles.

Par la suite, différentes topologies de réseaux de neurones Saturating Leaky Integrate and Fire (SLIF) ont été explorées, notamment une topologie en ligne, en losange et réseau multi-couches, afin de comprendre comment le réseau répond aux séquences d'impulsions.

Cette thèse établit ainsi les bases pour des recherches futures sur l'utilisation des réseaux neuromorphiques dans les dispositifs Internet des Objets (IoT) à faible consommation d'énergie, notamment dans les WuRs. Les travaux accomplis ouvrent la voie à la conception de réseaux de neurones capables de traiter des signaux temporels complexes tout en maximisant l'efficacité énergétique, répondant ainsi aux exigences des applications IoT.

Mots-clés

Réseau de neurones à impulsion, neurone à impulsion, Wake-Up Radios, détection de séquence d'activation, saturation synaptique, interaction synaptique, système faible puissance, intégration temporelle

Table des matières

Tables des acronymes

ANN Artificial Neural Network
AWGN Additive White Gaussian Noise
CMOS Complementary Metal-Oxide-Semiconductor
FA Fausse Alarme
FN False Negatives
FP False Positives
IF Integrate and Fire
IoT Internet des Objets
ISM Industrielle, Scientifique et Médicale
IST Inter-Spike Timing
LIF Leaky Integrate and Fire
MD MisDetection
OOK On-Off Keying
RF Radio Fréquence
SLIF Saturating Leaky Integrate and Fire
SNN Spiking Neural Network
STDP Spike-Timing-Dependent Plasticity
TN True Negatives
TP True Positives
TW TimeWidth
WuR Wake-up Radio

Chapitre 1

Introduction

Le domaine de l'IoT connaît une croissance exponentielle et occupe désormais une place de choix dans de nombreux secteurs, allant de l'industrie manufacturière à l'agriculture, en passant par les applications domestiques et les usines intelligentes. Cette prolifération est rendue possible notamment grâce à l'utilisation de capteurs interconnectés, qui permettent d'automatiser diverses tâches, telles que la surveillance en temps réel, la gestion des ressources ou encore la maintenance prédictive, contribuant ainsi à une efficacité accrue dans ces domaines clés [1, 2].

Toutefois, ces avancées technologiques s'accompagnent de défis, principalement liés à la gestion des ressources limitées des nœuds IoT. Ces dispositifs sont contraints par leur mémoire, leur puissance de calcul et, surtout, par l'énergie disponible. En effet, cette dernière constitue l'obstacle majeur à l'implémentation d'applications sur de longues durées, car elle limite directement la longévité des nœuds [3]. Maximiser l'efficacité énergétique est donc une priorité cruciale dans le domaine de l'IoT. La partie communication, en particulier, demeure l'une des opérations les plus gourmandes en énergie. Ce constat a conduit au développement de multiples techniques visant à optimiser la consommation, notamment la méthode du cycle de service, qui consiste à activer et désactiver l'émetteur-récepteur selon un planning précis, afin de réduire les périodes inutiles d'écoute [4]. Cependant, ces techniques ne parviennent pas à éliminer entièrement les pertes énergétiques associées aux phases d'écoute où aucune donnée n'est transmise. A l'inverse, rallonger le temps de veille introduit du délai. Cela induit donc un compromis entre la latence de transmission et la consommation d'énergie.

Les WuRs à ultra-basse consommation ont émergé comme une solution efficace pour pallier ce problème. Ces dispositifs permettent de surveiller le canal de diffusion de manière continue, tout en consommant beaucoup moins d'énergie que les récepteurs traditionnels. Contrairement aux systèmes habituels, ces WuRs ont pour seule mission de surveiller le canal. Ils activent le micro-contrôleur du nœud IoT uniquement lorsqu'un signal spécifique, appelé Wake-up Beacon, est détecté. Ce mécanisme, basé sur des interruptions, permet de réduire drastiquement les pertes énergétiques en laissant le noeud principal le plus en veille possible, et d'améliorer significativement le rendement global du système, tout en diminuant la latence [5, 6, 7]. Cependant, bien que les WuRs actuelles offrent déjà des avantages en matière de consommation d'énergie, la plupart reposent encore sur des micro-contrôleurs à faible puissance pour la reconnaissance des signaux. Ces micro-contrôleurs peuvent consommer jusqu'à $200\mu W$, ce qui limite les gains en termes d'efficacité énergétique des nœuds IoT [5].

Pour réduire au maximum la consommation d'énergie, les WuRs se caractérisent sou-

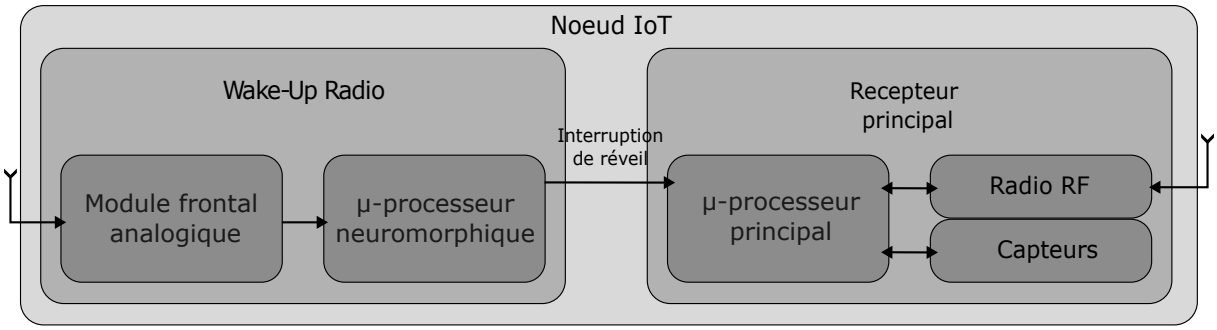


FIGURE 1.1 – Structure de l'implémentation d'une WuR à l'aide d'un SNN

vent par une sensibilité plus faible et un débit binaire inférieur à celui des émetteurs-récepteurs classiques de l'IoT [8]. Par conséquent, les séquences d'identification doivent être envoyées à des puissances plus élevées et à un débit binaire plus bas, ce qui augmente le coût énergétique de chaque transmission. Il est donc nécessaire de trouver un compromis entre la sensibilité de la WuR et la consommation énergétique de la transmission. Par exemple, une WuR d'une puissance de $100\mu W$ peut offrir une sensibilité de $-90dBm$, tandis qu'une WuR d'une puissance de $10\mu W$ ne peut garantir qu'une sensibilité de $-60dBm$ [9]. Ce compromis reste un enjeu majeur pour optimiser l'efficacité énergétique des systèmes IoT.

Dans un contexte où la loi de Moore atteint ses limites, les systèmes bio-inspirés apparaissent comme une solution particulièrement prometteuse pour surmonter ces contraintes. Le cerveau humain, avec son réseau de neurones qui communiquent via des impulsions électriques (ou spikes), est un exemple de système optimisé pour minimiser la consommation d'énergie tout en accomplissant des tâches extrêmement complexes. Chaque neurone émet une série temporelle d'impulsions, dont la position et la fréquence véhiculent l'information, permettant des fonctions telles que l'apprentissage, la classification ou encore la détection, tout en maximisant l'efficacité énergétique.

C'est dans cette optique que le projet ANR U-WAKE 20-CE24-0005 se concentre sur le développement d'une WuR bio-inspirée, en s'appuyant sur des composants Complementary Metal-Oxide-Semiconductor (CMOS) ainsi que sur un récupérateur d'énergie électromagnétique. L'originalité de cette solution repose sur la combinaison d'un démodulateur radiofréquence avec un détecteur neuro-inspiré et un SNN, permettant un traitement des données à très faible consommation. L'objectif est de créer une WuR ultra-basse consommation, fonctionnant avec une tension de quelques centaines de millivolts pouvant être alimentée avec l'énergie récupérée [10, 11].

Alors que les Artificial Neural Networks (ANNs) et le machine learning ont prouvé leur efficacité dans le domaine des télécommunications, notamment pour la gestion des réseaux [12] et de la couche physique [13], les SNNs n'ont pas encore été largement adoptés pour les communications sans fil, malgré leur potentiel en termes d'efficacité énergétique et leur capacité à réaliser des fonctions complexes [14].

Dans le cadre de ce projet, ma thèse porte sur le développement d'un détecteur neuromorphique capable de classifier des signaux Radio Fréquence (RF), en particulier des séquences courtes codées, en s'appuyant sur un réseau de neurones à impulsions. L'apprentissage en ligne étant hors du champ de cette recherche, les poids synaptiques du détecteur sont calculés à l'avance. Le système pourra fonctionner dans les bandes de fréquences $868MHz$ ou $2.4GHz$ et sera capable de reconnaître des signaux comme OOK

ou BPSK. Grâce à sa très faible consommation d'énergie, il pourra être auto-alimenté, augmentant ainsi la longévité des nœuds IoT par rapport aux solutions actuelles.

L'objectif final du projet est de réunir toutes ces contributions pour aboutir à la réalisation d'un prototype de nœud autonome fonctionnant dans les bandes Industrielle, Scientifique et Médicale (ISM), équipé d'une WuR consommant moins de $10\mu W$ tout en atteignant une sensibilité d'environ $-90dBm$, des performances qui n'ont jamais été atteintes ensemble jusqu'à présent.

Celui de cette thèse est d'évaluer la pertinence des SNNs pour la réalisation d'une WuR, puis l'étude des différents modèles pour déterminer le plus adapté à cette tache. Pour cela, nous allons nous appuyer sur les neurones ultra basse consommation développés par l'un des partenaires du projet (IEMN Lille). Enfin, le but sera de définir conjointement la topologie et le type de séquence à employer pour optimiser ce réseau de neurone.

Chapitre 2

Contexte scientifique

2.1 Introduction aux réseaux de neurones à impulsion

Les réseaux de neurones à impulsion, (SNN pour Spiking Neural Network) sont une des approches les plus prometteuses en neurosciences computationnelles et en intelligence artificielle. Ces réseaux modélisent le traitement de l'information de manière plus réaliste en imitant les neurones biologiques, qui utilisent des impulsions pour la communication et le calcul. Contrairement aux réseaux de neurones artificiels traditionnels (ANN) qui utilisent des fonctions continues pour représenter les activations neuronales, les SNNs utilisent des impulsions, apportant ainsi une dimension temporelle au traitement des données [15, 16]. Les réseaux de neurones à impulsions commencent à émerger en tant que solution à faible puissance, dans le cadre de l'analyse de données [17], reconnaissance d'images [18] ou encore en neurosciences pour modéliser l'activité cérébrale [19].

Bien qu'ils n'aient pas encore été envisagés dans ce contexte, les SNNs sont particulièrement adaptés aux applications de communication sans fil et de l'IoT grâce à leur efficacité énergétique [20], leur capacité à traiter les signaux en temps réel [21], et leur robustesse face aux environnements bruités [22]. Cet état de l'art dresse un premier tableau des SNNs (sans rentrer dans les expressions théoriques qui seront abordées dans les chapitres concernés), leurs modèles neuronaux, leurs mécanismes d'apprentissage, ainsi que leurs applications spécifiques dans les WuRs et les dispositifs IoT, tout en abordant les défis actuels et les perspectives futures dans ce domaine de recherche.

2.2 Mécanismes des neurones biologiques

Les neurones biologiques sont les cellules fondamentales du système nerveux, responsables de la transmission et du traitement des informations. Chaque neurone reçoit des signaux d'autres neurones via des connexions appelées synapses. On peut voir en fig. 2.1 une connexion neuronale. Celle-ci est dirigée du neurone i vers le neurone j. La synapse appelée s_{ji} envoie des impulsions depuis le neurone pré-synaptique i vers le neurone post-synaptique j.

Ces signaux modifient le potentiel électrique de la membrane du neurone j. Lorsqu'une stimulation suffisante est reçue, le neurone j atteint un seuil de potentiel et déclenche une impulsion électrique, appelée potentiel d'action. Les mécanismes que l'on retrouve dans la majorité des modèles, et qui sont essentiels à la compréhension du fonctionnement d'un SNN sont les suivants :

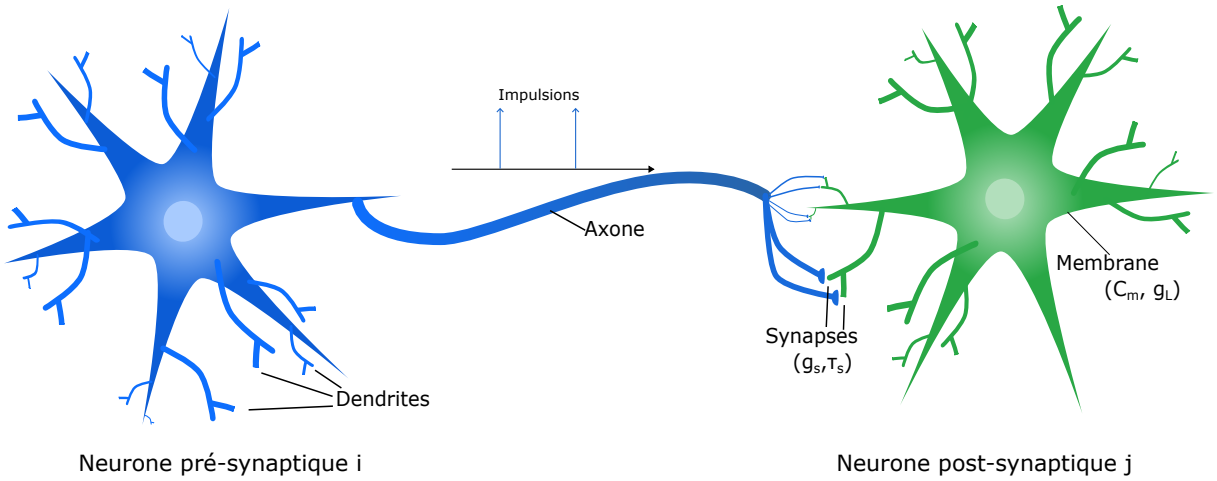


FIGURE 2.1 – Schéma de connexion neuronale

Intégration des signaux : Les neurones intègrent des signaux excitateurs et inhibiteurs provenant d'une ou plusieurs synapses. La somme de ces signaux détermine si le potentiel de membrane du neurone atteint le seuil nécessaire pour déclencher une impulsion. La tension est alors ramenée à la tension de repos.

Déclenchement d'impulsions : Lorsqu'un neurone voit sa tension atteindre un seuil, une impulsion est générée. Cette impulsion se propage le long de l'axone du neurone et est transmise aux neurones suivants par des synapses.

Fuite membranaire : La fuite membranaire est un phénomène où le potentiel d'un neurone tend à revenir progressivement à son potentiel de repos en l'absence de stimulation, en raison de la perméabilité de la membrane neuronale aux ions.

Période réfractaire : La période réfractaire est la durée suivant une impulsion pendant laquelle un neurone est temporairement incapable de déclencher une nouvelle impulsion, empêchant ainsi une sur-stimulation.

Poids synaptique : Les synapses pondèrent et propagent les impulsions reçues depuis la sortie du neurone pré-synaptique (le neurone précédent la synapse considérée) jusqu'au neurone post-synaptique (celui qui se situe après cette synapse). L'incrément de tension engendré par la réception d'une impulsion correspond au poids synaptique de la synapse ayant transmis l'impulsion. On dit que les synapses sont excitatrices ou inhibitrices selon que leur poids synaptique est positif ou négatif respectivement.

Délai synaptique : Le délai synaptique est le temps nécessaire pour qu'une impulsion soit propagée d'un neurone présynaptique à un neurone postsynaptique via une synapse. Dans les SNNs, le délai synaptique est un paramètre crucial qui modélise le temps de transmission des informations entre les neurones, influençant ainsi le traitement temporel des signaux et la synchronisation des réponses neuronales [23].

2.3 Modèles de neurones à impulsions

La modélisation des neurones à impulsions est une composante essentielle des SNNs. Les modèles de neurones dans les SNNs varient largement en termes de complexité computationnelle et de précision biologique. Cette diversité permet aux chercheurs et développeurs de choisir des modèles en fonction des phénomènes neuronaux qu'ils jugent essentiels pour leur application, tout en simplifiant ou en éludant ceux qui ont moins d'impact.

2.3.1 Modèles simples : prise en compte des phénomènes essentiels

Les modèles neuronaux simples, comme le modèle IF, se concentrent sur les aspects fondamentaux de l'intégration et de la génération d'impulsions sans inclure de mécanismes supplémentaires de biologie complexe. Ce modèle fonctionne en intégrant les signaux d'entrée jusqu'à ce que le potentiel de membrane atteigne un seuil, moment où une impulsion est déclenchée. Ensuite, le potentiel est réinitialisé à une valeur de repos. Ce modèle est computationnellement efficace, mais il néglige certains phénomènes biologiques tels que la fuite membranaire ou la variabilité de la période réfractaire, ce qui le rend moins réaliste [24].

2.3.2 Modèles intermédiaires : ajout de mécanismes biologiques spécifiques

Les modèles intermédiaires, comme le Leaky Integrate and Fire (LIF) et le Saturating Leaky Integrate-and-Fire (SLIF), introduisent une complexité supplémentaire pour mieux représenter les phénomènes biologiques tout en maintenant une efficacité computationnelle acceptable.

Leaky Integrate-and-Fire (LIF) : Le modèle LIF inclut un mécanisme de fuite membranaire, qui modélise la tendance naturelle du potentiel membranaire à revenir à son état de repos en l'absence de stimulation continue. Ce modèle est plus réaliste que le modèle IF car il tient compte de la dissipation naturelle du potentiel, rendant le neurone moins sensible aux impulsions isolées et plus apte à détecter des signaux pertinents dans un environnement bruité [25].

Saturating Leaky Integrate-and-Fire (SLIF) : Le modèle SLIF va encore plus loin en intégrant la saturation de la conductance synaptique, un phénomène qui limite l'effet des impulsions consécutives. Cela empêche une réponse excessive et améliore la capacité du neurone à discriminer des signaux dans des conditions de bruit élevé. Ce modèle est particulièrement utile dans les applications où une grande précision est nécessaire, tout en maintenant une complexité computationnelle gérable [26]. Ce modèle sera présenté plus en détail dans le chapitre 4.

2.3.3 Modèles complexes : représentation précise des phénomènes biologiques

Les modèles complexes, tels que le modèle Hodgkin-Huxley, offrent une représentation très détaillée des dynamiques neuronales en intégrant une variété de phénomènes biologiques, y compris les dynamiques des canaux ioniques et les phases du potentiel

d'action. Ces modèles sont utilisés dans des recherches nécessitant une simulation précise de l'activité neuronale, bien qu'ils soient beaucoup plus coûteux en termes de complexité computationnelle [27].

Hodgkin-Huxley : Ce modèle simule les processus électrochimiques à la base du potentiel d'action, incluant les variations de la conductance membranaire et la variabilité de la période réfractaire. Il est largement utilisé pour des études en neurosciences computationnelles où une compréhension détaillée du comportement neuronal est essentielle. Cependant, sa complexité le rend moins adapté aux applications nécessitant des calculs rapides ou économes en énergie.

Morris-Lecar : Le modèle de Morris-Lecar, développé pour simuler l'activité des neurones excitable et des cellules musculaires, décrit les processus électrochimiques de manière simplifiée en se concentrant sur deux variables clés : le potentiel de membrane et le courant ionique de potassium. Il est particulièrement efficace pour modéliser les transitions entre les états actifs et inactifs des cellules, ainsi que les oscillations neuronales. Ce modèle est souvent utilisé pour étudier des phénomènes comme les bifurcations et les oscillations de type spiking ou bursting. Bien que moins complexe que le modèle de Hodgkin-Huxley, il parvient à capturer des aspects importants des dynamiques neuronales avec une moindre complexité computationnelle, le rendant ainsi adapté aux simulations qui nécessitent un compromis entre précision et efficacité de calcul.

2.3.4 Choix de modèles selon les besoins de l'application

Le choix du modèle neuronal approprié dans les SNNs implique un compromis entre la complexité computationnelle et la précision biologique requise. Les modèles plus complexes, comme le Hodgkin-Huxley, sont utilisés pour des simulations détaillées et des recherches nécessitant une représentation fidèle des dynamiques neuronales, malgré un coût computationnel plus élevé. Les modèles plus simples comme l'IF ou le LIF sont souvent privilégiés dans des applications nécessitant une exécution rapide et efficace, comme ce dont nous avons besoin pour les Wake-Up Radios et les dispositifs IoT, où les ressources sont limitées. Nous allons donc favoriser ce type de modèles.

2.4 Codage de l'information dans les SNNs

Le codage de l'information est une dimension cruciale dans les SNNs, car il détermine la manière dont les neurones représentent et traitent les stimuli externes. Plusieurs schémas de codage ont été développés pour les SNNs, chacun ayant ses avantages et ses inconvénients en termes de précision, d'efficacité énergétique, et de complexité computationnelle. Les trois principaux types de codage utilisés dans les SNNs sont le codage par taux (rate coding), le codage temporel (temporal coding), et le codage par population.

2.4.1 Codage par taux (Rate coding)

Le codage par taux repose sur l'utilisation de la fréquence des impulsions pour représenter l'intensité d'un stimulus. Plus le signal représenté est fort, plus la fréquence des impulsions est élevée. Ce type de codage est simple à implémenter et largement utilisé dans les applications où une représentation approximative de l'intensité du stimulus est

suffisante. Cependant, il est moins efficace pour le traitement rapide de l'information, car il nécessite des périodes d'intégration plus longues pour estimer la fréquence d'impulsion avec précision [28]. Le codage par taux est couramment utilisé dans des applications de traitement de signaux où la consommation d'énergie et la rapidité de réponse ne sont pas des priorités comme la classification d'image [18]

2.4.2 Codage temporel (Temporal coding)

Le codage temporel utilise le timing précis des impulsions pour coder l'information. Contrairement au codage par taux, où l'information est représentée par la fréquence des impulsions, le codage temporel se concentre sur le moment exact des impulsions par rapport à un événement de référence ou à d'autres impulsions. Cela permet une transmission d'information plus rapide et plus efficace en termes d'énergie, car moins d'impulsions sont nécessaires pour représenter une information donnée. Le codage temporel est particulièrement avantageux dans des applications où le timing précis est essentiel, comme dans les systèmes de détection en temps réel et les réseaux de neurones sensoriels [21, 22].

2.4.3 Codage par population

Le codage par population implique l'utilisation d'un groupe de neurones pour représenter une information. Chaque neurone dans la population est sensible à un aspect spécifique du stimulus, et l'information globale est représentée par l'activité combinée de la population entière. Ce type de codage est utile pour des tâches de reconnaissance de motifs complexes, de classification et de prise de décision, car il offre une robustesse accrue contre le bruit et les erreurs individuelles des neurones. En outre, le codage par population permet une représentation plus riche et plus nuancée des stimuli, car il exploite la redondance et la diversité des réponses neuronales [29].

2.4.4 Conclusion

Le choix du schéma de codage dans les SNNs dépend des spécificités de l'application telles que la précision, l'efficacité énergétique et la rapidité de réponse. Le codage par taux, bien que simple, est moins adapté aux applications nécessitant une détection rapide et une consommation d'énergie minimale. En revanche, le codage temporel et le codage par population offrent des alternatives plus sophistiquées qui améliorent la vitesse de traitement et la robustesse, ce qui est particulièrement bénéfique pour des applications en temps réel et des environnements bruités. Dans cette thèse, nous utiliserons donc un codage temporel.

2.5 Apprentissage dans les SNNs

L'apprentissage est un aspect fondamental des SNNs, permettant aux réseaux de s'adapter à des tâches spécifiques en ajustant dynamiquement les poids synaptiques en fonction des stimuli et des expériences passées. Plusieurs mécanismes d'apprentissage ont été développés pour les SNNs, s'inspirant souvent de principes biologiquement plausibles. Parmi ces mécanismes, on trouve principalement la Spike-Timing-Dependent Plasticity (STDP), l'apprentissage supervisé, et l'apprentissage par renforcement.

2.5.1 Spike-Timing-Dependent Plasticity

La STDP est une règle d'apprentissage qui ajuste la force des connexions synaptiques en fonction de la synchronisation des impulsions entre les neurones pré- et postsynaptiques. Ce mécanisme repose sur le principe que si une impulsion du neurone présynaptique précède de peu une impulsion du neurone postsynaptique, la connexion synaptique est renforcée. Inversement, si l'impulsion postsynaptique précède l'impulsion présynaptique, la connexion est affaiblie. Cette plasticité synaptique temporelle permet aux SNNs de s'adapter aux motifs temporels des signaux d'entrée, optimisant ainsi leur capacité de reconnaissance et de classification des motifs [30, 31]. En favorisant les connexions synaptiques qui respectent des séquences temporelles pertinentes, la STDP améliore la précision des réponses neuronales en renforçant la détection des impulsions associées à des événements significatifs, tout en réduisant la sensibilité aux impulsions aléatoires ou bruitées.

2.5.2 Apprentissage supervisé

L'apprentissage supervisé dans les SNNs implique l'utilisation de données étiquetées pour entraîner le réseau à effectuer des tâches spécifiques, comme la classification ou la reconnaissance de motifs. Bien que l'application directe de la rétropropagation d'erreur soit difficile en raison de la nature discrète des impulsions, des adaptations de cette méthode ont été développées pour les SNNs. Par exemple, des techniques de rétropropagation du gradient approximée ou la rétropropagation à travers le temps (Backpropagation Through Time) peuvent être utilisées pour ajuster les poids synaptiques en fonction de l'écart entre les sorties attendues et celles générées par le réseau [32]. Ces approches permettent d'entraîner les SNNs à effectuer des tâches complexes tout en tirant parti de leurs avantages en termes de traitement temporel et d'efficacité énergétique. Cependant, il est important de noter que ces méthodes d'apprentissage ne sont pas biologiquement inspirées et ont tendance à consommer plus d'énergie par rapport à la STDP, en raison des calculs intensifs de gradient et des mises à jour des poids à chaque pas de temps.

2.5.3 Apprentissage par renforcement

L'apprentissage par renforcement dans les SNNs repose sur l'idée que les neurones peuvent apprendre à améliorer leurs performances en recevant des récompenses ou des pénalités en fonction de leurs actions. Dans ce cadre, les SNNs ajustent leurs connexions synaptiques pour maximiser une fonction de récompense. L'apprentissage par renforcement est utilisé dans des applications telles que la navigation autonome de robots [33], la gestion de l'énergie dans les réseaux de capteurs [34], et d'autres tâches nécessitant une prise de décision adaptative [35]. Bien que puissant pour les tâches adaptatives, l'apprentissage par renforcement n'est pas directement basé sur des mécanismes biologiques et peut demander une consommation d'énergie plus élevée, en raison de la complexité des calculs de récompense et des ajustements synaptiques associés.

2.5.4 Conclusion

Les différents mécanismes d'apprentissage dans les SNNs, allant de la STDP aux méthodes supervisées et par renforcement, offrent une grande flexibilité pour adapter les réseaux à diverses tâches et environnements. Chaque méthode présente des avantages

spécifiques en termes de capacité d'adaptation, de complexité computationnelle, et de pertinence pour des applications réelles. Il convient de noter que, bien que les méthodes d'apprentissage supervisé et par renforcement puissent améliorer les performances des SNNs, elles consomment généralement plus d'énergie que la STDP en raison de leur manque de fondements bio-inspirés et de la complexité des calculs nécessaires. La capacité des SNNs à apprendre de manière bio-inspirée et à traiter des informations temporelles complexes en fait néanmoins une technologie prometteuse pour de nombreuses applications dans les domaines de la reconnaissance de motifs, de la détection de séquences, et de l'automatisation intelligente. L'apprentissage des poids synaptiques était en dehors du scope de ma thèse. Toutefois, nous avons encadré un étudiant sur un sujet d'apprentissage de reconnaissance de signature temporelle qui a donné lieu à une soumission d'article dans une conférence.

2.6 Outils de simulation

Dans le but de simuler le fonctionnement de ces réseaux de neurones à impulsions, il a fallu choisir une solution de simulation. Parmi les plus utilisées dans la littérature on retrouve certaines librairies MATLAB, et d'autres en Python comme NEST, SNNTorch et Brian2 [36]. Nous avons privilégié Python car c'est un langage beaucoup plus utilisé et disponible partout. Les bibliothèques les plus utilisées sont les deux dernières, SNNTorch étant plus utilisée pour modéliser des réseaux à partir de modèle neuronaux pré-établis, et Brian2 plutôt utilisé pour paramétrer soit même ses neurones. Il a donc été choisi d'utiliser cette dernière librairie pour réaliser les simulations.

Brian2 est également beaucoup utilisé en neurosciences computationnelles, ce qui a grandement facilité la collaboration avec un expert du domaine, Romain CAZE. Le concepteur de cette librairie, Marcel Stimberg, a notamment mis à disposition une série de tutoriels basés sur Jupyter Notebook afin de faciliter la prise en main, ainsi que des scripts utilisant Brian2 issus d'articles de neuroscience. De plus, Brian2 possède une communauté sur son forum qui collabore activement.

Brian2 permet de simuler le comportement d'un neurone à travers toutes les équations données par son utilisateur, en calculant à chaque pas de temps, paramétrable, l'évolution de chaque variable de chaque entité, neurone ou synapse. Brian2 possède également quelques fonctions permettant d'utiliser des modèles d'entrées, de neurones ou de synapses pré-établis. De plus, il est aussi possible de simuler les interactions entre chaque neurones dues à la position de ceux-ci dans l'espace, mais cela sort du cadre de notre étude.

Brian2 est un très bon outil pour visualiser l'évolution d'un réseau de neurones à impulsions mais il simule beaucoup d'éléments très bas niveau de ces neurones et s'avère être assez lent lorsque l'on veut simuler un très grand nombre de neurones et de synapses. Lorsque nous nous sommes retrouvés confrontés à cette situation, nous avons fait nos simulations en calculant directement l'évolution des équations différentielles du modèle concerné grâce à la fonction de résolution d'équation différentielles de la librairie Scipy.

2.7 Applications des SNNs dans les WuRs et l'IoT

Les réseaux de neurones à impulsions (SNN) laissent percevoir un potentiel considérable pour diverses applications dans les systèmes de communication sans fil et l'IoT, en raison de leur efficacité énergétique, de leur capacité à traiter des signaux en temps réel,

et de leur robustesse face aux environnements bruités. Nous présentons maintenant ces applications.

2.7.1 Wake-Up Radios

Les Wake-Up Radios (WuR) sont des dispositifs conçus pour rester en veille pendant de longues périodes avec une consommation énergétique extrêmement faible. Leur rôle principal est de surveiller les canaux de communication pour détecter un signal d'activation spécifique, appelé Wake-up Beacon, et de réveiller le système principal seulement lorsque ce signal est reçu. Cela permet d'éviter que l'ensemble du système ne soit constamment en fonctionnement, ce qui réduirait considérablement la durée de vie de la batterie.

L'un des défis majeurs des WuRs classiques est de minimiser les fausses alarmes tout en maintenant une sensibilité élevée aux signaux pertinents. C'est ici que les réseaux de neurones à impulsions (SNN) entrent en jeu. Grâce à leur capacité à traiter les informations de manière événementielle et à répondre uniquement à des séquences d'impulsions spécifiques, les SNNs sont particulièrement adaptés à cette tâche. Ils peuvent être conçus pour réagir uniquement lorsqu'une séquence précise est détectée, ce qui réduit considérablement le risque de fausses alarmes et améliore la fiabilité des communications.

En utilisant des SNNs pour surveiller continuellement les canaux de communication, les WuRs restent en veille jusqu'à ce qu'une séquence d'impulsions définie soit reçue. Mais il faut éviter que le système ne réagisse à des interférences ou à des bruits aléatoires dans le canal, afin de ne pas consommer inutilement l'énergie. Cela est crucial pour les dispositifs IoT alimentés par batterie, car leur autonomie est souvent limitée par la consommation électrique des modules de communications. Les WuRs basées sur des SNNs offrent une approche plus efficace pour maximiser la durée de vie des batteries, particulièrement dans des environnements où les communications sont sporadiques mais où la réactivité reste essentielle [37, 38]. De plus, les SNNs sont capables de traiter des signaux en temps réel sans avoir besoin de recourir à des processeurs classiques énergivores, ce qui les rend encore plus adaptés à des scénarios à faible consommation.

2.7.2 Dispositifs IoT et détection de motifs

Dans les dispositifs IoT, qui sont souvent déployés dans des environnements où les contraintes énergétiques sont importantes, les SNNs se révèlent particulièrement efficaces pour des tâches telles que la reconnaissance de motifs, la détection d'événements, et la classification de données en temps réel [32]. Les capacités d'analyse temporelle des SNNs, associées à leur faible consommation d'énergie, en font des solutions idéales pour ces dispositifs, où chaque bit d'énergie compte.

Les SNNs sont capables de reconnaître des motifs complexes dans les données captées par les capteurs IoT, en particulier dans des environnements dynamiques où la détection d'événements spécifiques est cruciale, comme dans les systèmes de sécurité ou de surveillance industrielle [17]. Par exemple, dans un système de surveillance vidéo basé sur des dispositifs de l'IoT, les SNNs peuvent être utilisés pour analyser les flux vidéo en continu et réagir uniquement lorsque des mouvements ou comportements anormaux sont détectés, sans qu'il soit nécessaire de maintenir l'ensemble du système actif en permanence [39]. Cette approche permet de considérablement réduire la consommation d'énergie tout en assurant une surveillance en temps réel.

Dans la détection d'événements, les SNNs sont particulièrement efficaces grâce à leur

capacité à intégrer les informations temporelles. Ils peuvent réagir à des séquences temporelles précises, comme une série de détections de mouvement dans un environnement, ou des variations dans les données de capteurs, tout en consommant une énergie minimale. Contrairement aux réseaux de neurones traditionnels, les SNNs fonctionnent de manière événementielle, c'est-à-dire qu'ils n'actualisent leur état que lorsqu'un événement (ou un spike) se produit, ce qui évite les calculs inutiles.

De plus, les SNNs peuvent être utilisés pour la classification de données en temps réel, une tâche courante dans les environnements IoT où des capteurs multiples fournissent continuellement des données. Les SNNs permettent d'analyser ces données rapidement et avec une faible consommation d'énergie. Par exemple, dans un réseau de capteurs environnementaux, les SNNs peuvent être employés pour classifier les données de température, d'humidité ou de pollution, et envoyer des alertes ou prendre des décisions automatisées uniquement lorsque des seuils critiques sont dépassés [32, 20]. L'avantage majeur des SNNs dans ces applications réside dans leur capacité à traiter les données à bas niveau d'énergie tout en assurant une classification et une détection rapides.

En conclusion, les SNNs jouent un rôle clé dans l'optimisation énergétique des dispositifs IoT, en permettant des analyses complexes de motifs, tout en maintenant une consommation énergétique extrêmement basse. Cette capacité à détecter des événements critiques et à classer les données en temps réel sans compromettre la durée de vie des batteries rend les SNNs particulièrement attractifs pour les applications IoT dans divers secteurs.

2.7.3 Challenges

Les SNNs offrent des solutions prometteuses pour les WuRs et les applications IoT grâce à leur faible consommation d'énergie, leur traitement en temps réel, et leur robustesse face aux signaux bruités. En exploitant des mécanismes de détection basés sur des motifs temporels spécifiques, les SNNs permettent de développer des systèmes de communication et de détection plus intelligents et économes en énergie, adaptés aux environnements contraints en énergie typiques des dispositifs IoT. À mesure que la technologie des SNNs et des puces neuromorphiques continue de progresser, leur adoption dans les applications WuR et IoT pourrait s'étendre, offrant des améliorations significatives en termes de durée de vie des batteries. Nous essayons de mettre en oeuvre des SNNs complètement analogiques. Il existe cependant des circuits neuromorphiques numériques avec beaucoup de neurones et d'articles dans la littérature [23]. Mais dans ce cas la gestion du temps se fait par une horloge. Par contre très peu d'études ont été réalisées avec des circuits purement analogiques alors que c'est là que va résider réellement le gain en énergie. Mais cela implique que le temps ne se contrôle pas avec une horloge mais avec la propagation des signaux et la réaction des neurones, d'où le besoin des travaux présentés dans ce manuscrit.

2.8 Avantages et défis des SNNs dans les systèmes de communication sans fil

2.8.1 Avantages

Les SNNs offrent plusieurs avantages :

— **Efficacité Énergétique** : En consommant de l'énergie uniquement lors de la génération et de la transmission de spikes, les SNNs ont une puissance de seulement $100pW$, et sont plus économes en énergie que les ANNs traditionnels dont la puissance est plutôt de l'ordre de la centaine de μW ; ce qui est crucial pour les dispositifs à batterie limitée.
— **Traitement en Temps Réel** : La nature des SNNs permet un traitement des données asynchrone, facilitant une réponse rapide aux stimuli externes sans nécessiter de cycles de traitement continus.
— **Robustesse face au Bruit** : Les SNNs peuvent discriminer efficacement entre les signaux pertinents et le bruit de fond, améliorant ainsi la précision de la détection dans des environnements complexes.
— **Modélisation Biologique** : La capacité des SNNs à imiter les dynamiques neuronales biologiques ouvre la voie à des applications en neuroscience computationnelle et à l'interface cerveau-machine.

2.8.2 Défis

Malgré leurs avantages, les SNNs présentent plusieurs défis techniques :
— **Complexité de l'apprentissage** : La non linéarité induite par les impulsions rend l'application des méthodes d'apprentissage basées sur le gradient, comme la rétropropagation, difficile.
— **Optimisation des réseaux** : Concevoir des SNNs optimaux pour des tâches spécifiques nécessite une exploration approfondie des paramètres du réseau, y compris la topologie, les mécanismes d'apprentissage, et les techniques de codage.
— **Limitations matérielles** : Les implémentations matérielles des SNNs sur des plateformes neuromorphiques sont encore à un stade de développement précoce, avec des contraintes en termes de puissance de calcul et de capacité mémoire.
— **Déploiement à grande échelle** : Intégrer les SNNs dans des systèmes existants et les faire évoluer pour des applications à grande échelle restent des défis majeurs.
— **Utilisation de circuits analogiques** : L'utilisation de circuits purement analogiques permet d'économiser encore plus d'énergie qu'avec des circuits neuromorphiques actuels comme TrueNorth ou Loihi car la gestion du temps ne se fera plus grâce à une horloge, mais cela rajoute donc des contraintes.

2.9 Perspectives Futures

Les progrès récents dans le domaine des puces neuromorphiques, comme IBM TrueNorth et Intel Loihi, montrent le potentiel des SNNs pour des applications en temps réel et à faible consommation d'énergie. Ces puces sont capables de simuler des millions de neurones avec des exigences énergétiques réduites, offrant une base matérielle solide pour des déploiements à grande échelle des SNNs [40, 41].

Les recherches futures devraient se concentrer sur :
— **Amélioration des algorithmes d'apprentissage** : Les algorithmes d'apprentissage pour les SNNs évoluent encore. Actuellement, la rétropropagation à travers le temps (BPTT) est utilisée, mais elle reste énergivore et peu bio-inspirée. L'optimisation des règles locales comme la STDP, plus proche des mécanismes du cerveau, pourrait réduire la complexité computationnelle et améliorer l'efficacité énergétique. À l'avenir, des méthodes d'apprentissage locales et distribuées pourraient

permettre un traitement en temps réel sans recourir à des processeurs gourmands en énergie, ce qui serait particulièrement utile dans les applications IoT [31, 32].

— **Développement de nouvelles architectures matérielles** : Bien que des architectures neuromorphiques comme Intel Loihi et IBM TrueNorth existent déjà, elles nécessitent encore des améliorations en matière de scalabilité et d'efficacité énergétique. L'utilisation de matériaux avancés comme les memristors pourrait simuler plus fidèlement les neurones biologiques, tout en optimisant la consommation d'énergie. La miniaturisation des circuits neuromorphiques et la réduction des fuites d'énergie seront cruciales pour des applications mobiles ou IoT [42, 43].

— **Utilisation avec du machine learning classique** : Le couplage des SNNs avec des ANNs pourrait allier l'efficacité énergétique des SNNs au traitement de haute précision des réseaux de neurones classiques. Ces systèmes hybrides seraient idéaux pour des tâches complexes comme la classification d'images ou l'analyse en temps réel, en permettant une détection rapide des motifs par les SNNs, suivie d'une classification précise par les ANNs, tout en maintenant une consommation énergétique faible [44].

2.10 Conclusion

Les SNNs représentent une technologie prometteuse pour l'avenir des systèmes de communication sans fil et des dispositifs IoT. Leur capacité à traiter des signaux de manière efficace et économe en énergie, combinée à leur robustesse face aux environnements bruités, les rend idéaux pour une gamme d'applications allant des WuRs aux systèmes de surveillance de la santé. À mesure que les défis techniques seront surmontés, les SNNs ont le potentiel de jouer un rôle central dans le développement de nouvelles générations de dispositifs intelligents et connectés. C'est la raison pour laquelle cette thèse vise à combler le manque d'utilisation des SNNs dans le domaine de l'IoT en étudiant les modèles de neurones adaptés à ce contexte puis les topologies pertinentes et les séquences associées afin de réaliser un système de WuR le plus économe en énergie possible.

Chapitre 3

Réseau de neurones LIF pour Wake-Up Radio

3.1 Introduction

Notre cerveau traite l'information fournie par nos sens à l'aide de cellules spécialisées appelées neurones. Ces neurones communiquent grâce à des impulsions électriques appelées "spikes" envoyées aux autres neurones à travers des liaisons appelées synapses. Un neurone artificiel à impulsion fonctionne sur le même principe qu'un neurone présent dans le cerveau humain, c'est pourquoi il est dit bio-inspiré et que l'on parle de réseau neuromorphique pour qualifier les SNNs.

A l'instar d'un neurone du cerveau, un neurone à impulsion artificiel va venir intégrer dans sa tension de membrane les impulsions qu'il reçoit. Une fois sa tension de seuil atteinte, il va transmettre une impulsion aux neurones suivants. Les signaux traités doivent donc prendre la forme d'impulsions. Or, dans les transmissions radios, l'envoi d'impulsions occuperait une très grande bande de fréquence. La séquence sera donc transmise sur le canal en utilisant une modulation On-Off Keying (OOK). L'On-Off Keying est une technique de modulation qui consiste à allumer et éteindre la porteuse d'un signal pendant des durées déterminées, pour représenter l'envoi de 1 ou 0 binaires respectivement. La chaîne de transmission est alors réalisée grâce à l'émetteur et le récepteur.

Côté émetteur, l'information est transmise sous la forme d'un train d'impulsions électriques, qui se modélise mathématiquement comme une somme d'impulsions de Dirac aux instants d'émission :

$$y = \sum_{i=1}^{N} \delta(t - t_i) \tag{3.1}$$

avec t_i la position temporelle de l'impulsion i.

Côté récepteur, nous allons recevoir le même signal auquel va venir s'ajouter un bruit de canal modélisé par une bruit blanc :

$$y = \sum_{i=1}^{N} \delta(t - t_i) + n \tag{3.2}$$

Le principal avantage de l'OOK est qu'elle peut être démodulée grâce à un récepteur non-cohérent. Cela permet d'économiser de l'énergie en enlevant l'oscillateur. Le signal reçu OOK reçu est donc transformé en train d'impulsions. Lorsque la puissance du signal

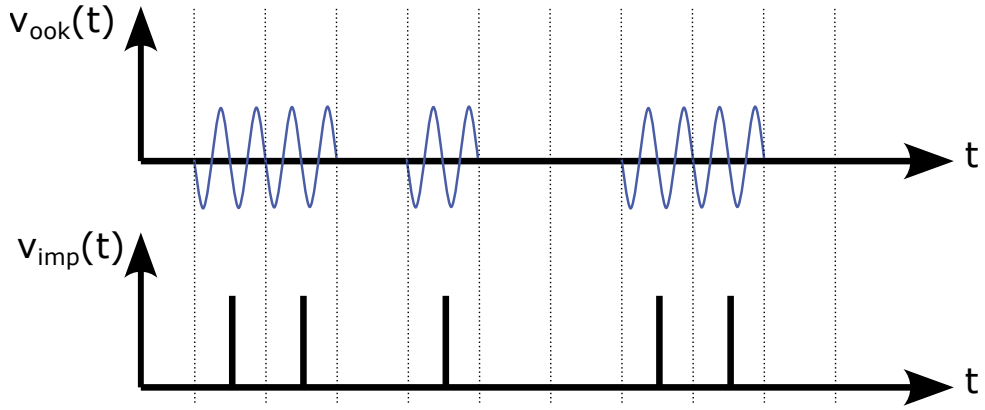


FIGURE 3.1 – Conversion d'un signal modulé OOK en un train d'impulsions

dépasse le seuil, cela génère une impulsion qui sera ensuite traitée par le SNN [45] comme illustré en fig. 3.1.

3.2 Modèle neuronaux

3.2.1 Modèle de neurone IF

Une fois le signal transformé en train d'impulsions, celui ci est transmis au neurone artificiel. Lors de la réception d'une impulsion, la tension membranaire du neurone $v(t)$ va subitement augmenter.

Le premier modèle, le plus simple, se nomme le IF, pour Integrate and Fire. Sa dynamique est de la forme :

$$\tau_m \frac{dv}{dt} = R_m I(t) \tag{3.3}$$

avec τ_m la constante de temps membranaire, R_m la résistance de membrane, et $I(t)$ le courant en entrée du neurone, directement proportionnel au train d'impulsions reçu.

On peut observer que, dans ce modèle, la dérivée de la tension membranaire est directement proportionnelle au courant d'entrée, qui est modélisé comme un peigne de Dirac. Cela signifie que chaque impulsion du courant provoque une variation instantanée de la tension. En conséquence, la forme de $v(t)$ dans le modèle IF prend la forme d'une somme d'échelons de Heaviside.

Lorsque le neurone reçoit une impulsion, sa tension membranaire augmente en conséquence. Une fois que la tension membranaire a atteint une valeur prédéfinie v_{th}, appelée tension de seuil, le neurone va envoyer une autre impulsion en sortie, et retourner rapidement vers sa valeur de repos v_{rest} comme on peut le voir en fig. 3.2. Dans cet exemple, nous avons envoyé une impulsion au neurone toutes les $5ms$, v_{rest} valait $0V$, et v_{th} est représentée par la ligne en pointillés et prend la valeur de $1V$. Ces valeurs sont très éloignées des valeurs biologiques, que l'on retrouvera dès la partie suivante, mais ont été choisies pour illustrer le modèle.

On intègre les impulsions reçues, donc l'évolution de la tension membranaire est nulle à l'exception des instants où l'on reçoit une impulsion. On a aussi un intégrateur sans perte, donc la valeur va augmenter à chaque impulsion reçue, et revenir d'un coup à sa valeur de repos une fois le seuil dépassé.

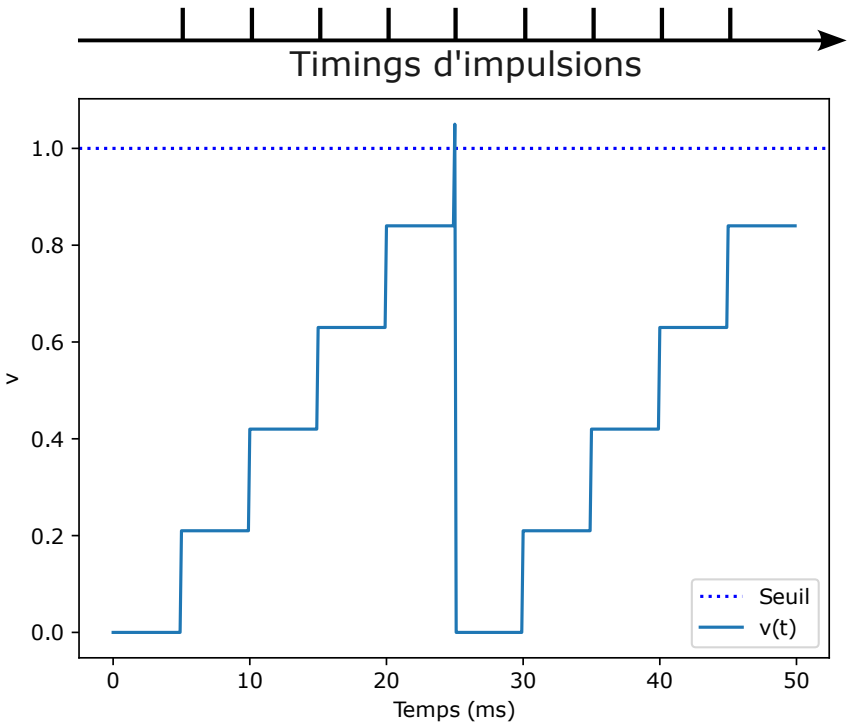


FIGURE 3.2 – Réponse d'un neurone IF à une série d'impulsions

3.2.2 Modèle de neurone LIF

Le neurone artificiel que nous considérerons dans ce chapitre a le même comportement, caractérisé par sa réponse à un train d'impulsions : le LIF, Leaky Integrate and Fire [46]. C'est l'un des modèles les plus utilisés. Ce modèle est une extension du modèle IF, que l'on va légèrement complexifier en prenant en compte un phénomène biologique appelé fuite de membrane. Ce phénomène se caractérise par une fuite de la tension membranaire qui ramène progressivement cette tension vers v_{rest} en l'absence de stimulus.

Son comportement est régi par l'équation :

$$C_m \frac{dv}{dt} = I(t) - g_L(v(t) - v_{rest}) \tag{3.4}$$

avec C_m la capacité de membrane, $v(t)$ la tension membranaire au temps t, g_L la conductance de membrane et v_{rest} la tension membranaire de repos avec $v_{rest} = -65mV$.

Le terme de droite est composé de deux parties correspondant aux effets de deux phénomènes différents. La partie $I(t)$ correspond à l'impact des impulsions entrantes sur la tension membranaire. Dans notre exemple, à l'instant d'une impulsion, le courant reçu de la part du neurone i, $I_i(t)$, est un dirac. $v(t)$ augmente donc instantanément. Cela correspond aux pics verts sur la fig. 3.3. C'est le phénomène qui a été vu précédemment pour le IF. On retrouve ensuite le phénomène de fuite, régit par g_L, qui va progressivement ramener la tension de membrane vers sa valeur de repos. Cela correspond à la partie bleue de la courbe de la fig. 3.3. De plus, une fois que le seuil est atteint, le neurone décharge, sa tension redescend instantanément vers sa valeur de repos, et le neurone transmet une impulsion à ses synapses de sortie pour les neurones suivants, visible sur la courbe $I_j(t)$ en fig. 3.3.

Toutefois, si ce seuil n'est pas atteint, le phénomène de fuite (Leak en anglais) va faire diminuer la tension au court du temps jusqu'à redescendre à v_{rest} (eq. 3.4).

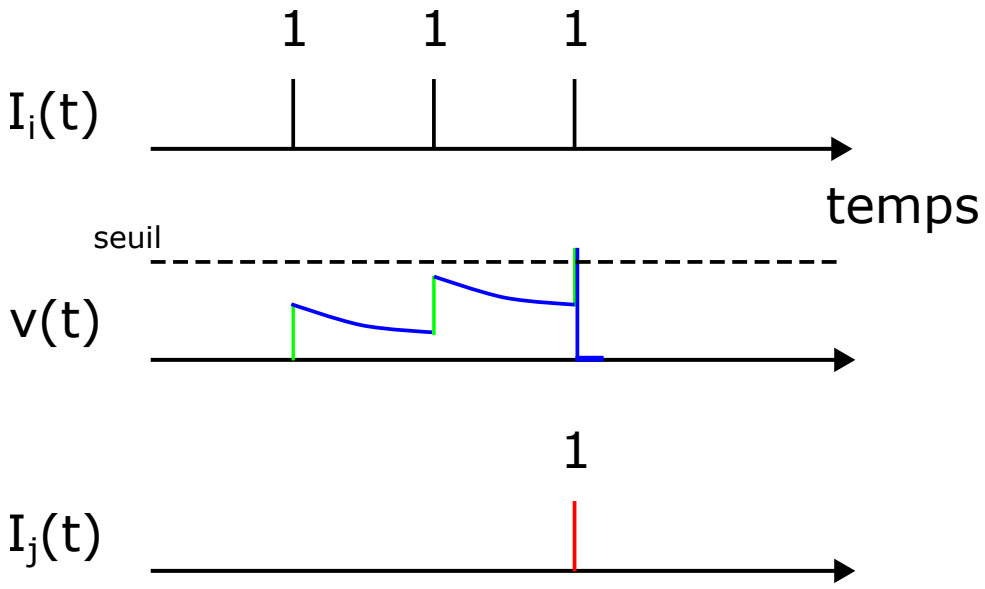


FIGURE 3.3 – Réponse d'un neurone LIF à une série d'impulsions

3.3 Co-design de la topologie et la signature

Notre premier objectif est de réaliser un réseau capable de discriminer une signature ciblée lors de la réception d'un signal sous la forme d'un train d'impulsions de la forme :

$$s^{(k)}(t) = \sum_k \delta(t - t_k) \tag{3.5}$$

où t_k représente les instants d'envoi des impulsions.

La signature (ou code) ciblée sera construite sous la forme d'un mot binaire à l symboles. Celle-ci sera alors transmise par OOK, sous la forme d'un signal créneléé dans lequel chaque créneau temporel correspond à l'envoi d'un bit. Enfin, ce signal sera transformé en train d'impulsions, lui aussi créneléé. Durant chaque créneau, on aura soit la présence d'une unique impulsion, si la porteuse était allumée, soit un silence, si elle était éteinte.

Pour cela, une fois que ce train est reçu, nous devons évaluer au même instant l bits successifs. Chaque fois que la signature est reconnue dans le flux de données reçu, on envoie une impulsion comme on peut le voir en fig. 3.5.

Si on reçoit une signature sous forme de signal série, on va venir incrémenter la tension membranaire pour chaque impulsion reçue. Si on a suffisamment de reçu de 1, le neurone va atteindre son seuil, et décharger. Sinon, il n'atteindra jamais le seuil. Cependant, il ne nous est pas possible d'évaluer la position de chacune des impulsions. En effet, le neurone est seulement capable de compter le nombre d'impulsions reçues. Aussi, si l'on reçoit plus d'impulsions que le nombre de 1 dans la séquence ciblée, le neurone atteindra aussi son seuil et désignera la séquence comme étant celle que l'on ciblait. Nous avons représenté en fig. 3.4 l'évolution de la tension membranaire lors de l'envoi de deux train d'impulsions différents au même neurone. Dans le premier cas on envoie une séquence binaire à 8 bits 11101011. Nous avons ensuite envoyé une deuxième séquence 11010111 et remarqué que l'amplitude maximale était supérieure à celle atteinte avec la première séquence. Il est ici impossible que le neurone décharge pour la première séquence et pas avec la seconde pour une séquence série.

Pour remédier à ce problème, les trains d'impulsions série doivent être parallélisés afin

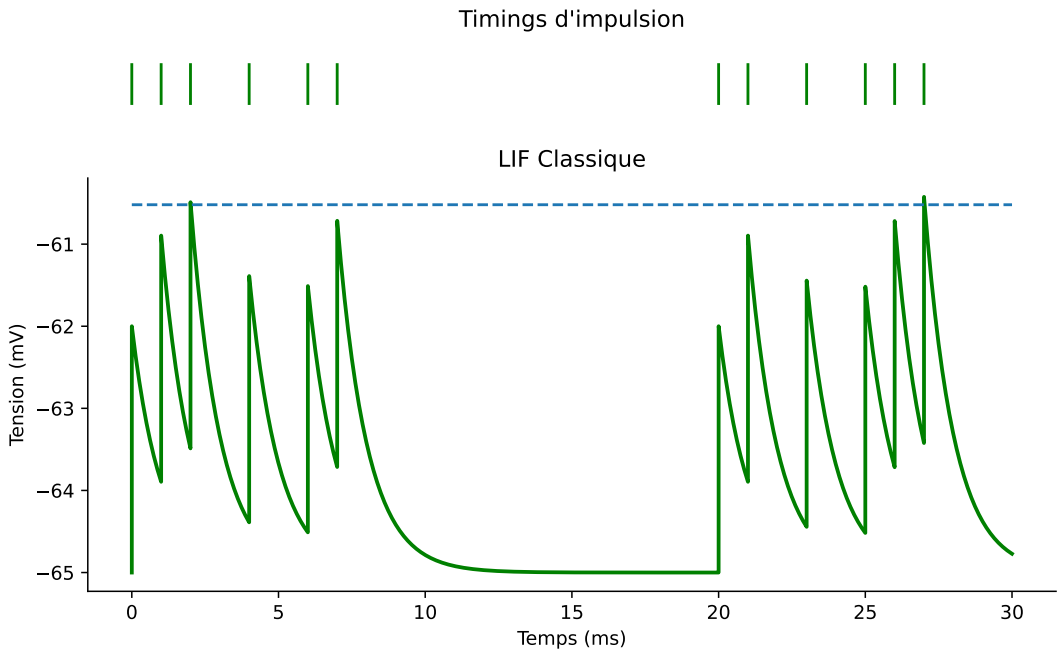


FIGURE 3.4 – Réaction d'un neurone LIF recevant la séquence cible et une séquence différente

de pouvoir être traités simultanément.

La solution que nous proposons consiste en un réseau composé de trois couches de neurones, connectés par des synapses, avec une topologie strictement en feedforward.

La première couche (en bleu) comprend un unique neurone dont le but est de démoduler le signal d'entrée OOK pour en faire un train d'impulsions (eq. 3.5) puis l'envoyer à la deuxième couche.

Dans le cas où nous n'avons pas de bruit ni de canal, le signal est parfaitement représenté. Dans le cas contraire, la démodulation peut amener à recevoir un flux erroné, dans lequel on aurait des impulsions en plus, en moins, ou les deux.

La seconde couche (en vert) contient $l-1$ neurones, qui renvoient exactement le train d'impulsions reçu. Mais les synapses entre la première et la deuxième couche ont pour but de retarder le train en entrée, de telle manière que le neurone i de la deuxième couche reçoive le signal décalé $s^{(k)}(t - i \times t_{bit})$ avec t_{bit} le temps entre deux bits du signal d'entrée. Pour une séquence de l bits, après un temps $l \times t_{bit}$ le neurone de la première couche et les $l-1$ neurones de la seconde vont simultanément lire les l derniers bits reçus.

Enfin, la couche de sortie (en rouge) contient autant de neurones qu'il y a de séquences que l'on veut reconnaître ($m = 1$ dans la fig. 3.5), et chacun est connecté aux deux premières couches via l synapses. De cette manière, un récepteur peut être réveillé par plusieurs signatures, par exemple son adresse dédiée et celle d'un groupe bien particulier de noeuds.

Afin de reconnaître des séquences différentes, les synapses entre les deux premières couches et la troisième sont excitatrices ou inhibitrices selon si la séquence à détecter a un 1 ou un 0 à cette position. C'est à dire qu'elles vont avoir une contribution positive ou négative respectivement sur la tension du neurone de sortie, comme représenté en figure 3.5.

Si un des neurones (in, $h1$, $h2$ ou $h3$) doit lire un 1, il est connecté à une synapse

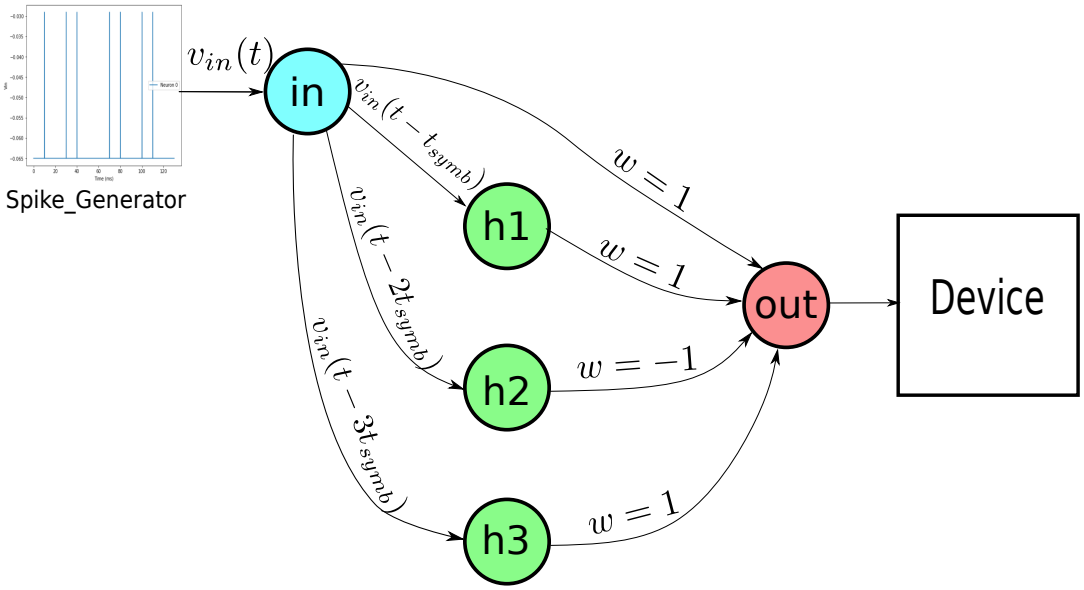


FIGURE 3.5 – Topologie de notre réseau pour une longueur de séquence $l = 4$

excitatrice qui va envoyer une contribution positive au neurone de sortie *out*. A l'inverse, s'il est sensé lire un 0, et reçoit une impulsion, la synapse inhibitrice va envoyer une contribution négative au neurone de sortie pour diminuer la tension et prévenir l'envoi d'une impulsion par le neurone de sortie.

Le seuil respectif de chaque neurone de sortie est déterminé de manière à ce qu'il réagisse lorsqu'il reçoit en même temps, à une petite tolérance près, autant d'impulsions excitatrices qu'il y a de 1 dans la séquence, et aucune inhibitrice.

A partir de la taille de la séquence l et le nombre que l'on veut en reconnaître m, on va créer notre réseau de $l+m$ neurones LIF comme décrit plus tôt. Notre réseau comprendra donc $n_{neu} = l + m$ neurones, et $n_{syn} = l(m + 1) - 1$ synapses pour les relier.

3.4 Résultats

Afin de voir si les performances de notre réseau sont comparables avec celles des technologies actuelles, nous évaluons les performances pour trois types de modèles de séquences d'activation.

Dans la première, appelée la méthode *Burst*, les séquences sont envoyées de manière continue, reproduisant le cas dans lequel il y a en permanence un noeud à réveiller. C'est à dire que dès qu'une séquence se termine, une autre est envoyée directement. La seconde méthode est appelée *Sporadic*, et correspond au cas où un noeud est rarement réveillé. Dans ce cas, les séquences sont envoyées de manière éparse, séparées par une période de silence, au moins aussi longue qu'une séquence.

Ces deux premières méthodes sont asynchrones. Mais nous voulons aussi utiliser une méthode d'envoi synchrone. Pour la dernière méthode, nommée *Synchro*, nous partons de

l'hypothèse selon laquelle les séquences sont périodiquement programmées, ce qui permet au récepteur de savoir à l'avance quand une séquence est censée démarrer et se terminer, si elle a bien été envoyée, grâce à la synchronisation.

De plus, de manière à se placer dans un cas réaliste, on considère un bruit Additive White Gaussian Noise (AWGN) (de l'anglais Additive White Gaussian Noise, ou bruit blanc additif gaussien en français, caractérisé par son écart-type σ), ajouté à l'entrée du premier neurone in, comme représenté en fig. 3.5. Le bruit peut amener à l'apparition d'erreurs dans le train d'impulsions généré, ce qui peut se traduire par des Fausse Alarmes (FAs) ou MisDetections (MDs), des détections manquées.

La probabilité de MD est donnée par $\frac{FN}{FN+TP}$, tandis que celle de FA correspond à $\frac{FP}{FP+TN}$; où :

- False Negatives (FN) (Faux Négatifs), lorsqu'un code était présent mais n'est pas détecté.
- False Positives (FP) sont les Faux Positifs, lorsque l'on détecte le code alors qu'il n'était pas présent.
- True Positives (TP) représente les vrais positifs, qui surviennent lorsque l'on détecte correctement qu'il y avait la séquence à reconnaître.
- True Negatives (TN) représente les vrais négatifs, qui surviennent lorsque le neurone ne réagit pas et qu'il n'y avait pas la séquence à reconnaître.

Nous avons réalisé ces simulations à l'aide de la librairie Brian2.

3.4.1 Codes totalement aléatoires

Notre première étape a été de considérer que n'importe quelle séquence binaire de l bits pouvait être utilisée comme un code approprié.

Dans un premier temps, nous avons développé l'expression théorique de la probabilité de MD et FA pour la méthode Synchro. Pour cela nous avons évalué la probabilité qu'un bit envoyé, un 0 ou un 1, a d'être inversé sur le neurone d'entrée. Ce neurone agit comme un récepteur OOK qui traduit la tension d'entrée en bit. Si la tension reçue est supérieure à un seuil, on considère un 1, sinon on considère un 0. Le potentiel de repos de ce neurone est à $-65mV$ et la réception d'une impulsion va faire monter son potentiel membranaire de $60mV$, pour lui faire atteindre $-5mV$. Le seuil doit donc être placé entre -65 et $-5mV$. Dans ce scénario, le seuil est placé à $-35mV$, qui est le milieu. La condition pour obtenir une erreur sur un bit est $n > 30mV$ si le bit transmis était un 0 et $n < -30mV$ si c'était un 1 :

$$P(E|0) = P(n > (v_{th} - v_{rest})) \tag{3.6}$$

$$P(E|1) = P(n < (v_{th} - v_{spike})) \tag{3.7}$$

Le bruit n suit une loi normale centrée $\mathcal{N}(0, \sigma)$. La loi normale étant une fonction symétrique, $P(n > 30) = P(n < -30)$, et on a $P(E|0) = P(E|1) = P(E)$. De plus, la probabilité d'obtenir une MD est la probabilité qu'au moins un des bits ait été retourné. On a donc :

$$P(MD) = 1 - (1 - P(E))^l \tag{3.8}$$

De la même manière, si une signature est la même que celle que l'on recherche, à k bits près, et que ces k bits exactement sont inversés, alors on a une la présence d'une FA.

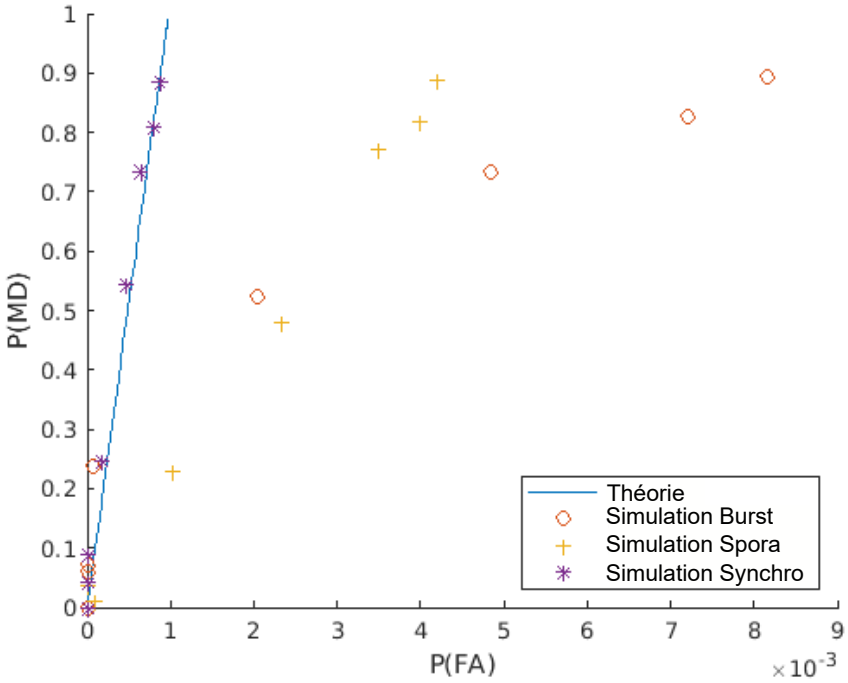


FIGURE 3.6 – Impact du bruit sur les probabilités de FA et MD pour un code totalement aléatoire de 10 bits

La probabilité de FA est donc :

$$P(FA|N) = \frac{P(FA \cap N)}{P(N)} \tag{3.9}$$

avec

$$P(FA \cap N) = \sum_{k=1}^{l} \left(\left(\frac{1}{2}\right)^{l} \binom{l}{k} P(E)^{k} (1 - P(E))^{l-k} \right) \tag{3.10}$$

et

$$P(N) = 1 - \left(\frac{1}{2}\right)^{l} \tag{3.11}$$

Avec FA l'évènement "Le réseau renvoie une fausse alarme", et N "La signature émise est une mauvaise séquence", c'est à dire que la séquence émise n'est pas celle que l'on essaie de reconnaître.

Dans cette première configuration, nous présentons les performances obtenues en simulation avec σ variant de 0 à 60 pour $l = 10$ en fig. 3.6. Nous pouvons tout d'abord noter que la probabilité de MD est plus importante que celle de FA. C'est dû au fait qu'une MD apparaît lorsqu'il y a au moins une erreur dans la séquence. Alors que pour une FA, il faut avoir une séquence initialement différente, et que tous les bits initialement différents soient inversés à cause du bruit, alors que ceux initialement identiques ne sont pas inversés.

De plus, on peut observer que ces performances sont bien meilleures en termes de FA et MD dans le cas Synchro que dans les autres scénarios. Pour la méthode Burst, cela est dû à la fenêtre glissante qui peut amener des erreurs à cause de la combinaison de deux séquence consécutives. Pour le cas sporadique, cela est dû au fait que le réseau de neurones ne peut pas différencier une absence d'impulsion due à la non transmission de séquence, d'une absence d'impulsion due à l'envoi d'un 0 dans la transmission OOK de la séquence. Les cas de FA sont donc plus enclins à apparaître dans le cas Sporadic quand une séquence contient un 0 au début ou à la fin. Dans ces cas, la génération d'une impulsion en sortie de réseau se fait au milieu de la transmission d'une séquence. S'il était possible (comme avec la méthode Synchro) de ne s'intéresser qu'aux impulsions générées à la fin des séquences, ces alarmes indésirables n'apparaîtraient pas comme on peut le voir en comparant la courbe Synchro et la courbe théorique, qui ne prend pas en compte la méthode d'envoi. Néanmoins, la synchronisation consomme de l'énergie, et n'est pas réalisable si l'on veut rester un système à très faible puissance [47]. C'est pourquoi nous proposons dans ce travail de forcer les codes à démarrer et terminer par un 1. En faisant cela, nous créons une forme de synchronisation (avec les 1 aux extrémités servant de délimiteurs temporels).

3.4.2 Code de synchronisation

En prenant en compte ce code amélioré, c'est à dire en démarrant et terminant par un 1, il nous faut adapter les probabilités de MD et FA. Si les premier et dernier bits sont forcément à 1, le nombre de bits restant permettant de différencier les séquences devient $l-2$. On définit o le paramètre qui décrit si le code est totalement aléatoire ($o=0$), ou s'il contient la séquence de synchronisation ($o=1$). Dans ce cas, eq.3.10 et eq.3.11 deviennent :

$$P(FA \cap N) = (1-P(E))^{2o} \sum_{k=1}^{l-2o} \left(\left(\frac{1}{2}\right)^{l-2o} \binom{l-2o}{k} P(E)^k (1-P(E))^{l-2o-k} \right) \tag{3.12}$$

$$P(N) = 1 - \left(\frac{1}{2}\right)^{l-2o} \tag{3.13}$$

Comme espéré, on peut voit en fig. 3.7 qu'ajouter les 1 (croix oranges) permet d'obtenir de meilleures performances que pour le cas où les codes n'ont pas de contrainte (courbe verte), pour le même nombre de bits utiles. Notamment, les points sont sous la courbe théorique, ce qui montre qu'on obtient de meilleurs résultats en termes de MD. Néanmoins, ces performances sont moins bonnes si l'on compare à taille de signature égale (courbe bleue). Cependant, on obtient cela au prix d'une taille de signature accrue (et donc une consommation également accrue). La partie la plus intéressante de cette figure est la courbe pour une séquence entourée par les 1 avec la méthode Synchro. On peut voir qu'à partir d'une certaine valeur de bruit on continue d'avoir une probabilité de MD qui augmente, mais celle de FA commence à diminuer jusqu'à atteindre une valeur limite, autour de 10^{-3}. Cela est dû au fait que les premier et dernier bits des codes recherchés et émis sont toujours à 1. C'est pourquoi, s'il y a une erreur sur un de ces deux bits, il n'est plus possible d'avoir une FA et, pour de grandes valeurs de σ, la probabilité que ces bits

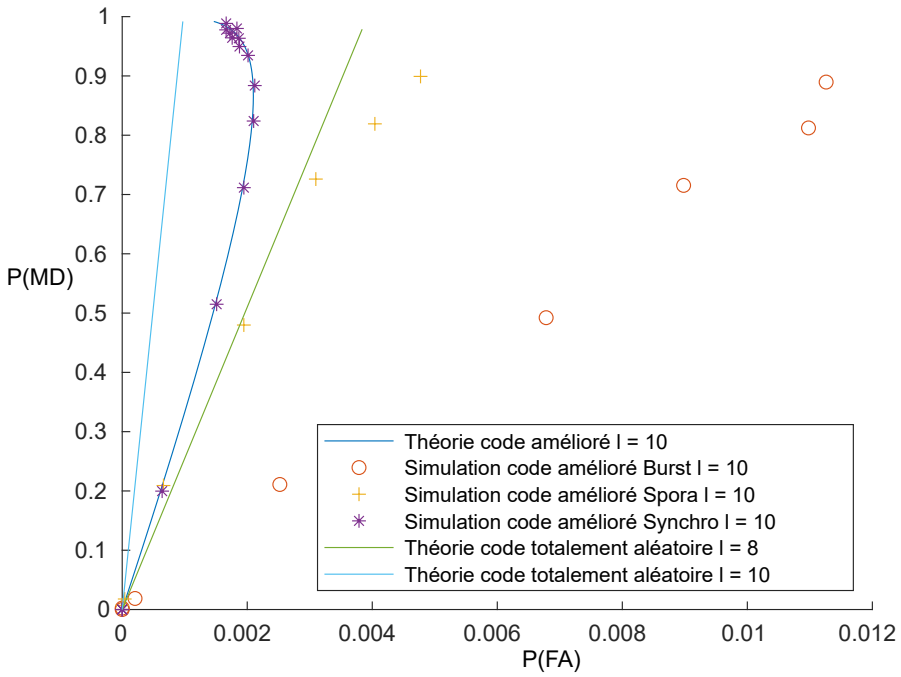


FIGURE 3.7 – Impact du bruit sur la probabilité de FA et MD lors de l'envoi d'un code de synchronisation

placés aux extrémités soient inversés augmente plus vite que la probabilité que tous les bits s'inversent pour devenir la séquence recherchée.

3.4.3 Analyse du seuil

Pour pouvoir protéger les 1 de synchronisation, on peut modifier la tension de seuil. Avec un seuil qui n'est plus placé à égale distance entre le potentiel de repos et le potentiel atteint après l'intégration d'une impulsion, nous obtenons deux nouvelles équations pour $P(E)$ car la probabilité d'erreur sur 0 (eq. 3.6) et celle sur 1 (eq. 3.7) ne seront plus égales.

Nous devons donc maintenant évaluer la probabilité d'obtenir une erreur selon chaque combinaison possible de signatures, ce qui va modifier également les probabilités de MD (eq. 3.8) et celle de FA $\cap N$ (eq. 3.12) :

$$\begin{aligned} P(FA \cap N) = (1 - P(E|1))^{2o} \sum_{k=1}^{l-2o} \Bigg(\left(\frac{1}{2}\right)^{l-2o} \binom{l-2o}{k} \\ \times \sum_{m=0}^{k} \left(\left(\frac{1}{2}\right)^{k} \binom{k}{m} P(E|0)^m P(E|1)^{k-m} \right) \\ \times \sum_{n=0}^{l-2o-k} \Bigg(\left(\frac{1}{2}\right)^{l-2o-k} \binom{l-2o-k}{n} \\ \times (1 - P(E|0))^n (1 - P(E|1))^{l-2o-k-n} \Bigg) \Bigg) \end{aligned} \tag{3.14}$$

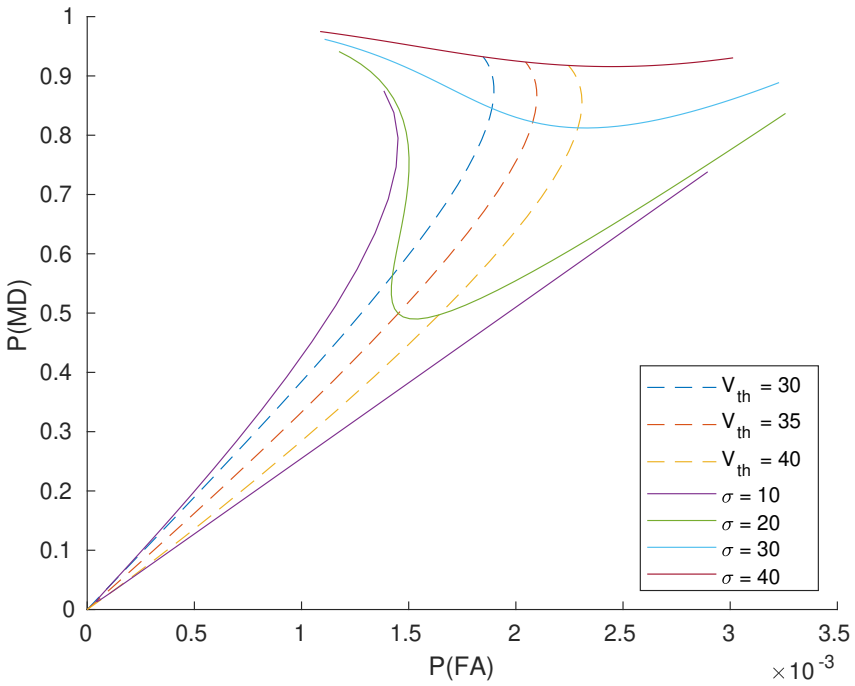


FIGURE 3.8 – Performances théoriques pour un code amélioré et avec la méthode Synchro en fonction de σ et v_{th} pour l = 10

$$P(MD) = 1 - \frac{1}{2^{l-2o}} \sum_{k=1}^{l-2o} \left(\binom{l-2o}{k} (1 - P(E|0))^k (1 - P(E|1))^{l-k} \right) \tag{3.15}$$

Sur la fig. 3.8 nous avons tracé les probabilités théoriques de FA et MD dans deux cas. Dans le premier, on trace les courbes pour une valeur spécifique de σ, pendant que v_{th} varie (lignes continues). Cela nous permet de définir, pour un groupe précis de paramètres, le seuil qui amène les meilleures performances. On peut observer que chacune de ces courbes a une valeur optimale, que nous tentons de trouver. Pour cela, nous traçons également les courbes avec σ qui varie pendant que v_{th} est constant (lignes en pointillés). Si notre signature inclut en moyenne autant de 1 que de 0, on s'attend à ce que le seuil soit optimal s'il est placé autour de $v_{th} = -35mV$, $30mV$ au dessus de la tension de repos, avec $v_{spike} = 60mV$ la tension ajoutée au potentiel membranaire lorsqu'une impulsion est reçue. En résumé, à équidistance entre la tension lorsqu'on reçoit un 0 et celle atteinte lorsqu'on reçoit une impulsion. Toutefois, si l'on force les bits des extrémités à 1, on aura en moyenne plus de 1 que de 0 dans les séquences, c'est pourquoi le seuil optimal est déplacé à $v_{th} = -30mV$ pour $l = 10$. Ce phénomène disparaît au fur et à mesure que la taille de la signature augmente, car la proportion moyenne de 1 se rapproche de $\frac{l}{2}$.

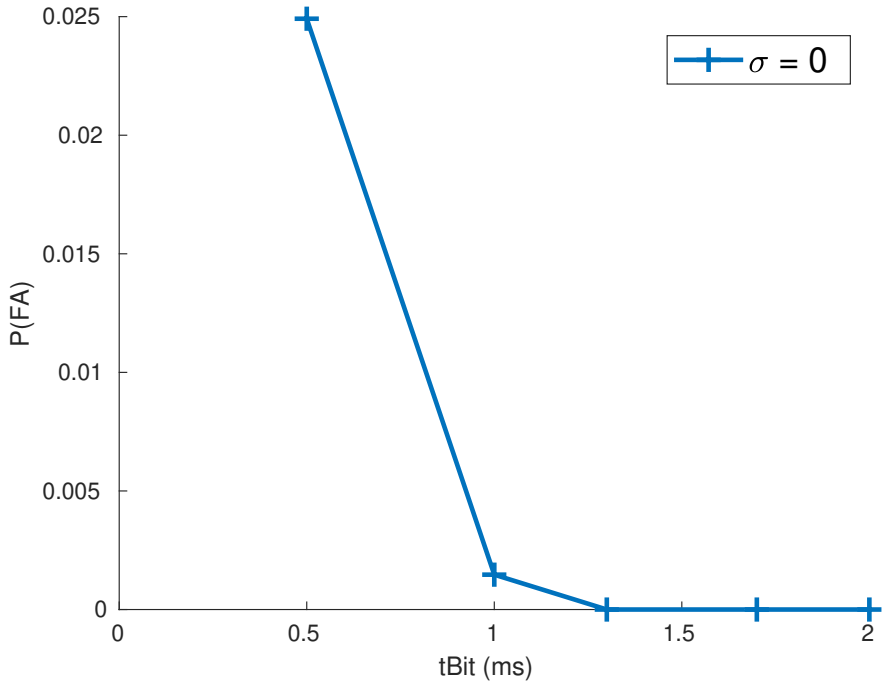


FIGURE 3.9 – Evolution des performances pour un code amélioré et la méthode Synchro en termes de FA pour différentes durées d'envoi de bits pour l = 10 et $\sigma = 0$

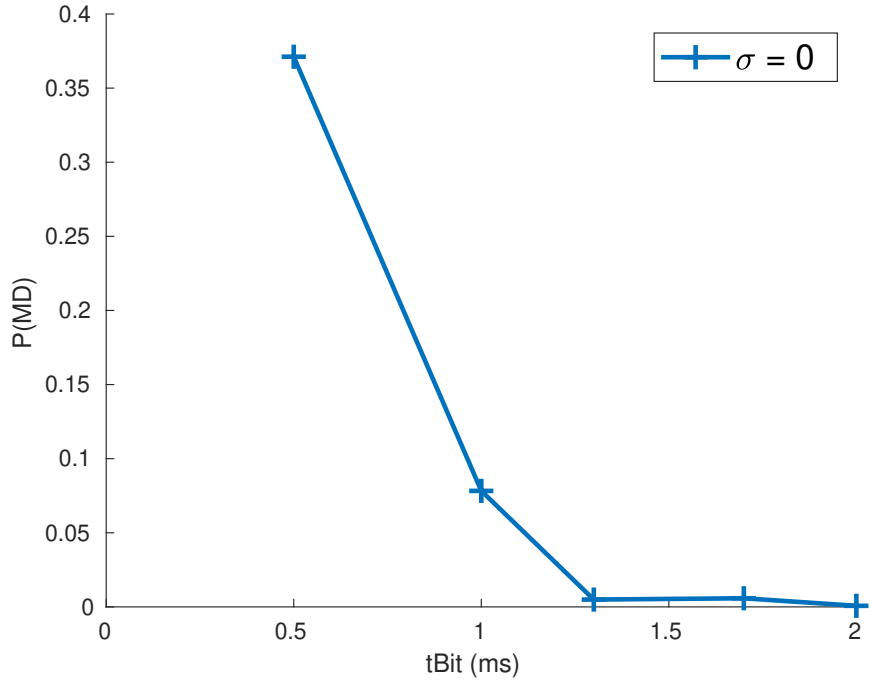


FIGURE 3.10 – Evolution des performances pour un code amélioré et la méthode Synchro en termes de MD pour différentes durées d'envoi de bits pour l = 10 et $\sigma = 0$

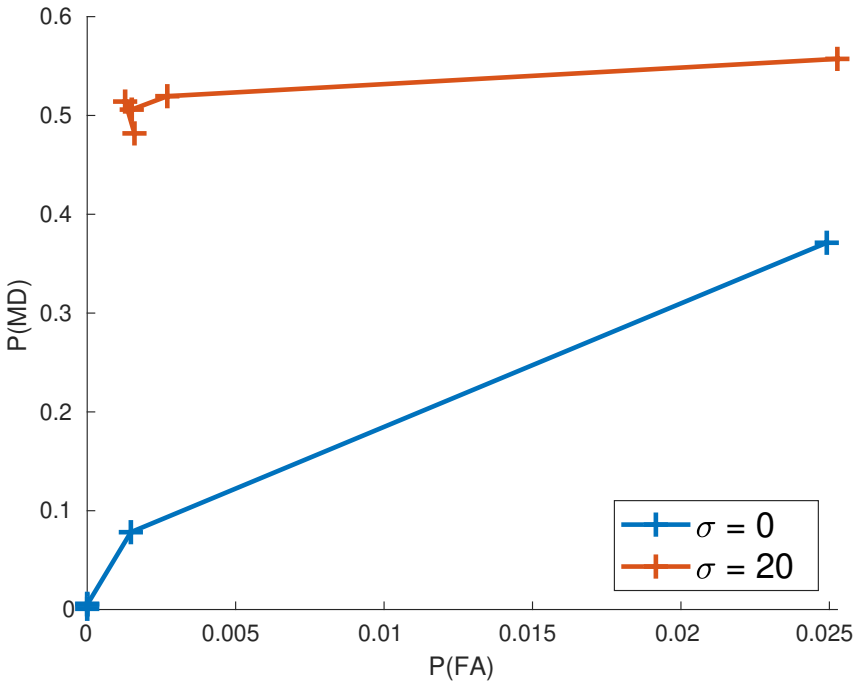


FIGURE 3.11 – Evolution des probabilités de FA et de MD pour un code amélioré et la méthode Synchro pour différentes durées d'envoi de bits pour l = 10 et pour σ = 0 et $\sigma = 20$

3.4.4 Impact de la contrainte temporelle

Enfin, nous nous concentrons maintenant sur une contrainte matérielle des neurones. Le comportement de ceux-ci se base sur leur constante de temps. Notamment, la diminution de la tension due à la fuite n'est pas instantanée, mais prend du temps. Si le débit des séquences est trop rapide, la tension membranaire n'a pas le temps de revenir à sa valeur de repos entre deux réceptions, ce qui va favoriser l'apparition de FA pour la séquence qui suit. Dans ce scénario, on évalue les performances en faisant varier le durée d'un bit. Pour cette étude, on utilise un troisième paramètre, $tBit$ qui représente cette durée. Nous avons effectué ces simulations avec la méthode Synchro et des signatures entourées par des 1.

On peut voir en fig. 3.9, 3.10 et 3.11 que laisser un temps plus important à notre neurone entre deux bits successifs contribue à améliorer la précision en termes de FA et MD. En effet, le neurone n'a pas assez de temps pour voir sa tension redescendre à sa valeur de repos entre deux réceptions d'impulsion. On peut également voir que pour notre modèle, il y a une durée à partir de laquelle il n'y a plus d'impact sur la précision, pour notre jeu de paramètres. Le délai entre deux symboles OOK est donc contraint. Le débit des codes a donc une borne supérieure de $800bit/s$. Cependant, on peut adapter C_m, la hauteur des impulsions, ou les caractéristiques internes du neurones pour atteindre plus rapidement v_{rest} et ainsi palier à cette potentielle limitation du débit.

TABLE 3.1 – Évolution de la puissance en fonction du nombre de synapse excitatrice

Nombre de 1	0	1	2	3	4	5	6	7	8
Puissance (en pW)	123	163	193	224	263	306	351	390	420

3.5 Réalisation matérielle du réseau

Pour compléter ces résultats de simulation, nous avons pu réaliser quelques tests sur la première itération du circuit contenant le réseau de neurones à impulsions artificiel.

Au sein du projet UWAKE, la partie située à l'IEMN est chargée de réaliser matériellement le réseau de neurone à impulsions et a réalisé des tests sur leur logiciel de simulation de composants électroniques afin de vérifier que le fonctionnement que l'on souhaitait marchait aussi sur leur implémentation. Après une série de tests concluants, nous avons confirmé le choix de la topologie et envoyé le réseau en fonderie.

Notre réseau fonctionne dans un cas non bruité en simulation côté neuroscience et côté électronique, et promet des résultats positifs tout en consommant bien moins que les WuRs actuelles. Il nous faut donc déterminer si ses performances sont bien comparables à ces WuRs.

Nous avons tout d'abord choisi une séquence test 11010110. Nous avons fait fonctionner le circuit et vérifié si la séquence ciblée entraînait bien la création d'une impulsion en sortie. Le réseau était paramétré avec $v_{rest} = 0V$ et un incrément par impulsion $v_{spike} = 60mV$. La fig. 3.12 présente l'affichage de l'analyseur. La partie basse de l'image représente les signaux d'entrée du réseau. La séquence se lit de haut en bas. Par exemple, la première reçue par le réseau a été la séquence cible 11010110. Dans cet exemple, le neurone 0 a donc reçu un 1, et le neurone 7 a reçu un 0. La courbe jaune représente ce que reçoit une sonde placée juste après le neurone de sortie. On observe des créneaux très rapides, qui nous servent d'impulsions dans le circuit réel, lorsque le circuit reconnaît la séquence. Celle ci a été envoyée deux fois dans l'expérimentation représentée, et on a bien deux impulsions à la sortie du neurone. Nous donc pu vérifier que lorsque le réseau était configuré avec les bons poids synaptiques, la réception de cette séquence permet au circuit d'atteindre son seuil, comme on peut l'observer en fig. 3.12.

Une fois le fonctionnement validé, nous avons regardé ce qu'il advenait du réseau lorsqu'on envoyait des séquences qui différenciaient très peu de celle ciblée, par exemple un ou deux bits inversés par rapport à celle ci. On peut observer en fig. 3.12 que parmi les sept mauvaises séquences envoyées, aucune n'a permis au réseau de décharger. Nous avons donc pu conclure au bon fonctionnement de notre réseau, la reconnaissance d'adresse est bien fonctionnelle.

Nous avons mesuré la puissance consommée par le système. Ces résultats ont été reportés dans la table 3.1 Celle-ci dépend du nombre de synapses excitatrices, c'est à dire du nombre de 1 dans la séquence recherchée. Le circuit a une puissance minimale de Stand-By, qui correspond à la puissance pour aucun 1 dans la séquence, et va voir sa puissance augmenter en fonction du nombre de 1.

Cette puissance reste dans l'ordre de grandeur de la centaine de pico Watts, ce qui est très faible comparé aux WuRs actuelles.

Pour les performances, nous n'avons pas pu mener les mêmes tests qu'en simulation. Nous avons en effet envoyé un signal avec un bruit additif (le même sur tous les bits à

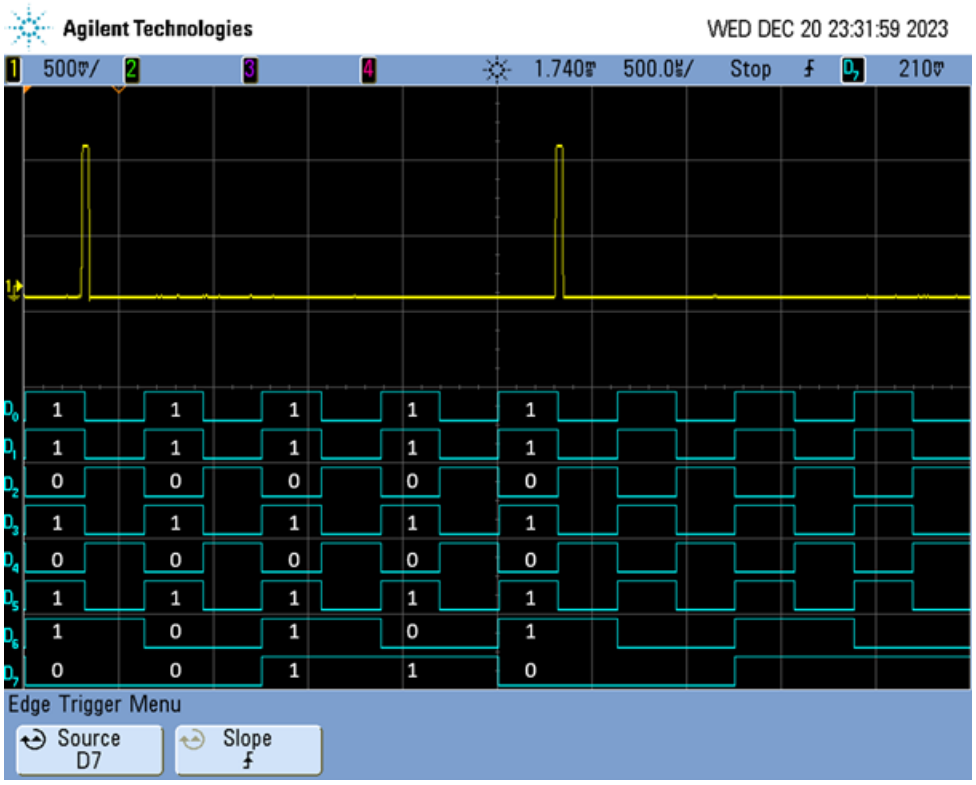


FIGURE 3.12 – Capture d'écran de l'oscilloscope pour valider la reconnaissance de la séquence

TABLE 3.2 – Évolution de la probabilité de MD en fonction de la tension de seuil

Seuil (en mV)	295	297	299	300	301	302
P(MD) (en %)	0	0.1	5	40	80	100

cause de contraintes matérielles), et compté le nombre d'impulsions que l'on perdait sur l'oscilloscope. Nous avons pu relever que pour un bruit blanc d'un écart-type de $20mV$, on ne relevait aucune MD, et qu'autour de $40mV$ on comptait une perte d'à peu près une sur quinze, soit approximativement 7%.

Nous avons ensuite regardé l'influence du seuil sur les FA et les MD sans bruit. Nous avons commencé par envoyer plusieurs fois la séquence ciblée, 11010110, et mesuré la probabilité de MD en faisant varier le seuil v_{th}. Les résultats sont à retrouver en table 3.2. On peut observer que le seuil maximal qui nous permet d'éviter les MDs se trouve être autour de $295mV$. Si les 5 impulsions représentant notre code augmentaient la tension d'exactement $60mV$ chacune, on n'aurait aucune MD pour un seuil inférieur à $300mV$, et 100% pour les valeurs de seuil supérieures à $300mV$. Il y a donc un très léger aléatoire sur l'incrément de tension dû à une impulsion, c'est pourquoi nous avons placé le seuil légèrement en dessous, afin de limiter les MDs.

Nous avons ensuite changé un bit "1" à "0" dans la séquence émise, afin de recevoir une séquence 'proche' de celle ciblée, et baissé la valeur de seuil pour mesurer l'apparition de fausses alarmes, en table 3.3. Nous avons pu observer qu'en baissant le seuil, on augmentait

TABLE 3.3 – Évolution de la probabilité de FA en fonction de la tension de seuil

Seuil (en mV)	295	287	286	285
P(FA) (en %)	0	17	95	100

la probabilité de FA si trop d'impulsions incrémentaient la tension d'une valeur supérieure à $60mV$. Nous avons donc choisi la valeur qui nous permet d'optimiser à la fois les MDs et les FAs, $295mV$.

Les valeurs mesurées restent toutefois peu précises car le mode de lecture des résultats ne permettaient pas une grande précision.

Enfin, nous avons observé la plage de fréquence avec laquelle il était possible de travailler. Le matériel contraint la fréquence maximale à $10kHz$, soit un temps entre deux impulsions $tBits$ de $100\mu s$. Ces valeurs sont valables pour les composants testés, et la disparités entre les différents circuits peut altérer ces valeurs.

Les valeurs relevées ici sont très différentes de celles obtenues en simulation. En effet, la manière dont ces neurones sont conçus fait que les paramètres utilisés sont très différents et qu'il n'a pas été possible de les mesurer pour reproduire exactement leur comportement.

3.6 Conclusion

Dans ce chapitre, nous avons proposé une architecture basée sur des neurones à impulsions et permettant la reconnaissance d'un motif de séquence, ainsi que la définition des séquences associées. Nous avons montré qu'ajouter les 1 aux extrémités de ces séquences nous permettait de gagner en précision jusqu'à coller aux performances de la méthode Synchro, sans circuit de synchronisation. Cela implique d'adapter la tension de seuil en fonction des séquences d'activation afin d'améliorer à la fois la probabilité de fausse alarme et de détections manquées. Nous avons également observé que l'on était bornés en termes de débit entrant par la dynamique de notre neurone et que, pour avoir une détection optimale, nous avions une limitation sur ce débit. Cependant, cette limitation n'en est pas vraiment une car nous souhaitons seulement réveiller un noeud, et pas transmettre des données.

Le circuit revenu de chez le fondeur a montré qu'il était fonctionnel et que ce module pourrait être utilisé dans le cadre du projet, couplé au module de récupération d'énergie RF, afin de créer un WuR a très faible puissance basé sur un circuit neuromorphique.

De plus, cette solution a une consommation encore plus faible que les WuRs utilisant des micro-contrôleurs dédiés à la reconnaissance de motifs, ce qui est prometteur.

De nos jours, les architectures de neurones consomment des puissances de l'ordre de $100pW$ par neurone [11]. De ce fait, pour reconnaître une séquence spécifique unique, avec notre réseau constitué de 11 neurones, la puissance totale du réseau serait de l'ordre du nW. C'est drastiquement en dessous des pics de puissance de l'ordre de la centaine de μW des WuRs actuelles. Si l'on passait à des séquences de longueur 100, ce qui permettrait de générer 2^{100} différentes séquences la puissance serait réduite d'un facteur 20000 par rapport aux WuRs actuelles.

Cependant, la conception tout analogique du SNN rend extrêmement difficile la mise en oeuvre de délais synaptiques, utilisé pour retarder le signal entre la couche d'entrée et la couche cachée. Malgré le fait qu'une solution annexe ait été trouvée, cela a limité

fortement les solutions à notre disposition, car il devenait impossible de mettre en place des architectures de réseau employant une forme de rétroaction, et limite les possibilités pour mettre en place de l'apprentissage dans ce réseau. Cela est dû au fait que l'on ne peut plus synchroniser le temps symbole et le temps que met une impulsion à parcourir une synapse, ce qui est problématique lorsque l'on travaille avec des signaux temporels.

Nous avons donc cherché s'il n'était pas possible d'utiliser des modèles de neurones qui pourraient faire de la reconnaissance de motifs, tout en s'affranchissant du phénomène de délai synaptique.

Chapitre 4

Modèle SLIF pour reconnaissance de patterns basés sur un IST

4.1 Introduction

Comme nous l'avons vu, un cerveau biologique traite l'information à l'aide de cellules spécialisées appelées neurones. Ces neurones communiquent entre eux en s'envoyant des pulsions électriques appelées impulsions, à travers des canaux nommés synapses. Ces neurones vont agréger les impulsions reçues dans leur tension de membrane, et ne vont renvoyer une impulsion qu'une fois que leur tension de seuil a été atteinte. Le fait de dépasser ce seuil va déclencher immédiatement l'envoi d'une impulsion au neurone suivant. Ce procédé correspond à celui qu'utilise le cerveau pour traiter chaque information.

Le modèle considéré dans le chapitre 3 est le modèle le plus utilisé en neurosciences computationnelles et dans le domaine des SNNs, le modèle Leaky Integrate and Fire (LIF) [46] dont l'équation est :

$$C_m \frac{dv}{dt} = I(t) - g_L(v(t) - v_{rest}) \tag{4.1}$$

avec C_m la capacité de membrane, $v(t)$ le potentiel de membrane au temps t, g_L la conductance de membrane et v_{rest} le potentiel membranaire de repos avec $v_{rest} = -65mV$.

$I(t)$ modélise ici le courant synaptique entrant dans un neurone, c'est à dire le courant provenant d'un neurone pré synaptique. Il correspond à une séquence d'impulsions, et peut être écrit comme :

$$I(t) = I_0 \sum_k \delta(t - t_k) \tag{4.2}$$

avec t_i correspondant aux différents temps d'envoi des impulsions, et δ la fonction Dirac.

A chaque fois qu'une impulsion arrive au neurone, la tension augmente en conséquence. Lorsque cette tension atteint son seuil, noté v_{th}, il déclenche une impulsion en direction du neurone suivant, et sa propre tension redescend à v_{rest}. Il est à noter que dans le cas d'un neurone LIF, il y a une augmentation immédiate de la tension lorsqu'on reçoit une impulsion, suivie par une décroissance exponentielle due au phénomène de fuite du neurone.

Le phénomène de délai synaptique correspond au fait d'ajouter une caractéristique à la synapse, résultant en un délai entre le moment où le neurone pré synaptique atteint

son seuil et envoie une impulsion à ses synapses de sortie, et le moment où le neurone post synaptique reçoit l'impulsion considérée.

Ce mécanisme neuronal était très important pour réaliser le circuit présenté dans la partie précédente, mais aucune solution n'a été trouvée pour l'implémenter électroniquement. Il nous a donc fallu trouver une alternative afin de pouvoir traiter des signaux temporels avec des neurones bio-inspirés.

4.2 Modèle SLIF

Le modèle LIF est un outil très utilisé, mais également très simplifié. Cependant, il est possible de l'améliorer en ajoutant la modélisation de phénomènes peu considérés comme le comportement des canaux ioniques qui parcourent la membrane [48], ou la théorie du câble linéaire [49]. Dans notre cas, nous ajoutons à ce modèle le phénomène de saturation synaptique [50]. Ce nouveau modèle est appelé le LIF saturant (ou Saturating LIF : SLIF). Cette modification prend en compte l'altération temporaire des paramètres de la synapse, juste après avoir reçu une impulsion [26, 50]. Il en résulte que le courant n'est plus seulement déterminé par le courant d'entrée, mais également par les évènements précédents.

La nouvelle expression de la tension de membrane est donnée par :

$$C_m \frac{dv}{dt} = g_s(t)(E_s - v(t)) - g_L(v(t) - v_{rest}) \tag{4.3}$$

Où g_s est la conductance synaptique, E_s est le potentiel de réversion synaptique placé à $0mV$, $v(t)$ est le potentiel de membrane, g_L est la conductance de membrane, et $v_{rest} = -65mV$ est comme précédemment le potentiel de membrane.

L'équation (4.3) est régie par deux phénomènes : le premier décrit la réponse à des impulsions, régie par $g_s(t)$ augmentant la tension de membrane, et l'autre décrit la tension de fuite du neurone, régie par g_L, et réduisant cette tension. La compétition entre ces deux termes résulte en la prédominance du premier lors de la réception d'une impulsion et directement après, jusqu'à ce que le deuxième redevienne prédominant après un court moment.

Par défaut au repos, $g_s(t)$ est initialisé à $0S$ ce qui annule le premier terme de l'équation. Cependant, lorsqu'une impulsion arrive, $g_s(t)$ augmente instantanément jusqu'à atteindre sa borne maximale, appelée g_s^{max}. $g_s(t)$ est contraint à saturer à g_s^{max} et ne peut pas dépasser cette valeur même en cas de réception de plusieurs impulsions rapprochées. C'est pour cela que l'on parle de Synapse Saturante. Cette augmentation permet au terme $g_s(t)(E_s - v(t))$ de devenir strictement positif, car $Es = 0V$ et $v(t)$ est négatif, et supérieur au terme de fuite ce qui entraîne une hausse de la tension de membrane. Dans le cas du LIF, on pourrait considérer $g_s(t)$, qui serait directement proportionnel à $I(t)$ dans (4.1) comme on peut le voir dans la partie haute de la fig 4.1.

Ensuite, $g_s(t)$ subit une décroissance exponentielle régie par l'équation :

$$\frac{dg_s(t)}{dt} = \frac{-g_s(t)}{\tau_s} \tag{4.4}$$

où τ_S représente la constante de temps synaptique qui régit le comportement de $g_s(t)$.

La partie basse de la fig. 4.1 représente l'évolution de g_s pour une synapse saturante. Si deux impulsions sont trop rapprochées temporellement, cela va réduire l'impact de la seconde impulsion sur $g_s(t)$ qui va connaître une augmentation réduite. A l'inverse, si les

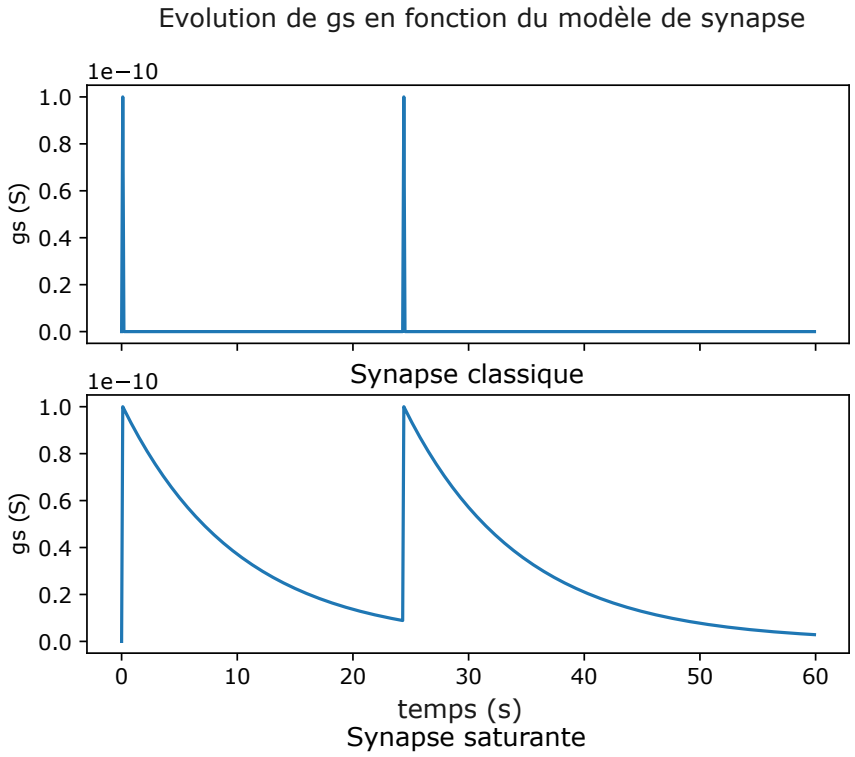


FIGURE 4.1 – Évolution de la conductance synaptique des 2 modèles de neurones.

impulsions sont assez éloignées, g_s aura le temps de redescendre à 0 avant de remonter à g_s^{max}. On a vu dans l'eq. 4.3 que la tension dépend de la valeur de g_s. On peut comprendre que l'aire sous la courbe de g_s atteint un maximum si g_s a le temps de redescendre à 0 entre les deux impulsions.

Nous proposons d'exploiter la structure de cette synapse dans le but d'obtenir un système composé d'un neurone et de ses synapses qui ne génère des impulsions que lorsqu'on reçoit un train d'impulsions à un rythme ciblé, et ignore n'importe quel autre signal entrant. Pour cela, nous avons réalisé le co-design du système neurone-synapse et du train d'impulsions qui lui sert de séquence d'activation, de manière à ce que le neurone atteigne une amplitude maximale pour cette séquence spécifique.

4.3 Étude du modèle

Nous présentons ici une étude détaillée de la réponse d'un neurone couplé à des synapses saturantes. Tout d'abord, nous illustrons le phénomène considéré en envoyant deux impulsions successives à deux types de neurones : le LIF standard et le SLIF. Il peut être observé en comparant les deux courbes sur la fig. 4.2. L'écart temporel entre la réception des deux impulsions est appelé Inter-Spike Timing (IST) (aussi appelé ISI pour Inter Spike Interval dans certains articles de la littérature). Pour une simplification des notations, envoyer un IST à un neurone correspond au fait de lui envoyer deux impulsions séparées de ce temps inter impulsion. En comparant les réponses du LIF et du SLIF à un IST, nous présentons une analyse des effets et des avantages de l'utilisation du modèle de neurone et synapses SLIF.

Nous avons tracé en fig. 4.2 l'évolution de la tension membranaire d'un neurone LIF et d'une SLIF lors de la réception de deux impulsions envoyées aux mêmes instants. Nous

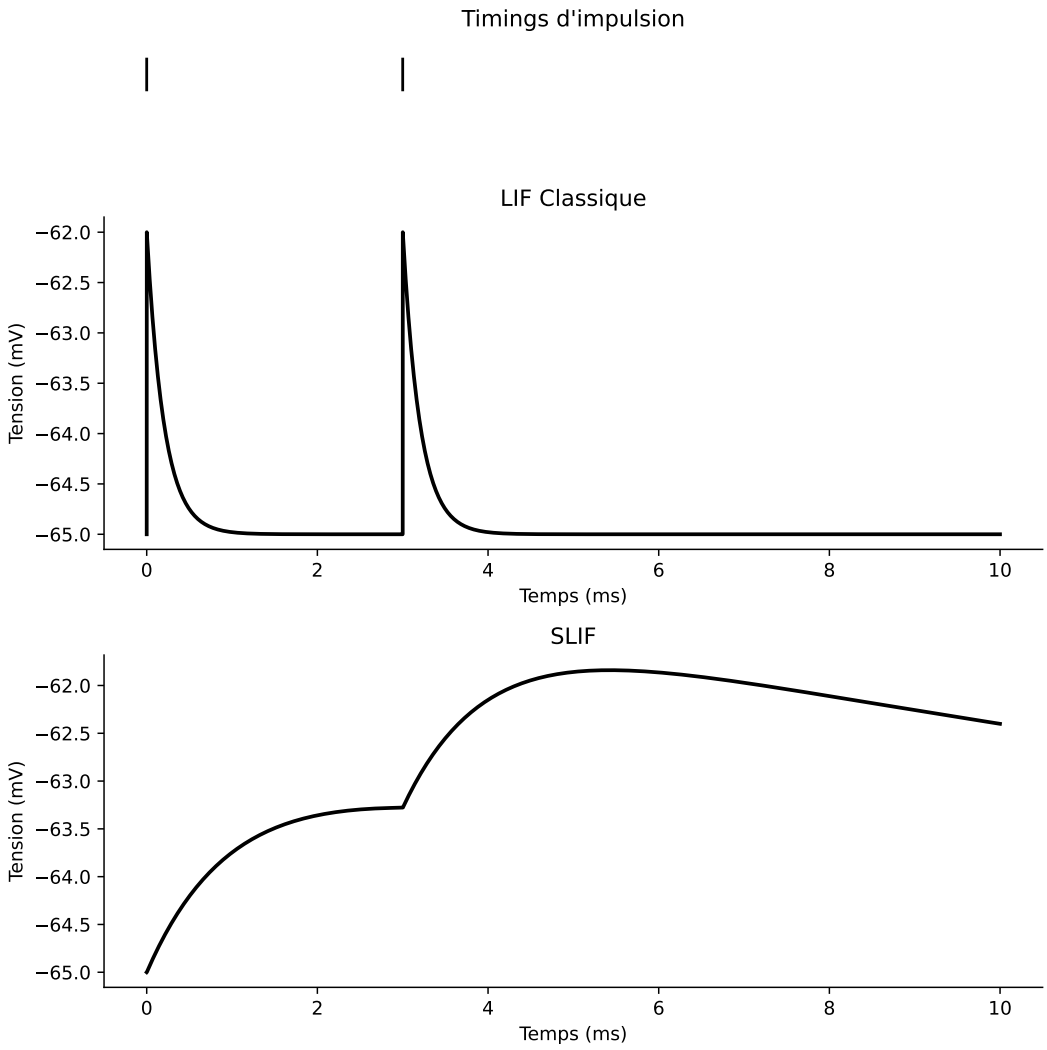


FIGURE 4.2 – Évolution de la tension membranaire des 2 modèles de neurones pour 1 IST.

avons utilisé l'eq. (4.1) (resp. eq. (4.3)) pour simuler le comportement de notre neurone classique (resp. saturant), en calculant la valeur de la tension membranaire lors de la réception d'une impulsion. Cela nous permet de déterminer l'évolution de cette tension au cours du temps. On peut retrouver comme vu dans le chap. 2 que lorsqu'un neurone LIF classique reçoit une impulsion, sa tension membranaire augmente instantanément. Cependant, elle va ensuite descendre graduellement au cours du temps à cause de la fuite dans le neurone. Maintenant, lorsqu'une deuxième impulsion est reçue, la tension va à nouveau augmenter, de la même amplitude que lors de la réception de la première impulsion. L'amplitude maximale est donc atteinte lorsque l'IST est minimal, c'est à dire lorsque le neurone a eu le moins de temps possible pour laisser le phénomène de fuite diminuer cette tension. Si au contraire la seconde impulsion est envoyée longtemps après le premier en comparaison des temps considérés, la tension membranaire aura le temps de revenir vers sa tension de repos, et donc l'amplitude maximale due à cette séquence sera amoindrie en comparaison au cas précédent. On peut donc en conclure que dans le cas du LIF, l'amplitude maximale atteinte par le neurone en fonction de l'IST reçu est décroissante. L'IST conduisant à une amplitude maximale d'un neurone LIF est donc nul.

La première partie de l'eq. (4.3) décrit comment la réception d'une impulsion permet

d'augmenter la tension du neurone. Par défaut, $g_s(t) = 0$, annulant ainsi le premier terme. Cependant, une impulsion entrante fera instantanément augmenter $g_s(t)$ jusqu'à sa valeur de saturation g_s^{max}.

Ce terme $g_s(t)(E_s - v(t))$ augmente et devient supérieur à 0, tendant à augmenter la tension. Ensuite, $g_s(t)$ diminue exponentiellement jusqu'à la réception de la suivante.

L'intérêt de ce modèle devient évident lors de la transmission successive de plusieurs impulsions. La croissance limitée de $g_s(t)$ ralentit la montée vers la tension membranaire cible, ce qui, couplé à la diminution du terme $E_s - v(t)$, contribue au comportement observé sur la courbe de la fig. 4.2.

Nous pouvons ainsi tracer l'évolution de la tension pour plusieurs séquences d'entrée en fig. 4.3, qui montre quatre courbes différentes, correspondant chacune à une configuration différente. Les temps auxquels sont envoyés les impulsions sont représentés en haut de la figure en 4.3a. Les quatre courbes sont indépendantes les unes des autres. La courbe grise correspond à la réponse du neurone lors de la réception d'une seule impulsion. Les trois autres de chaque graphe correspondent à la réponse du neurone à une séquence de deux impulsions (la grise et celle qui a la même couleur en haut de la figure), dont l'IST varie selon la couleur. De gauche à droite on peut retrouver ces trois ISTs : $0.5ms$ pour les courbes rouges, $3ms$ pour celles en noir, et $7ms$ pour les vertes respectivement. Ces valeurs ont été choisies pour pouvoir bien illustrer le phénomène, et peuvent être adaptées comme on pourra le voir en section 4.5.

Lorsque nous envoyons deux impulsions, l'amplitude maximale atteinte par le potentiel membranaire dépendra du délai entre ces deux impulsions, l'IST. Si ces deux impulsions sont envoyées trop rapprochées, alors $g_s(t)$ n'aura que légèrement diminué. Ainsi, la seconde impulsion, qui ramène $g_s(t)$ à son maximum, n'aura qu'un faible impact.

Habituellement, lorsque la tension d'un neurone atteint son seuil, elle retombe instantanément à son potentiel de repos et le neurone envoie une impulsion aux neurones suivants comme on a pu le voir au chapitre 3. Cependant, les seuils représentés par des lignes pointillées sur la fig. 4.3 sont fictifs et n'ont pas été implémentés dans les neurones montrés, afin que nous puissions observer toute la dynamique des neurones. La courbe noire sur la fig. 4.3 représente la réponse à l'IST Favori, menant à l'amplitude la plus élevée. Le seuil est placé arbitrairement 0.1 mV sous cette amplitude. Cependant, il existe d'autres ISTs que le favori qui permettent d'atteindre une amplitude supérieure au seuil. Ces ISTs se trouvent dans un intervalle autour de l'IST favori. Nous appelons cet intervalle d'ISTs la "largeur temporelle" (TimeWidth (TW) pour TimeWidth).

Il est intéressant de noter que contrairement au cas du LIF, l'incrément de tension membranaire apporté par la réception de la seconde impulsion de la séquence n'est pas constant, il varie en fonction de l'IST de cette séquence. Par exemple, sur la courbe rouge, les impulsions sont trop proches l'une de l'autre. La saturation synaptique va donc être très importante et limiter fortement l'impact de la deuxième impulsion. Il en résulte donc que cette deuxième impulsion ne va avoir qu'un très léger effet sur la tension membranaire.

A contrario, pour la courbe verte, les impulsions sont trop éloignées l'une de l'autre, et le phénomène de fuite va faire redescendre la tension jusqu'à quasiment atteindre la valeur de repos avant la réception de la seconde impulsion. Dans ce cas, la saturation synaptique a presque disparu et la seconde impulsion aura un impact équivalent à celui de la première. Toutefois, vu que l'on était revenu à une tension proche de celle de repos, l'amplitude maximale atteinte après la seconde impulsion sera très proche de celle que l'on avait déjà atteinte avec la première contribution.

La courbe noire, quant à elle, représente le compromis entre les deux comportements

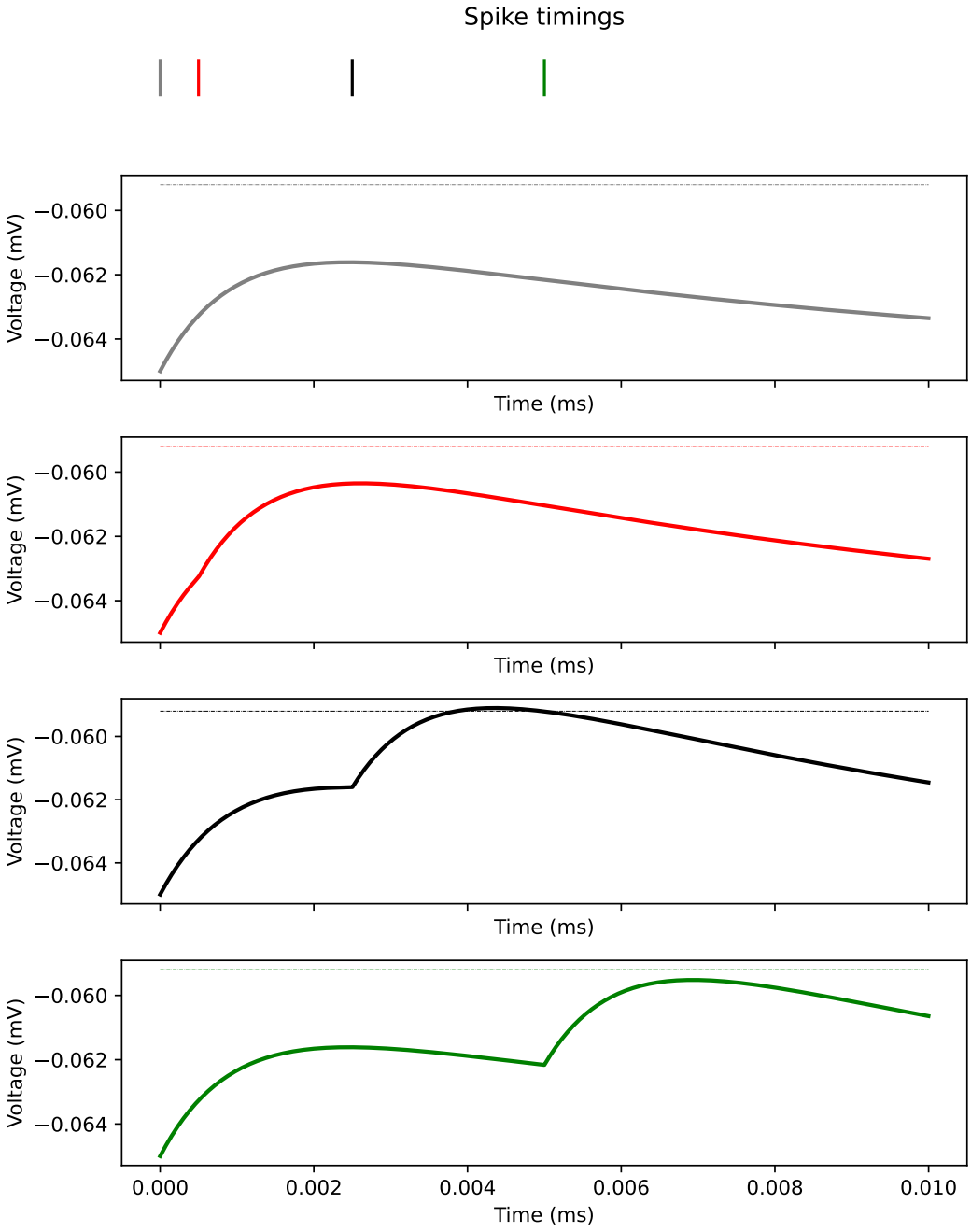


FIGURE 4.3 – Réponse d'un neurone SLIF à différents ISTs

démontrés par les courbes rouge et verte. L'IST stratégiquement choisi pour cette illustration permet d'atteindre une amplitude maximale pour le premier lobe, juste avant l'arrivée de la seconde impulsion. Dans le cas d'ISTs plus courts, la saturation rend l'impact de la seconde beaucoup moins significatif. Pour de plus grands ISTs, l'incrément de tension membranaire dû à la première impulsion a quasiment disparu lors de la réception de la deuxième. En conséquent, le second lobe peut se manifester avant que le phénomène de fuite ne gagne trop d'influence sur la dynamique, et empêche de visualiser ce lobe. La propriété essentielle à notre approche correspond au fait d'empêcher l'apparition d'un deuxième pic dans la tension membranaire représentée en rouge, c'est ce qui fait la plus grande différence avec le modèle LIF. Ce phénomène peut être observé en biologie [50].

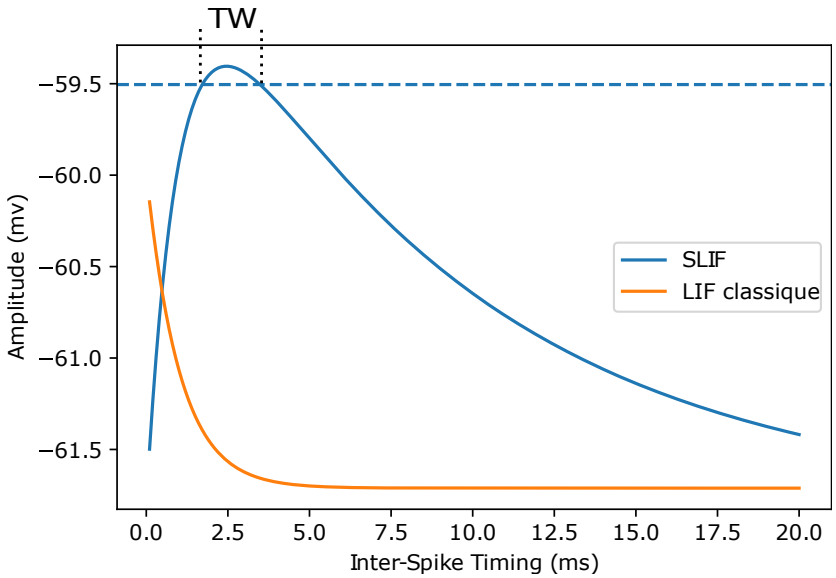


FIGURE 4.4 – Amplitude atteinte en fonction de l'IST pour un neurone LIF et un SLIF.

TABLE 4.1 – Différences entre les modèles LIF et SLIF

Caractéristiques	**Modèle LIF classique**	**Modèle LIF avec Synapses Saturantes**
Paramètres	C_m, g_L	C_m, g_L, τ_S
Conductance Synaptique	Conductance fixée	Conductance variable, saturante
Dynamique du neurone	Intégration et Fuite	Aussi influencée par les synapses
Potential Membranaire	Croissance instantanée	Croissance lente due à la saturation

Afin de voir l'évolution de la tension sur la fig. 4.4, nous avons tracé les amplitudes atteintes en fonction de l'IST de la séquence d'entrée. Si on utilise des synapses classiques, on peut observer une fonction monotone qui décroît de manière exponentielle. Par contre, pour le neurone avec des synapses saturantes, un comportement distinct apparaît.

La figure révèle que pour notre jeu de paramètres ($C_m = 2.83.10^{-12}$ F, $g_L = 3.77.10^{-10}$ S and $\tau_S = 3.10^{-3}$ s), l'amplitude maximale observée est atteinte pour un IST de $3ms$. La courbe de la réponse montre deux phases distinctes. Pour des valeurs d'ISTs inférieures à $3ms$, l'effet de saturation entraîne une amplitude plus basse que l'optimum. Pour des valeurs d'IST plus grandes que l'IST Favori, c'est le phénomène de fuite qui prend le relais et entraîne une réduction progressive de la tension membranaire en fonction de l'IST. Cela suggère que pour ce neurone SLIF spécifique, un IST proche de $3ms$ nous place dans le cas le plus favorable pour atteindre une réponse en amplitude maximale.

Pour illustrer la manière dont on peut bénéficier de la saturation synaptique, nous avons rajouté une ligne horizontale en pointillés qui représente une valeur de tension de seuil fictive. Dans ce scénario, le neurone devrait envoyer envoyer une impulsion si et seulement si la tension membranaire dépasse ce seuil. Cela revient à dire que le neurone ne va renvoyer une impulsion que si la séquence de deux impulsions qu'il reçoit est configurée avec un IST compris dans une plage de valeurs très précise. Dans cet exemple, le neurone va atteindre son seuil si les deux impulsions sont espacées d'un IST compris entre 2 et $3.5ms$. Ici, $TW = 1.5ms$.

Le tableau 4.1 résume les principales différences entre le modèle LIF classique et le modèle SLIF.

4.4 Influence des paramètres

La forme de la courbe de la fig. 4.4 (comprenant le temps de montée, le temps de descente, l'amplitude maximale, et les temps d'apparition de ces phénomènes), est influencée par les différents paramètres des équations utilisées pour modéliser le comportement de notre système :

— C_m (capacité de membrane) permet de dimensionner la dynamique globale du neurone. Une plus grande valeur de C_m va ralentir l'évolution de la tension membranaire, et logiquement, diminuer C_m va augmenter les variations de tension.

— g_L (conductance du neurone) régule le phénomène de fuite de la tension de membrane. Une plus grande valeur de g_s va entraîner un retour à la tension de repos (v_{rest}) plus rapide, alors qu'une valeur inférieure va ralentir cette dynamique.

— g_S (conductance synaptique) est impactée par le paramètre τ_S (constante de temps synaptique). g_s représente la conductance synaptique, variant au cours du temps. Elle joue un rôle crucial dans la dynamique de la tension membranaire. Une grande valeur de τ_S va permettre à g_s de revenir plus lentement à sa valeur de repos (ici 0), ce qui va augmenter la durée durant laquelle les impulsions auront un impact sur la tension de membrane. A l'inverse, diminuer τ_S va faire redescendre g_s beaucoup plus vite, ce qui va réinitialiser g_s plus rapidement, et permettre à des impulsions plus rapprochées de la première d'avoir un grand impact sur $v(t)$. On pourrait réaliser un parallèle entre les modèles LIF et SLIF en considérant le modèle LIF comme un SLIF pour lequel on aurait $\tau_{S\_LIF} = 0$, c'est à dire qu'il redescend instantanément à 0 après avoir atteint g_s^{max}. Il prendrait ainsi la forme d'une suite de plusieurs Dirac. Dans nos simulations, g_s est contraint à rester entre 0 et $100pS$, de la même manière que dans [50].

En modifiant ces paramètres, il est possible de choisir la réponse du neurone. Par exemple en fig. 4.5, un nouveau jeu de paramètres a été utilisé pour passer d'une plage d'ISTs en millisecondes (comme celle utilisée sur la fig. 4.4) à une plage en microsecondes. Cette modification a permis de rendre la courbe plus étroite en temps tout en gardant l'amplitude de la réponse constante. L'ajustement de ces paramètres permet d'affiner le comportement du neurone et d'adapter sa réponse aux différents types d'entrée et aux séquences temporelles.

De plus, la fig. 4.5 permet d'illustrer nos quatre principales métriques qui seront évaluées dans la suite du manuscrit, et qui nous permettent de décrire la réponse du neurone : l'amplitude, la marge, l'IST favori et le TW. L'**amplitude** représente la différence entre la tension maximale atteinte lors de la réception de la séquence, et la tension de repos qui correspond à la tension la plus petite que peut atteindre le neurone pour les ISTs considérés. L'amplitude indique l'ampleur de la variation de la tension membranaire. La **marge** représente la différence entre la tension maximale atteinte par le neurone après la réception de la dernière impulsion et la tension maximale du lobe précédent. Pour mesurer le **TW**, nous avons établi un seuil placé $0.1mV$ sous la tension maximale, et regardé la largeur de la plage d'ISTs qui permettent de dépasser ce seuil. Si l'**IST Favori** représente la séquence idéale grâce à laquelle le neurone répond de la manière la plus robuste, le TW représente la plage d'ISTs pour lesquels le neurone peut générer des impulsions.

Pour illustrer le fonctionnement de ce type de système, nous avons étudié deux neurones différents répondant donc à deux ISTs différents. Chaque neurone reçoit deux sé-

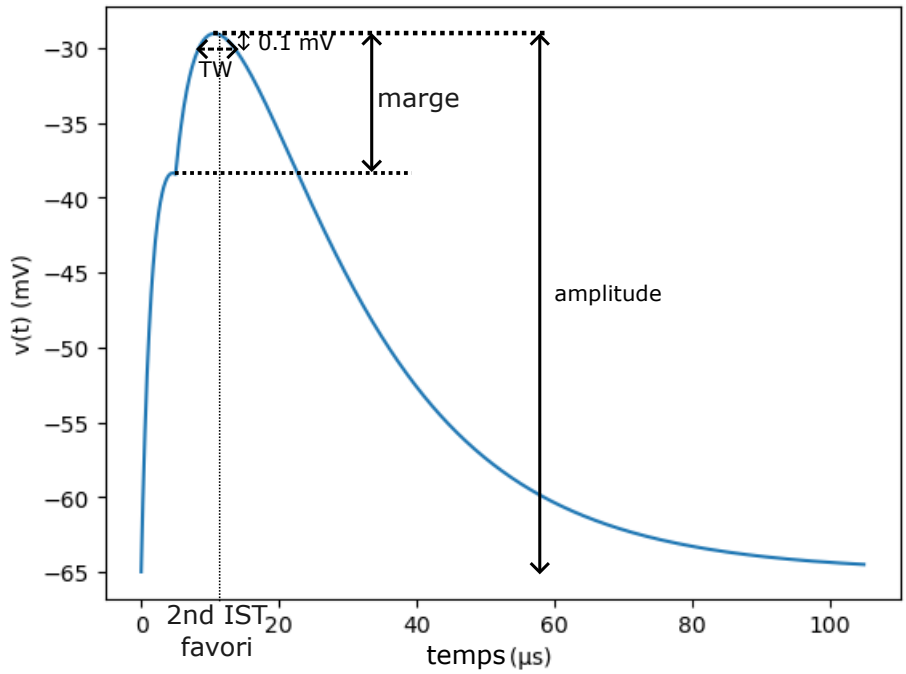


FIGURE 4.5 – Représentation des différentes métriques utilisées

quences : la séquence réalisée à l'aide de l'IST du premier neurone, puis celle correspondant au second IST. L'objectif principal était de valider que chaque neurone réponde spécifiquement à sa séquence associée et non aux autres.

Comme le montre la fig. 4.6, le neurone N_1 présente une forme de réponse différente entre l'une et l'autre des séquences. On reconnaît que le premier neurone atteint son seuil pour les 4 premiers pics au fait que la décroissante est instantanée, alors que pour les 4 suivants elle est plus lente car le seuil n'a pas été atteint. L'inverse est également vrai pour le neurone 2. Cette observation souligne la sélectivité de nos neurones, permettant la discrimination des ISTs et, par conséquent, des séquences. Cependant, il est important de noter que le TW de nos neurones n'est pas nul, ce qui signifie qu'un léger décalage dû au bruit aurait pu potentiellement empêcher la décharge d'une impulsion dans le réseau. De même, il aurait pu engendrer une impulsion pour une paire d'impulsions dont le décalage était proche de l'IST ciblé. Par conséquent, notre prochaine étape consiste en une évaluation complète et une quantification de cette sélectivité afin de déterminer les indicateurs de performance globaux du réseau. De même, il aurait pu engendrer une impulsion pour une paire d'impulsions dont le décalage était proche de l'IST ciblé.

Afin d'optimiser notre système, nous avons besoin de réaliser deux objectifs. Tout d'abord, nous cherchons à maximiser l'amplitude afin d'améliorer la robustesse de notre neurone, et sa capacité à gérer une gamme plus large de niveaux d'entrée, de manière à le rendre plus résilient en cas de perturbations. Dans un second temps, nous tentons de minimiser la largeur du TW, afin d'améliorer la précision du neurone dans la sélection de débits spécifiques et dans se capacité à ne répondre qu'aux motifs de séquences désirés.

Le choix de l'IST Favori dépend de l'échelle temporelle de la séquence choisie, en fonction de la tache ou du calcul requis. En déterminant le bon jeu de paramètres, il est possible de trouver un équilibre entre la maximisation de l'amplitude et la minimisation du TW pour un IST favori visé.

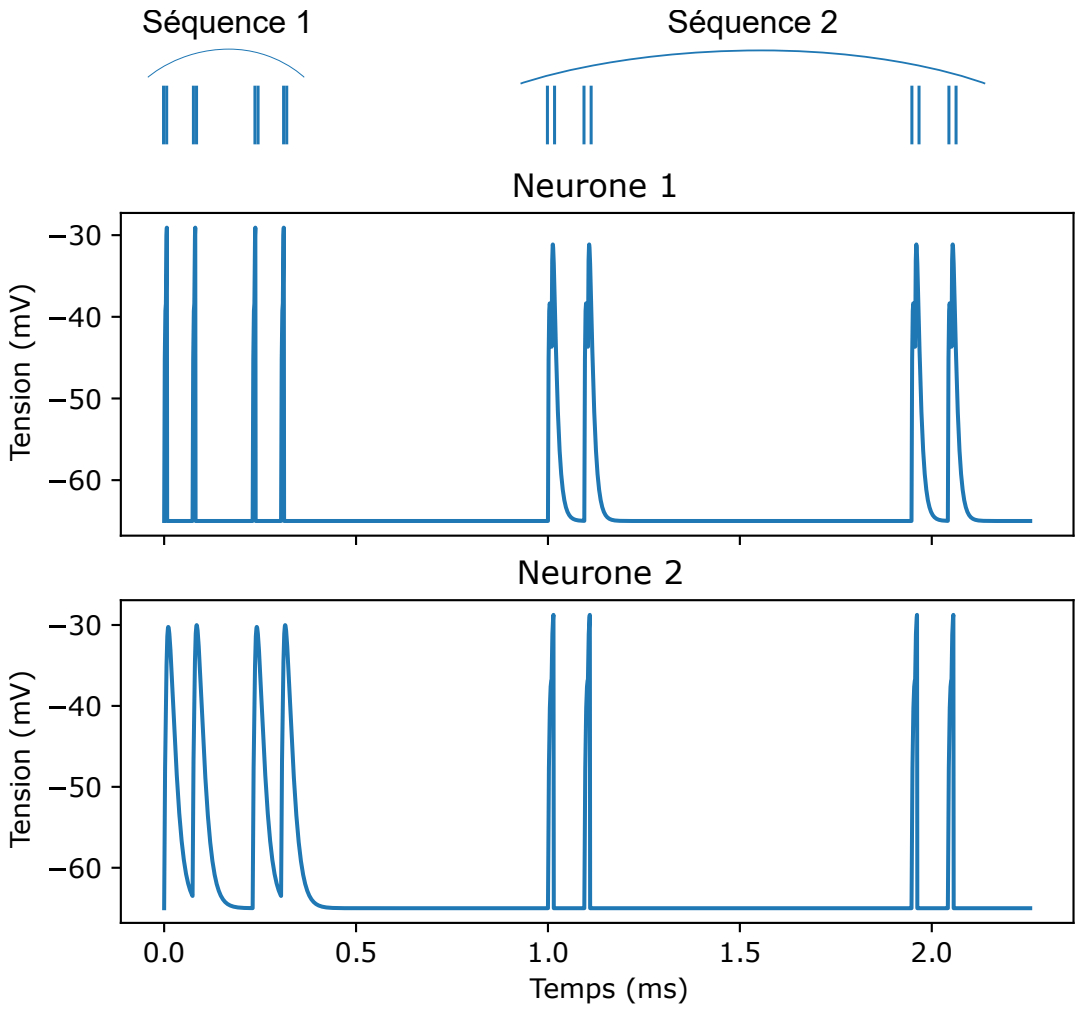


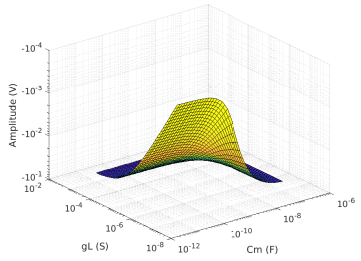
FIGURE 4.6 – Évolution de la tension de deux neurones différents lors de la réception de deux séquences basées sur des ISTs différents

4.5 Résultats

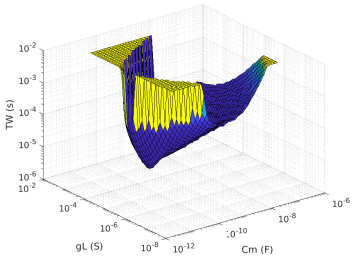
Nous présentons ici une étude dans laquelle nous avons établi un ensemble de paramètres de référence. Dans ce jeu de paramètres, chacun d'entre eux a été fixé à une valeur précise de référence : $C_m = 10^{-9}$ F.cm^{-2}, $g_L = 10^{-5}$ S.cm^{-2}, et $\tau_s = 10^{-4}$ s. Ce jeu correspond aux ordres de grandeurs biologiques de ces paramètres [50], ce qui nous donne des ISTs de l'ordre de la milliseconde. Afin de tester une large plage de paramètres, nous allons systématiquement faire varier ces paramètres entre 0.001 et 1000 fois leur valeur de référence définie ci-dessus. En réalisant ces variations de paramètres, nous cherchons à fournir une compréhension plus approfondie de la manière dont C_m, g_L, et τ_s impactent à la fois l'amplitude et le TW dans la réponse du neurone.

Sur la fig. 4.7, chaque colonne représente l'observation d'une métrique différente, respectivement l'amplitude, le TW et l'IST favori, alors que chaque ligne est obtenue en faisant évoluer conjointement un couple de paramètres, le troisième restant fixé à la valeur mentionnée précédemment.

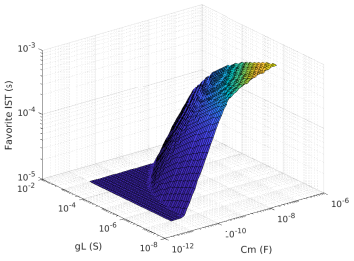
— C_m : En raison de son influence directe sur la dynamique du neurone, nous avons commencé par le paramètre C_m. Notre but est de déterminer l'impact de ce paramètre sur l'IST Favori. Les figures 4.7c et 4.7f montrent que l'évolution de l'IST



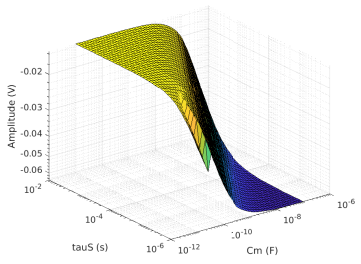
(a) Amplitude maximale avec l'IST favori pour $\tau_S = 0.1ms$



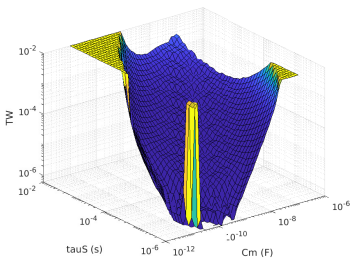
(b) Timewidth autour de l'IST favori pour $\tau_S = 0.1ms$



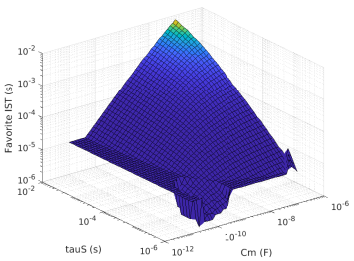
(c) IST favori pour $\tau_S = 0.1ms$



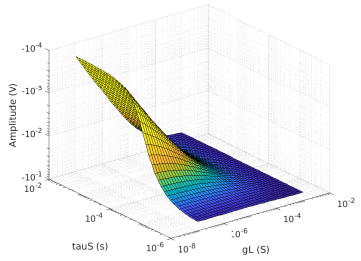
(d) Amplitude maximale avec l'IST favori pour $g_L = 10^{-5} S.cm^{-2}$



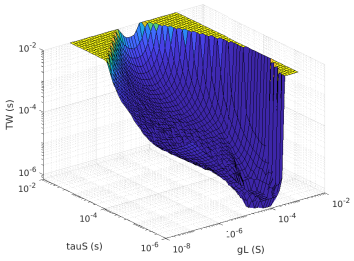
(e) Timewidth autour de l'IST favori pour $g_L = 10^{-5} S.cm^{-2}$



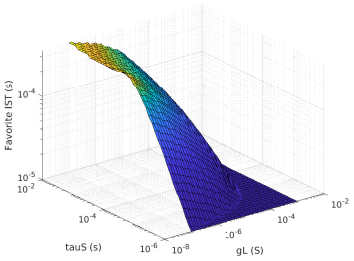
(f) IST favori pour $g_L = 10^{-5} S.cm^{-2}$



(g) Amplitude maximale avec l'IST favori pour $Cm = 10^{-9} F.cm^{-2}$



(h) Timewidth autour de l'IST favori pour $Cm = 10^{-9} F.cm^{-2}$



(i) IST favori pour $Cm = 10^{-9} F.cm^{-2}$

FIGURE 4.7 – Évolution de l'amplitude, l'IST favori et le TW en fonction de C_m, g_L et τ_s pour 1 IST

Favori, allant de quelques microsecondes jusqu'à plusieurs millisecondes, s'aligne avec les variations de C_m. Quand C_m augmente, l'IST Favori correspondant suit la même tendance. Si l'on examine les figures 4.7a et 4.7d, on peut également remarquer une tendance : une augmentation de C_m résulte en une diminution de l'amplitude maximale. Cela correspond pour nous à une détérioration de notre métrique, car cela rend notre système moins résistant au bruit s'ajoutant au signal. Ce comportement souligne l'importance du rôle que joue C_m dans notre modèle car c'est lui qui détermine la forme de la réponse caractéristique de notre neurone. En se penchant cette fois ci sur le TW, on voit que les figures 4.7b et 4.7e mettent en évidence le fait que lorsque C_m approche la valeur $8 \times 10^{-11}\,\mathrm{F.cm^{-2}}$, le TW démontre une diminution. Cette valeur particulière de C_m démontre la possibilité d'atteindre un TW plus fin, ce qui permettrait une identification plus précise de l'IST ciblé, au détriment d'être moins résistant au bruit.

— g_L : La conductance g_L entretient quant à elle un lien direct de proportionnalité avec la vitesse à laquelle la tension membranaire retourne à v_{rest} à la suite de la réception d'une impulsion. Cet impact apparaît dans les figures 4.7a et 4.7g, où on peut voir que l'augmentation de g_L entraîne une réduction de l'amplitude maximale. Cette observation met en lumière le fait que de grandes valeurs de g_L amplifient l'impact du phénomène de fuite. En conséquent, la tension revient à sa tension de repos plus rapidement.
En observant en détail les figures 4.7c et 4.7i, on peut observer l'évolution de l'IST favori. On peut voir que dans ces deux cas, lorsque g_L augmente, la valeur de l'IST favori diminue, indépendamment de l'autre paramètre observé. On peut également noter que sur les figures 4.7b et 4.7h le TW se resserre pour des valeurs de g_L avoisinant $4 \times 10^{-5}\,\mathrm{S.cm^{-2}}$. Ajuster g_L contribue donc à affiner la résolution d'identification des débits d'impulsions.

— τ_S : Ajuster τ_S permet de régler plus précisément le TW. À la réception d'une impulsion, g_s augmente instantanément jusqu'à avoir atteint sa borne supérieure, puis va redescendre plus lentement jusqu'à sa borne inférieure. Le vitesse à laquelle g_s redescend est régie par le paramètre τ_S. En effet, τ_S influence la dynamique de g_S, et agit en quelque sorte dans ce modèle comme un poids capable de pondérer l'impact d'une impulsion reçue. Lorsque τ_S prend une valeur plus faible, la décroissance de g_S est plus rapide, elle retourne à zéro plus vite. En conséquent, l'effet de la saturation dure moins longtemps. Il est particulièrement évident que g_S satures moins fort comparé à des scénarios où τ_S serait plus grand, comme on peut le voir sur les figures 4.7d et 4.7g.
Le rôle de τ_S étant de définir le gain en tension apporté par la seconde impulsion de notre séquence, elle influence de manière significative le TW. Une grande valeur de τ_S va engendrer une plus lente variation de g_S ce qui va engendrer un plus grand TW. En effet, si τ_S est plus grand, et donc que g_S reste près de sa borne supérieure plus longtemps, il faudra de plus grands écarts d'ISTs pour voir une différence sur l'amplitude, qui va diminuer plus lentement quand on s'éloigne de l'IST favori. De ce fait, le TW est plus large car les variations sont plus faibles. Donc, on exploite τ_S afin d'ajuster de manière plus précise la largeur de la fenêtre de TW, c'est à dire la précision nécessaire sur l'IST afin de déclencher une impulsion en sortie du neurone.
Cependant, les observations des figures 4.7f et 4.7i révèlent que modifier τ_s n'entraîne qu'une légère altération de l'IST favori, comparé à l'impact qu'ont les deux autres paramètres. En conséquent, nous avons opté pour ne dimensionner la valeur de τ_s qu'à la fin de notre processus d'optimisation, car il nous semblait important de fixer la largeur de TW après avoir à peu près fixé l'amplitude et l'IST favori, afin d'influencer le moins possible ces deux métriques.

Nous avons conclu de cette étude que le modèle SLIF est totalement adaptable et paramétrable pour permettre l'utilisation de ce modèle de neurone pour de la reconnaissance de signaux temporels, tout en n'utilisant pas de fonction de délai synaptique, difficilement reproductibles matériellement si l'on utilise seulement des technologies à très faible puissance. Toutefois, n'utiliser que des séquences à deux impulsions n'est clairement pas suffisant pour adresser des dizaines de noeuds IoT dans un environnement réel. Dans cette logique, nous nous sommes demandé ce qu'il adviendrait de notre système si maintenant on lui envoyait des séquences de plus de deux impulsions, et nous avons étudié le comportement de notre neurone pour ces plus longues séquences.

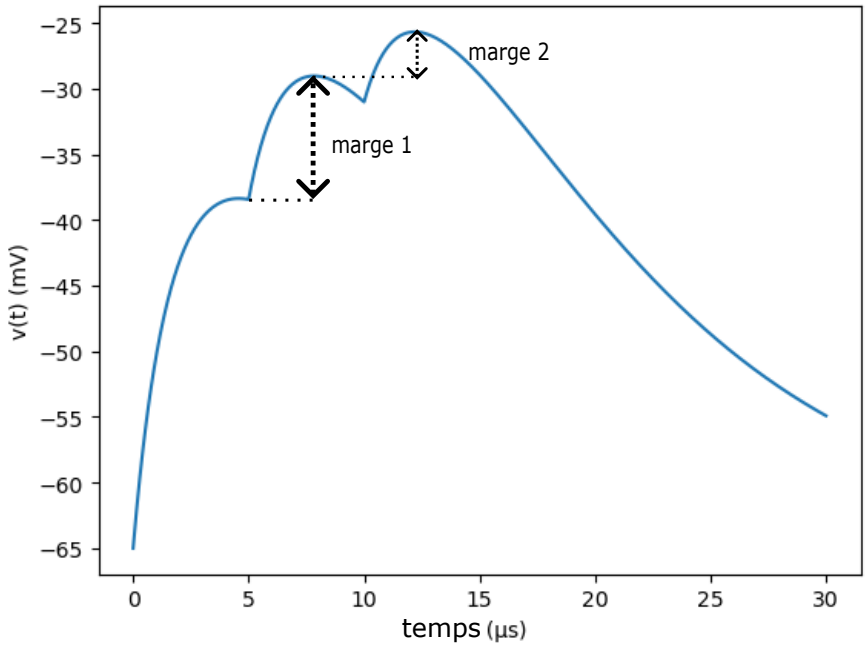


FIGURE 4.8 – Réponse du neurone à la réception de 3 impulsions avec deux ISTs identiques.

4.6 Comportement du SLIF pour des séquences à plusieurs ISTs

Notre but est maintenant de reconnaître des séquences spécifiques d'impulsions. Le seuil doit maintenant être atteint lorsque plusieurs impulsions, et potentiellement plusieurs ISTs différents, arrivent au neurone. Pour cela, nous estimons si nos métriques (amplitude, IST favori et TW) évoluent lors de la réception de plus de 2 impulsions, et si c'est le cas, leur tendance. Nous rappelons que l'IST est le temps séparant deux impulsions, ce qui implique qu'une séquence à n ISTs est une séquence qui comporte $n + 1$ impulsions.

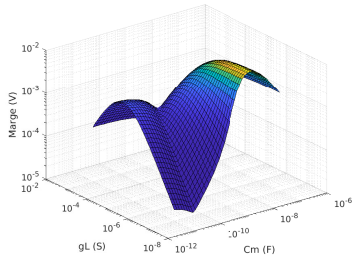
Sur la fig 4.8, nous avons tracé la réponse d'une neurone SLIF à une séquence de 3 impulsions séparées deux fois par un IST identique. Nous pouvons observer que le troisième lobe, engendré par la troisième impulsion, a une marge plus petite que le deuxième lobe. On en déduit donc que nos métriques ne dépendent pas seulement du dernier IST, mais aussi des ISTs précédents et du nombre d'impulsions reçues.

En conséquent, nous évaluons l'impact des paramètres de notre modèle sur les métriques mentionnées précédemment. L'amplitude représente l'amplitude totale obtenue après avoir intégré les trois impulsions. La marge est encore une fois la différence entre l'amplitude totale, et celle atteinte après l'avant dernière impulsion, ici la seconde. Le TW est déterminé sur le dernier lobe.

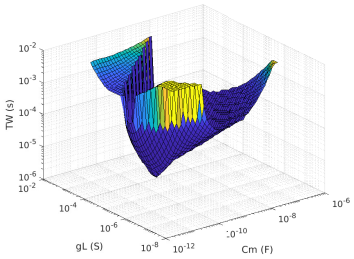
4.6.1 Séquences à 2 ISTs identiques

Dans cette simulation, nous commençons par envoyer une séquence de 3 impulsions, séparées par deux ISTs identiques, c'est à dire qu'il y a le même temps entre la première et la deuxième impulsion, qu'entre la deuxième et la troisième.

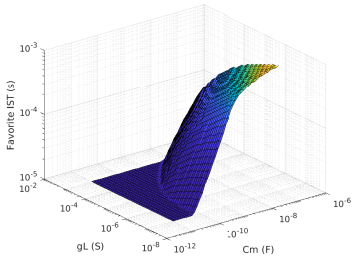
Les résultats que nous avons obtenus sont présentés en fig 4.9. Nous observons les mêmes tendances qu'en fig 4.7. Plus précisément, on peut noter que l'on a des valeurs



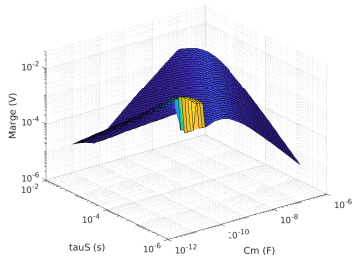
(a) Marge avec l'IST favori pour $\tau_S = 0.1ms$



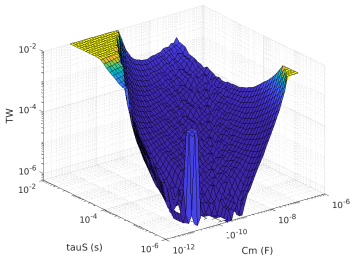
(b) Timewidth autour de l'IST favori pour $\tau_S = 0.1ms$



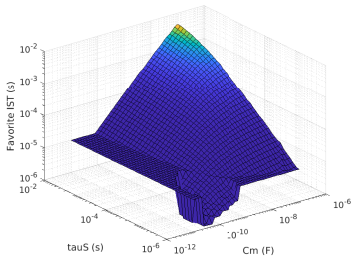
(c) IST favori pour $\tau_S = 0.1ms$



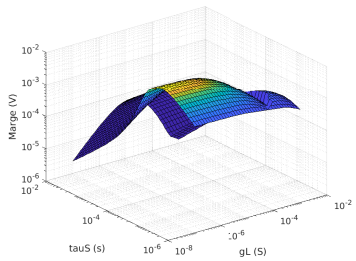
(d) Marge avec l'IST favori pour $g_L = 10^{-5}S.cm^{-2}$



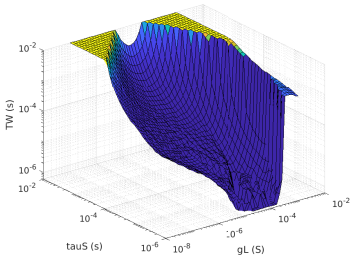
(e) Timewidth autour de l'IST favori pour $g_L = 10^{-5}S.cm^{-2}$



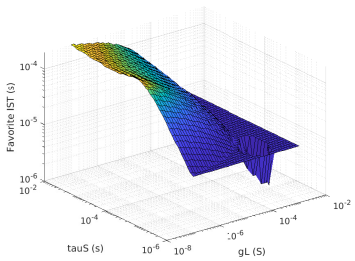
(f) IST favori pour $g_L = 10^{-5}S.cm^{-2}$



(g) Marge avec l'IST favori pour $Cm = 10^{-9}F.cm^{-2}$



(h) Timewidth autour de l'IST favori pour $Cm = 10^{-9}F.cm^{-2}$



(i) IST favori pour $Cm = 10^{-9}F.cm^{-2}$

FIGURE 4.9 – Évolution de l'amplitude, l'IST favori et le TW en fonction de C_m, g_L et τ_s pour 2 ISTs identiques

d'amplitude supérieures à ce que l'on avait, et un léger élargissement du TW.

Cependant, nous avons une augmentation de 50% du nombre d'impulsions, alors que l'amplitude reste dans le même ordre de valeurs avec seulement une petite augmentation. La marge démontre une diminution du même ordre de grandeur, ce qui correspond à ce que l'on pouvait observer en fig 4.8 lorsque la troisième impulsion arrivait en suivant un IST optimal. En conséquent, la marge pour la troisième impulsion se place à peu près aux deux tiers de la marge de la deuxième.

En explorant plus en profondeur les résultats du TW, on remarque que la forme de la courbe est exactement la même qu'en 4.7. Toutefois ces valeurs sont légèrement supérieures à celles obtenues précédemment ce qui montre que le TW s'élargit au fur et à mesure que l'IST augmente. On remarque aussi que g_L est toujours le paramètre qui a le plus d'impact sur l'évolution du TW.

Enfin, nous avons également observé un décalage dans les valeurs de l'IST favori, ce qui est cohérent et nous conforte dans l'idée qu'il ne parait pas optimal de choisir un

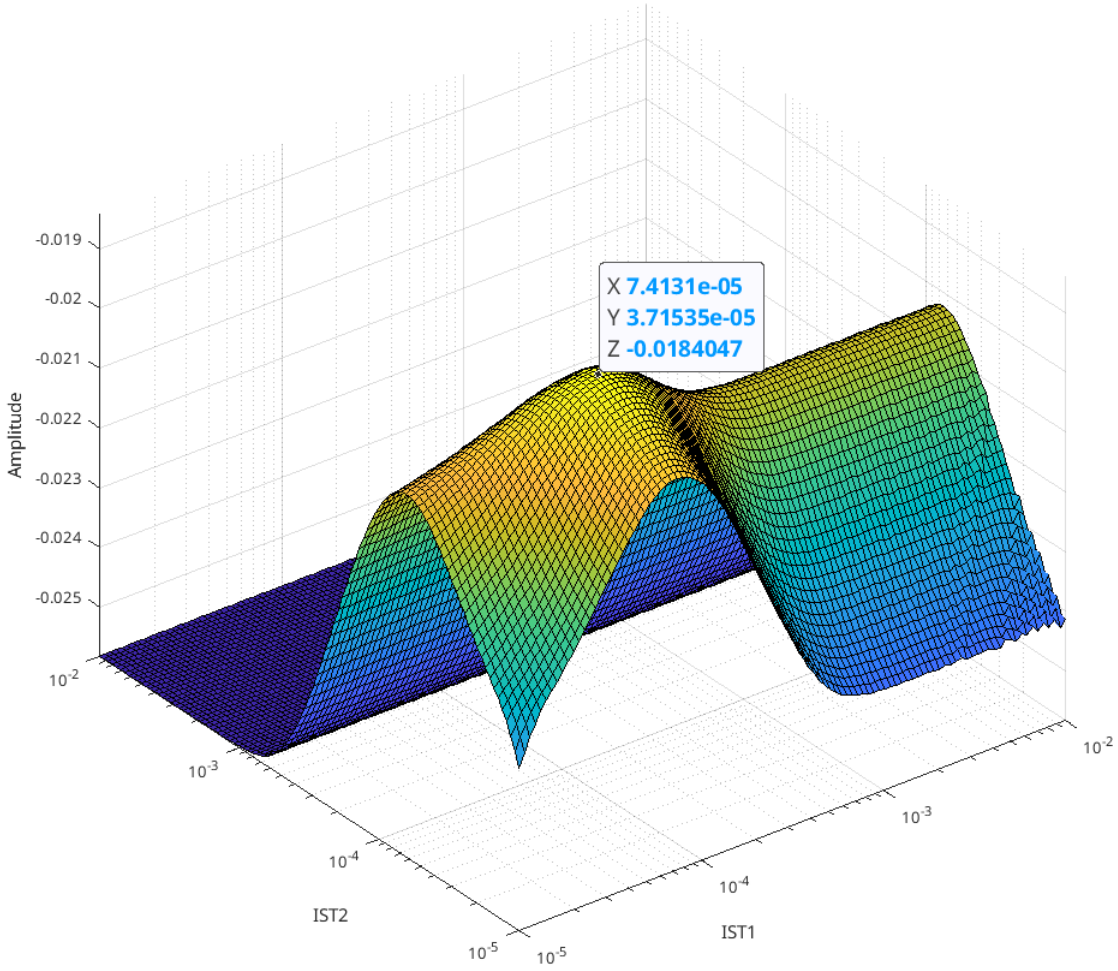


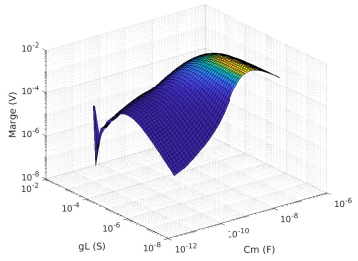
FIGURE 4.10 – Amplitude atteinte par un neurone SLIF en fonction des deux premiers IST

second IST égal au premier en termes d'amplitude.

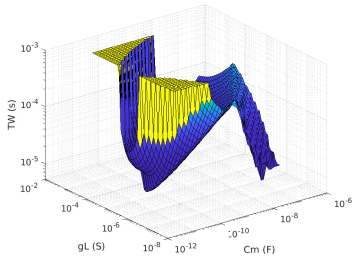
En résumé, le fait d'introduire une troisième impulsion avec un IST identique au premier amène à améliorer la sélectivité du système, mais aux dépends de la consommation, car on ajoute une impulsion. Ces observations nous ont permis de réaliser ce que pouvait nous apporter l'ajout d'un second IST, qui plus est un IST optimisé, différent du premier. Une telle approche peut potentiellement nous permettre de considérer de nouvelles séquences, tout en améliorant la sélectivité du système. Afin de valider ces hypothèses, nous allons maintenant explorer l'évolution de nos métriques pour des séquences de trois impulsions, mais cette fois ci avec un second IST optimisé par rapport aux métriques obtenues après la réception des deux premières impulsions.

4.6.2 Séquences à 2 ISTs optimisés

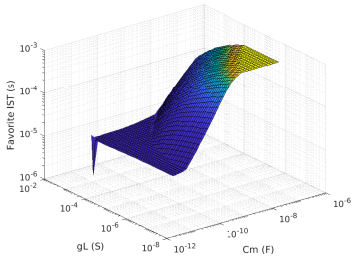
Notre objectif est de maximiser l'amplitude atteinte après la réception de deux impulsions. Pour cela, nous avons tout d'abord tracé l'évolution de cette métrique en fonction des deux ISTs en fig. 4.10. On remarque que l'amplitude la plus élevée est atteinte pour $IST_1 = 7.41 \times 10^{-5}$ et $IST_2 = 3.71 \times 10^{-5}$. Le premier IST correspond à celui qui donnait la plus grande amplitude sur la fig. 4.7, avec un seul IST. Cela signifie que les deux ISTs sont obtenus en choisissant le premier IST optimal en accord avec les données obtenues en fig 4.7 puis le deuxième est choisi de manière à optimiser l'amplitude totale. Grâce à



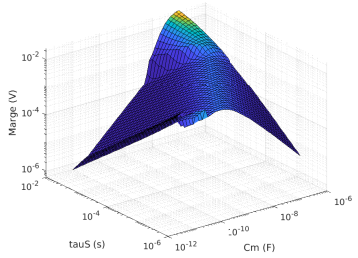
(a) Marge avec l'IST favori pour $\tau_S = 0.1ms$



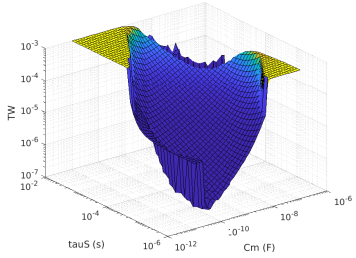
(b) Timewidth autour de l'IST favori pour $\tau_S = 0.1ms$



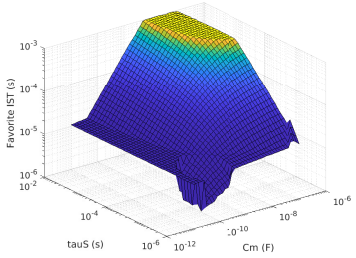
(c) IST favori pour $\tau_S = 0.1ms$



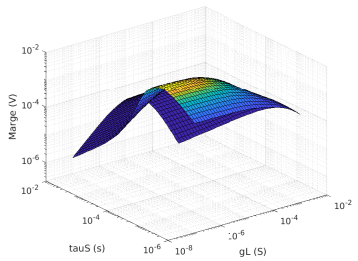
(d) Marge avec l'IST favori pour $g_L = 10^{-5}S.cm^{-2}$



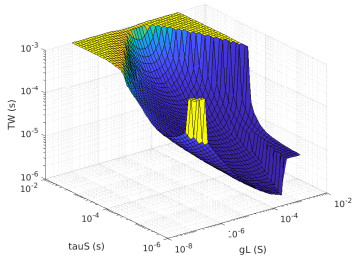
(e) Timewidth autour de l'IST favori pour $g_L = 10^{-5}S.cm^{-2}$



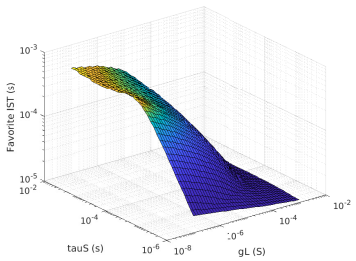
(f) IST favori pour $g_L = 10^{-5}S.cm^{-2}$



(g) Marge avec l'IST favori pour $Cm = 10^{-9}F.cm^{-2}$



(h) Timewidth autour de l'IST favori pour $Cm = 10^{-9}F.cm^{-2}$



(i) IST favori pour $Cm = 10^{-9}F.cm^{-2}$

FIGURE 4.11 – Évolution de l'amplitude, l'IST favori et le TW en fonction de C_m, g_L et τ_s pour 2 ISTs optimaux

cette propriété on peut déterminer les ISTs optimaux de manière itérative.

On va donc, dans cette nouvelle simulation, envoyer 3 impulsions avec des ISTs optimaux. Cela signifie que les deux ISTs seront différents, et choisis afin de maximiser l'amplitude totale. Nos résultats sont présentés en fig 4.11. Une fois de plus, on observe des tendances similaires à celles que l'on avait obtenues en fig 4.7. L'amplitude montre une évolution peu conséquente, mais se rapproche quand même de $E_S = 0$ mV, qui représente la tension maximale atteignable par $v(t)$ dans notre modèle. On peut également voir que la marge subit une augmentation ressemblant à celle du scénario à 1 IST, atteignant jusqu'à approximativement 13mV.

La courbe représentant l'IST favori présente cette fois ci aussi la même tendance qu'en 1) mais avec un décalage encore un peu plus grand, ce qui confirme notre décision de ne pas utiliser de valeurs d'ISTs identiques pour déterminer notre séquence.

Pour ce qui est du TW, on observe des valeurs très proches de celles présentées en fig 4.7. Cette tendance montre un plus haut niveau de sélectivité lorsque le second IST

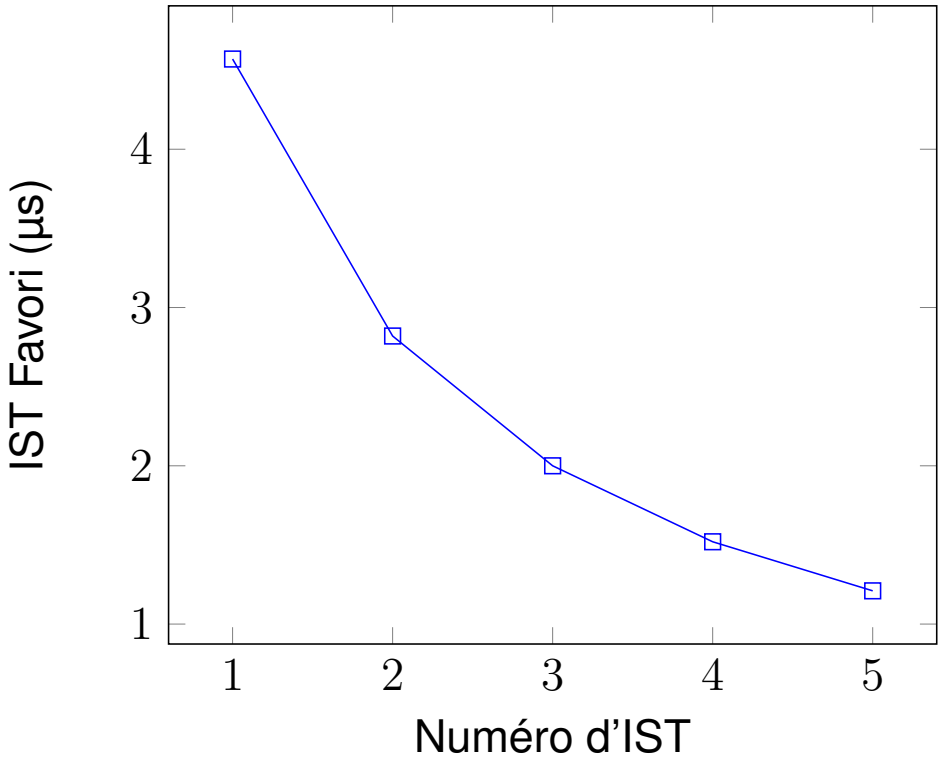


FIGURE 4.12 – Évolution de l'IST favori en fonction du numéro d'IST

est choisi de manière à optimiser l'amplitude, comparé au cas en 1). Ce résultat est principalement dû au fait que la marge est plus haute dans ce cas ci, et donc qu'atteindre la tension de seuil nécessite plus de précision sur le second IST favori.

Nos résultats ont mis en évidence l'intérêt d'optimiser la valeur de $2nd$ IST par rapport à la marge, et donc l'amplitude. Cette approche stratégique nous a permis d'améliorer la sélectivité de notre système, et donc d'obtenir de meilleurs résultats pour des opérations temporelles. De plus, on sait maintenant que si on utilise un nombre d'ISTs supérieur à 2, il nous faudra réaliser une nouvelle étude de ces métriques afin de tirer le plein potentiel de notre système en termes de sélectivité et de capacité d'opérations, contrairement au cas où l'on aurait utilisé des ISTs successifs identiques.

4.6.3 Séquences à plus de 2 ISTs

Dans l'optique d'aller un peu plus loin, nous avons réalisé cette étude de nos métriques pour plus de trois impulsions.

Pour réaliser ces simulations, nous avons cherché dans les courbes de la fig. 4.7 les paramètres qui donnaient le neurone au TW le plus étroit. Ensuite, pour chaque impulsion suivante, on détermine le n^{ime} IST favori et on l'utilise pour déterminer le $(n+1)^{ime}$. Le but est de nous permettre d'utiliser un maximum d'ISTs différents dont les TWs ne se chevauchent pas, et d'avoir chaque neurone le plus sélectif possible. Les paramètres de ce neurones sont les suivants : $C_m = 10^{-10}\,\mathrm{F}\cdot\mathrm{cm}^{-2}$, $g_L = 10^{-5}\,\mathrm{S}\cdot\mathrm{cm}^{-2}$, et $\tau_s = 3\times10^{-6}\,\mathrm{s}$.

Une fois de plus, notre décision de ne pas utiliser des ISTs identiques est confirmée par ce qui est tracé en fig 4.12, car on observe que les valeurs d'IST favori successifs tendent à décroître au fil des impulsions. De plus, la fig 4.13 illustre une décroissance dans les valeurs de marge, au cours de l'ajout d'impulsions. Ce phénomène s'explique par

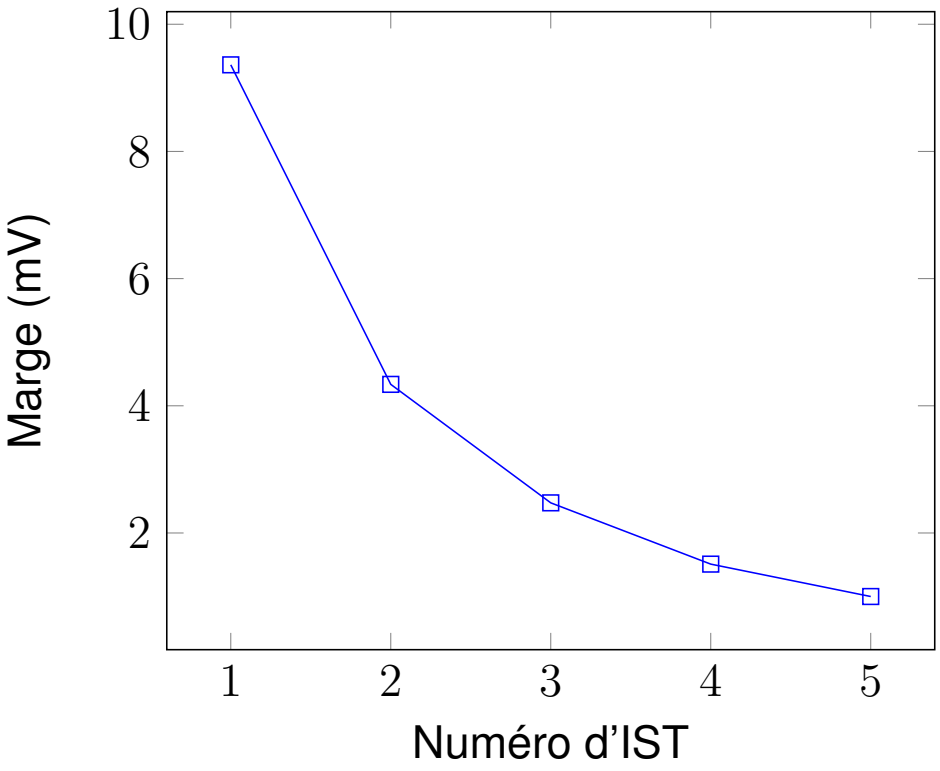


FIGURE 4.13 – Marge due au dernier IST favori en fonction du numéro d'IST

l'objectif de notre modèle, s'approcher de $E_S = 0\,\text{mV}$ à chaque impulsion.

Cela entraîne qu'à force de s'approcher petit à petit de E_S, nous devrions atteindre un point à partir duquel notre marge tombe en dessous de $0.1\,\text{mV}$ après un certain nombre d'impulsions. Or, la tension de seuil est actuellement placée $0.1\,\text{mV}$ en dessous de l'amplitude. Dans ce cas, la tension de seuil diminuerai au passage à l'IST suivant.

En conséquent, tous les $n + 1^{ime}$ ISTs feraient atteindre la tension de seuil si les n premiers sont choisis de manière optimale, et ils feraient tous apparaître une impulsion en sortie. Cela nous amène donc à considérer qu'il existe un nombre maximal d'ISTs utilisables dans une séquence, déterminable en fonction de la tension de seuil et du jeu de paramètres utilisé.

La tension de seuil influence grandement les performances du neurone en termes de sélectivité. Afin d'avoir une meilleure idée de cette sélectivité, nous allons évaluer l'influence de la tension de seuil et du nombre d'impulsion dans la séquence envoyée sur l'apparition de Fausses Alarmes et de Misdetections.

4.7 Performances

Dans cette partie, nous évaluons les performances des séquences obtenues dans la partie précédente, lorsqu'elles sont utilisées comme séquence d'activation pour des systèmes de WuR.

Notre objectif est de mesurer la robustesse de notre neurone pour reconnaître une séquence en présence de perturbations.

Notre objectif est de réagir à une unique séquence spécifique, et ne pas réagir le reste du temps. Malheureusement, le signal sera soumis à des perturbations provenant de

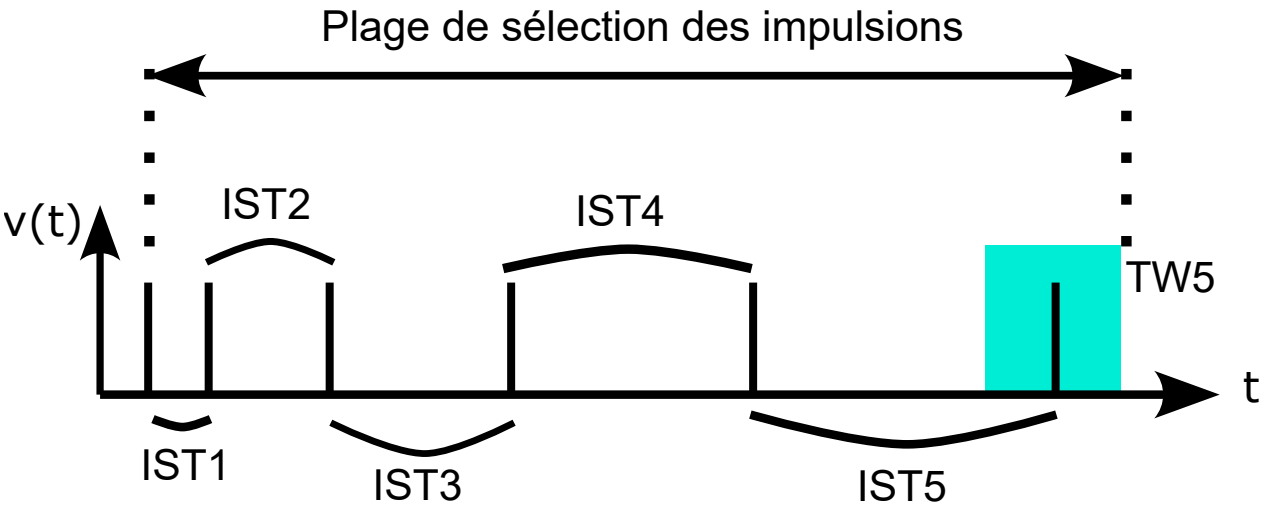


FIGURE 4.14 – Illustration de la plage d'ISTs dans laquelle on pioche les timings d'impulsion pour les simulations de FA

l'environnement qui font que l'on pourrait rater la séquence, c'est une Misdetection (MD) ou qui pourrait nous faire croire que l'on a détecté la séquence alors que l'on a reçu que du bruit, c'est une Fausse Alarme (FA). Pour cela, nous continuons d'utiliser les paramètres mis en avant dans la dernière sous-partie, ainsi que les ISTs favoris correspondants. Nous évaluons la probabilité de Fausses Alarmes (FA), puis la probabilité de Misdetections (MD). Dans ces deux scénarios, on considère un noeud IoT qui reçoit des séquences depuis une antenne, doit être réveillé si il reçoit exactement la séquence ciblée, et rester en veille s'il reçoit n'importe quelle autre séquence. Dans ces deux scénarios, on utilise des séquences à 5 ISTs, ce qui veut dire que l'on envoie 6 impulsions pour chaque séquence.

Notre objectif ici est multiple. Dans un premier temps, nous souhaitons identifier une valeur de tension de seuil optimale qui nous amènerait aux meilleures performances de reconnaissance. Ensuite, nous cherchons à identifier le nombre maximal d'impulsions qu'il est possible d'utiliser dans une séquence, sans que la sélectivité ne soit dégradée en dessous d'un niveau acceptable. Enfin, nous allons estimer le gain en termes de puissance consommée par rapport à des récepteurs classiques.

4.7.1 Fausses Alarmes

Nous voulons mesurer la probabilité de FA du processus de reconnaissance de notre neurone. Pour cela, nous avons imaginé un scénario dans lequel les séquences sont créées de manière aléatoire, et nous analysons quelles séquences parmi celles-ci sont détectées. Nous présentons ici notre analyse basée sur des simulations et des approximations théoriques.

Pour nos simulations, nous sommes partis de la séquence composée des ISTs favoris du neurone considéré, puis nous avons créé une plage de temps d'impulsions allant du début du temps de simulation jusqu'à la fin du TW de la dernière impulsion composant cette séquence, en bleu sur la fig. 4.14. Avec notre jeu de paramètres, où le 5^{ime} temps d'impulsion est $1.25\times10^{-5}s$ et son TW est $0.08\times10^{-5}s$, la borne supérieure de notre plage est $T_{tot} = \sum_{i=1}^{5} IST_i + \frac{TW_5}{2} = 1.29 \times 10^{-5}s$. Au sein de cette plage, on tire aléatoirement 5 temps d'impulsions avec une probabilité uniforme, on les trie par ordre croissant, et on les utilise comme les temps d'impulsions pour notre séquence d'entrée. Le début de la séquence est fixé à $t = 0s$, puis on y ajoute les 5 temps obtenus aléatoirement. Une FA

TABLE 4.2 – Paramètres des neurones et leurs métriques

set	$C_m(F.cm^{-2})$	$g_L(S.cm^{-2})$	$\tau_S(s)$	amplitude (mV)	$IST_1(s)$	$IST_2(s)$	$IST_3(s)$	$IST_4(s)$	$IST_5(s)$
TW Etroit	10^{-10}	10^{-5}	3×10^{-6}	-19.87	4.57×10^{-6}	7.39×10^{-6}	10^{-5}	1.15×10^{-5}	1.25×10^{-5}
TW Moyen	10^{-9}	10^{-5}	10^{-4}	-16.78	7.41×10^{-5}	1.10×10^{-4}	1.33×10^{-4}	1.49×10^{-4}	1.62×10^{-4}
TW Large	10^{-8}	10^{-4}	10^{-3}	-49.66	2.31×10^{-4}	3.33×10^{-4}	3.99×10^{-4}	4.46×10^{-4}	4.83×10^{-4}

est enregistrée si la tension maximale atteinte au cours du traitement de la séquence ainsi obtenue est supérieure à un seuil préétabli.

En simulation, nous avons réalisé des tests pour cinq valeurs de marge de seuil. La marge de seuil correspond à l'écart entre l'amplitude atteinte avec la séquence composée uniquement des ISTs favoris, donc la plus grande amplitude possible pour le jeu de paramètres considéré, et la tension de seuil. Jusqu'à présent nous aurions placé ce seuil $10^{-4}V$ en dessous de l'amplitude, qui était alors notre marge de seuil. Ici nous allons réaliser ces tests pour les valeurs de marge de seuil suivantes : $10^{-5}V$, $3\times10^{-5}V$, $10^{-4}V$, $3\times10^{-4}V$, et $10^{-3}V$. Nous avons tracé les résultats pour ces cinq marges de seuil, et pour trois jeux de paramètres différents. Ces derniers ont été choisis ainsi pour représenter 3 neurones dont la principale différence est le rapport entre l'IST et le TW qui représente la sélectivité du neurone. La courbe bleue correspond à un neurone ayant le TW le plus fin, donc le plus sélectif, la courbe orange représente les résultats pour un neurone au TW de largeur moyenne, et la courbe verte correspond au neurone ayant le TW le plus large. Les jeux de paramètres correspondant à ces trois neurones sont donnés en table 4.2.

Les résultats visibles en fig 4.15 illustrent que la probabilité de FA croît lorsque le seuil s'éloigne de l'amplitude maximale. Cette tendance est en accord avec nos attentes, car réduire la sélectivité entraîne une plus grande chance d'atteindre le seuil pour des tensions membranaires plus faibles, elles mêmes entraînées par des séquences dont les temps d'impulsions sont un peu plus éloignés de ceux de la séquence optimale. Il y a notamment un point d'inflexion après la marge de seuil à $10^{-4}V$, ce qui indique l'intérêt de cette valeur dans la minimisation de la probabilité de FA.

On peut également observer que ces courbes sont tracées de haut en bas (vert vers bleu), ce qui correspond à l'ordre du TW le plus large au TW le plus fin, ce qui est cohérent avec l'idée selon laquelle le TW le plus fin est celui qui permet d'avoir le moins de FA, et inversement.

De plus, nous avons réalisé une analyse théorique de la probabilité de FA. Pour cela, nous avons fait l'hypothèse selon laquelle, on a la présence d'une FA si on tire aléatoirement un temps d'impulsions dans chaque TW. Par exemple en fig. 4.14, si les 4 premiers timings d'impulsions ont été tirés dans leurs TWs respectifs, il faut que le cinquième soit dans le $TW5$ représenté en bleu. Nous avons donc calculé la probabilité d'obtenir exactement un temps dans chaque TW distinct à l'intérieur de la plage dans laquelle les valeurs ont été tirées. Cette probabilité est exprimée par :

$$P(FA) = \frac{\prod_{i=1}^{n} TW_i \times n!}{(T_{tot})^n} \tag{4.5}$$

Ici, n représente le nombre de temps d'impulsion à tirer, TW_i correspond au TW du i^{me} IST basé sur l'IST favori correspondant, et T_{tot} est la largeur de la plage dans laquelle on tire ces temps.

Il est essentiel de remarquer que (4.5) donne une approximation, car il y a quelques rares cas dans lesquels on tire moins de n impulsions à l'intérieur des TWs, et où on

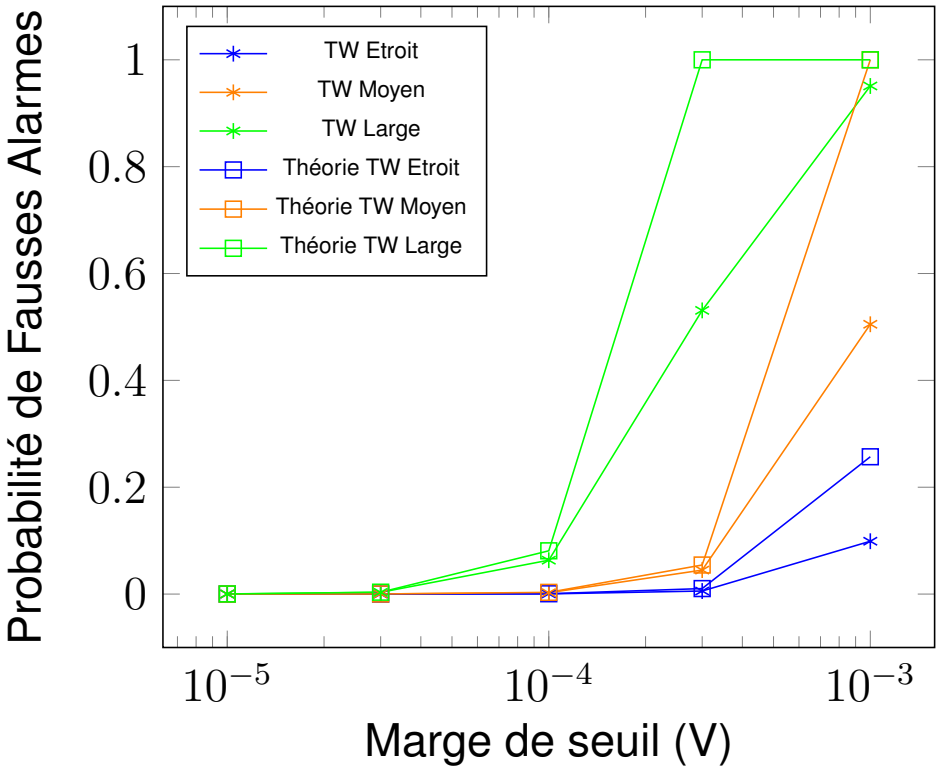


FIGURE 4.15 – P(FA) pour 5 marge de seuils et 5 ISTs, pour 3 jeux de paramètres différents, en simulation et en théorie

atteint quand même le seuil avec $n-1$ timings parmi les n qui sont bien placés, si ces $n-1$ sont extrêmement proches de l'IST favori, et le dernier est proche du TW visé. Il y a également de rares cas où les n impulsions sont dans les TWs ciblés, mais sont tous à une extrémité de ce TW et trop loin de l'IST favori, ce qui peut empêcher d'atteindre le seuil même si l'on était dans un scénario qui aurait dû correspondre à une FA.

Nous avons rajouté les valeurs théoriques du scénario précédent sur la fig 4.15. On peut y observer que les valeurs théoriques de P(FA) suivent la tendance des courbes obtenues en simulation, tout en restant constamment légèrement supérieures. Cela est dû au fait que, lorsqu'on génère nos ISTs aléatoirement à l'intérieur des TWs, mais aux extrémités de ces TWs, la théorie va compter une FA, mais en simulation on ne dépassera pas toujours le seuil.

D'ailleurs, certaines valeurs théoriques sont bornées à 1 à cause du fait que lorsqu'on utilise une marge de seuil de $10^{-3}V$, le TW résultant peut excéder la durée totale de la plage T_{tot}.

On peut donc utiliser l'eq. 4.5 comme une borne supérieure de la probabilité de Fausses Alarmes.

4.7.2 Misdetections

Une Misdetection (MD) survient lorsqu'on a une mauvaise calibration de fréquence ou un décalage fréquentiel entre le transmetteur et le récepteur.

Pour mesurer la probabilité de MD, nous avons réalisé des simulations comportant un jitter fréquentiel. Pour cela nous avons tiré aléatoirement une valeur de jitter d'une distribution normale centrée autour de 1, avec un écart type (σ) de 0.1. On multiplie

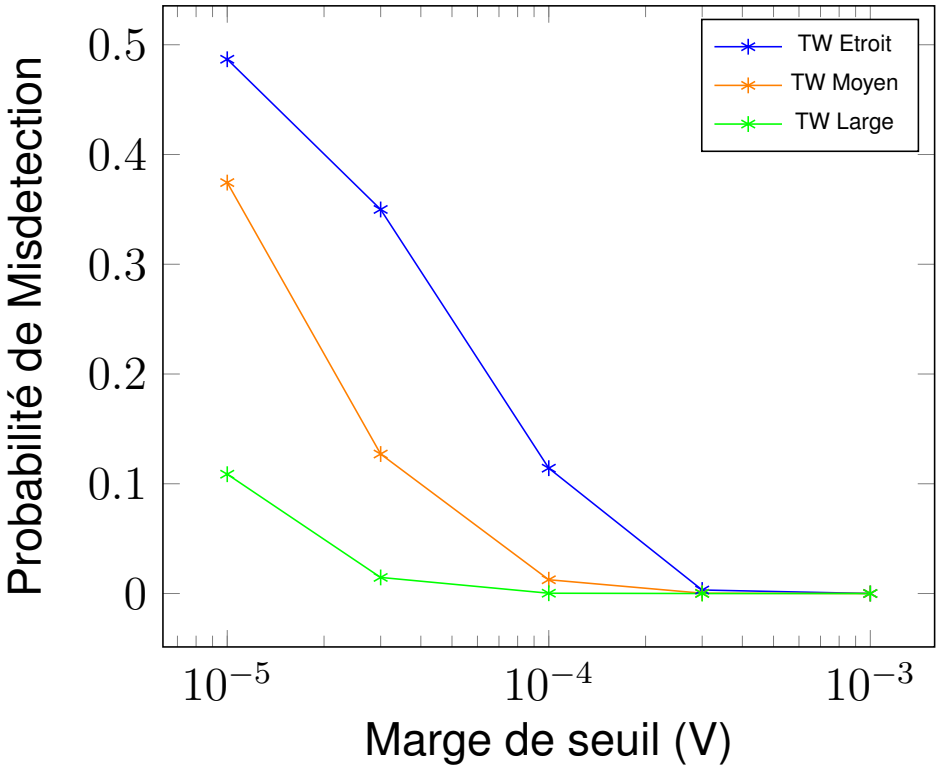


FIGURE 4.16 – P(MD) pour 5 marges de seuil différentes, 5 ISTs, sigma = 0.1 et pour 3 jeux de paramètres différents

ensuite tous les ISTs favoris successifs par cette valeur aléatoire. Ensuite, on envoie cette nouvelle séquence à notre neurone, et on regarde si la tension membranaire atteint le seuil préétabli. Si la tension n'atteint pas ce seuil, on enregistre une MD. Ce scénario a été simulé 100000 fois pour chacune des 5 marges de seuil déterminées pour le scénario des FAs. L'ordre de grandeur de la probabilité de MD dépend de sigma, fixé ici à 0.1 pour illustrer la tendance. Si on considérait des valeurs plus faibles et plus réalistes de *sigma*, nous obtiendrions des ordres de grandeurs également plus faibles.

La figure 4.16 donne des intuitions sur la probabilité de MD, révélant une décroissance lorsque le tension de seuil s'éloigne de l'amplitude maximale. Cette tendance tire racine du fait qu'accepter une sélectivité amoindrie nous donne des TWs plus larges. De ce fait, les temps d'impulsions seront toujours dans leurs TWs respectifs même si la valeur de jitter s'éloigne un peu plus de 1.

Dans ce scénario, il faut noter que les courbes présentées progressent du haut vers le bas, du plus fin TW au plus large, de la courbe bleu à la courbe verte. Ce schéma souligne notamment qu'un TW plus sélectif correspond à une plus haute probabilité de FA, ce qui est exactement la tendance inverse au comportement observé pour la probabilité de MD.

La probabilité de MD commence à décroître après la marge de seuil de $3 \times 10^{-4}V$. Il est intéressant de remarquer que la probabilité de FA montre la même tendance pour les valeurs inférieures à 10^{-4}, puis augmente à partir de la marge à 3×10^{-4}. Pour optimiser à la fois P(FA) et P(MD) pour un même jeu de paramètres, il serait judicieux de choisir une marge de seuil qui serait située entre entre ces deux valeurs.

Ainsi, notre exploration du comportement de notre neurone dans différents scénarios a permis de mettre en lumière le compromis entre l'optimisation de la probabilité de FA et de celle de MD sur le choix de la marge de seuil. Un seuil plus proche de l'amplitude

va augmenter la sensitivité, ce qui réduira la probabilité de FA mais augmentera celle de MD. À l'inverse, un seuil plus éloigné de l'amplitude va améliorer la spécificité, diminuant donc le nombre de MDs tout en causant une hausse de la probabilité de FA. Cela montre que le choix de la marge de seuil joue un rôle critique dans la réalisation de ce compromis. Trouver un équilibre optimal entre ces paramètres en concurrence est une des clés pour réussir à ce que notre modèle neuronal puisse réaliser des opérations temporelles robustes et précises.

4.8 Conclusion

Ce chapitre a porté sur la dynamique particulière du modèle de neurone LIF Saturant (SLIF), et explore son comportement en réponse à des conditions et des paramètres variés. Au cours d'une observation méticuleuse des interactions entre le ou les ISTs de nos séquences, la tension de seuil, et la précision de reconnaissance, nous avons découvert quelques pistes.

Premièrement, nous avons réalisé une étude en profondeur de ce nouveau modèle, d'abord en examinant le cas d'un seul IST, puis ensuite en étendant notre analyse à de multiples ISTs par séquence. Pour cela, nous avons évalué de manière exhaustive plusieurs métriques à travers une large plage de paramètres, montrant la versatilité du modèle à opérer efficacement dans plusieurs ordres de grandeurs, à la fois dans le domaine de l'IoT et potentiellement dans des applications biologiques.

Nous avons examiné individuellement l'impact de plusieurs paramètres, comme capacité de membrane (C_m), la conductance de membrane (g_L) ou la constante de temps synaptique (τ_s), sur la réponse caractéristique du neurone. Notamment, g_L influence l'amplitude, la marge et le TW, mettant en évidence son rôle dans l'évolution des performances de reconnaissance. De plus, τ_s affecte significativement le TW, démontrant son influence sur la sélectivité de la séquence.

Pour aller plus loin, nous nous sommes penchés sur les effets de l'introduction d'une troisième impulsion avec un même IST, ce qui nous a ouvert à la possibilité d'étendre nos séquences malgré une sélectivité réduite. Nos découvertes nous ont permis de souligner l'importance de l'optimisation successives des différents ISTs lorsque l'on cherche à améliorer les performances, plutôt que de les garder constants. De plus, notre analyse indique que plus on dépasse les deux ISTs, plus l'adaptation des ISTs devient primordiale pour maintenir une sélectivité et une capacité de reconnaissance optimales.

Pour une évaluation complète de la robustesse du neurone à des perturbations, nous avons évalué les probabilités de Fausses Alarmes (FA) et de Misdetections (MD) en utilisant des scénarios comportant de l'aléatoire. En ajustant les valeurs de tension de seuil et en examinant leur impact sur les FAs et les MDs, nous avons découvert l'existence d'un compromis entre l'optimisation de ces métriques. Bien choisir où placer la tension de seuil peut d'avérer crucial, car choisir un seuil trop près de l'amplitude maximale va minimiser le nombre de MD, mais augmenter le nombre de FA, et vice versa.

Notre étude a mis en évidence l'interaction entre les paramètres du neurone, les ISTs des séquences, et la précision de reconnaissance. Cette exploration nous a fourni les fondations pour une optimisation plus précise et plus en profondeur du modèle neuronal SLIF pour des possibles applications concrètes. Tout en continuant à démêler les complexités de la dynamique des neurones, nous nous attendons à ce que nos découvertes contribuent à l'avancée des possibilités des réseaux de neurones artificiels à réaliser des opérations sur des signaux temporels. Malheureusement, aucune comparaison avec d'autres travaux

n'ont pu être réalisées car, autant que nous sachions, aucun autre travail n'a été réalisé sur des neurones SLIF opérants sur des séquences basées sur des ISTs.

Pour compléter, de nos jours la puissance consommée par une architecture de neurones avoisine les $100pW$ par neurone [11]. Donc, pour identifier une séquence basée sur un seul IST, la puissance consommée par un neurone SLIF se placerait entre $100pW$ et $1nW$. Cela représente une réduction significative comparé aux pics de puissance des récepteurs de réveil, qui s'élèvent à près de $200\mu W$ [5].

Ainsi, à l'aide de ce nouvel outil qu'est le modèle SLIF, il va maintenant nous être possible de créer un réseau de neurone capable de réaliser des opérations sur des signaux temporels, en s'affranchissant du phénomène de délai synaptique, compliqué à réaliser sur une architecture matérielle, en gardant une architecture très faible puissance. Il nous sera également intéressant d'étudier un moyen de modifier les paramètres afin d'automatiquement changer d'ordre de grandeur en termes d'IST sans dégrader complètement les métriques, dans le but de pouvoir s'ouvrir à des applications plus complexes et versatiles.

Chapitre 5

Réseau de neurones SLIF : Topologie en ligne

5.1 Introduction

Après avoir considéré des structures plus simples dans les chapitres précédents, nous étudions ici une structure en réseau. Les réseaux de neurones sont une technologie en plein essor, de plus en plus appliquée dans des domaines variés. Ils sont désormais utilisés pour résoudre une multitude de problèmes, allant des tâches complexes en intelligence artificielle à des applications pratiques du quotidien.

Ces réseaux de neurones sont composés d'un très grand nombre de micro-ordinateurs souvent chargés de réaliser une seule tache i.e. transformer son entrée à l'aide de sa fonction de transfert, puis transmettre sa sortie au neurone artificiel suivant en le pondérant de manière à reproduire le phénomène de plasticité synaptique. Dans notre cas, nous utilisons des neurones bio-inspirés, qui fonctionnent donc comme des neurones biologiques, en utilisant des impulsions plutôt que des nombres. Toutefois, si l'on veut réaliser des opérations complexes avec des neurones, bio-inspirés ou non, il nous faut les utiliser en réseau.

Nous avons aussi vu dans la partie précédente qu'il est possible d'utiliser un seul neurone SLIF pour reconnaître une séquence particulière, et discriminer toutes les autres. Cependant, nous avons également pu observer que l'utilisation d'un seul neurone se confronte très rapidement à une limitation, car la sélectivité du neurone quant à la reconnaissance de séquence d'au moins 5 ISTs, dans notre exemple, devenait très mauvaise. Cela était dû au fait que l'incrément sur la tension membranaire due à la dernière impulsion était plus petite que la marge de tension de seuil permettant la sélectivité.

Nous avons vu dans le chapitre 4 que la sélectivité d'un neurone est bonne pour le premier IST mais se dégrade rapidement pour plus d'ISTs. Dans ce chapitre nous allons donc seulement exploiter la réaction des neurones à un unique IST.

Nous avons mis au point une autre approche pour à la fois détecter des séquences plus complexes et améliorer la robustesse de notre système de reconnaissance. Pour cela, nous avons conçu un réseau constitué de plusieurs neurones SLIF.

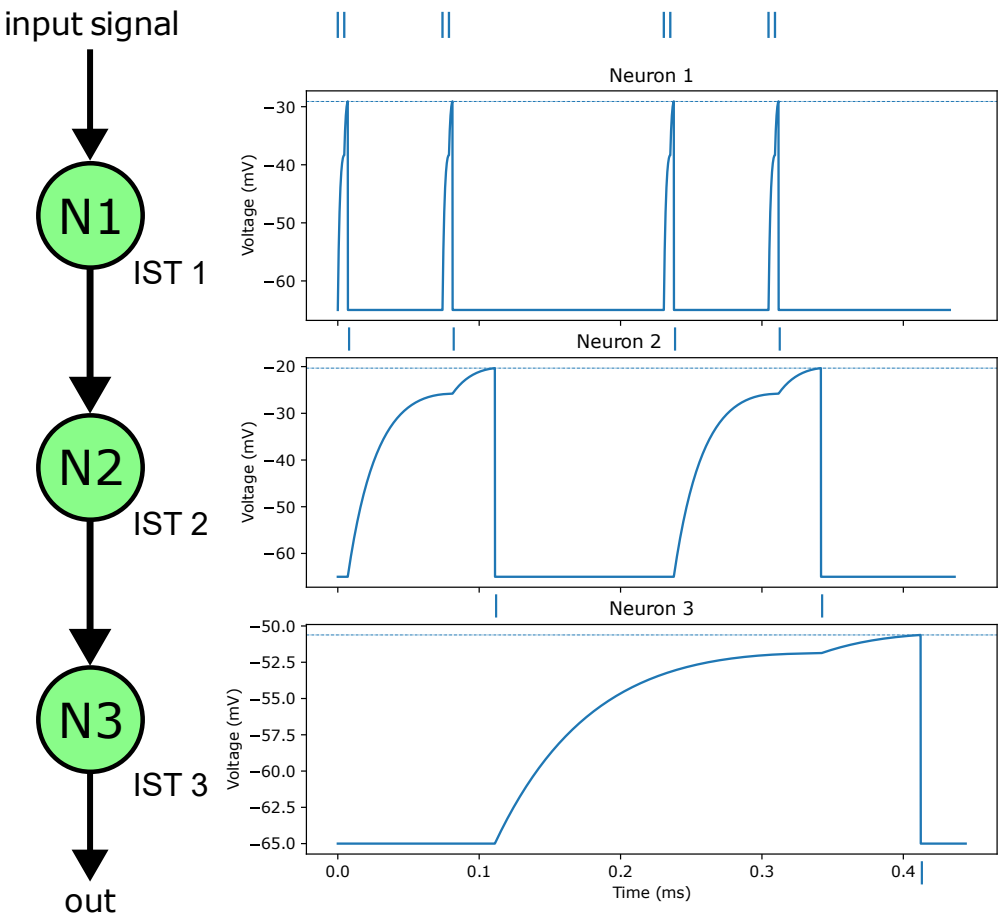


FIGURE 5.1 – Fonctionnement du réseau de SLIF

5.2 Première topologie : Réseau en ligne

5.2.1 Modèle de topologie

Dans la première partie de cette étude, nous allons nous focaliser sur l'évaluation d'un réseau constitué de trois neurones SLIF en série, chacun ayant un IST favori différent, comme on peut le voir sur la partie gauche de la fig. 5.1. La propagation des impulsions se fait d'une manière totalement feedforward. La création de l'architecture de notre réseau et le choix de la séquence d'activation du réseau considérés doivent être réalisés conjointement.

En effet, chaque neurone a un IST favori, et va donc être utilisé pour reconnaître des morceaux de séquences de deux impulsions, qui seront discriminées en fonction du délai entre ces deux impulsions, l'IST. Le premier neurone de notre réseau va donc lire chaque paire d'impulsions et propager une impulsion si cette sous-séquence correspondait à l'IST visé. Le suivant va ensuite fonctionner de la même façon avec les impulsions propagées. Chaque étage va donc diviser le nombre d'impulsions par deux si on reçoit la bonne séquence au global. En toute logique, pour n neurones dans notre réseau, il nous faudra donc 2^n impulsions dans la séquence d'entrée pour que notre dernier neurone envoie une unique impulsion à la réception de la bonne séquence.

Notons IST_i, l'IST favori du neurone N_i. Le neurone N_i va donc propager une impulsion au neurone suivant s'il reçoit une paire d'impulsions séparées de IST_i, et ne rien

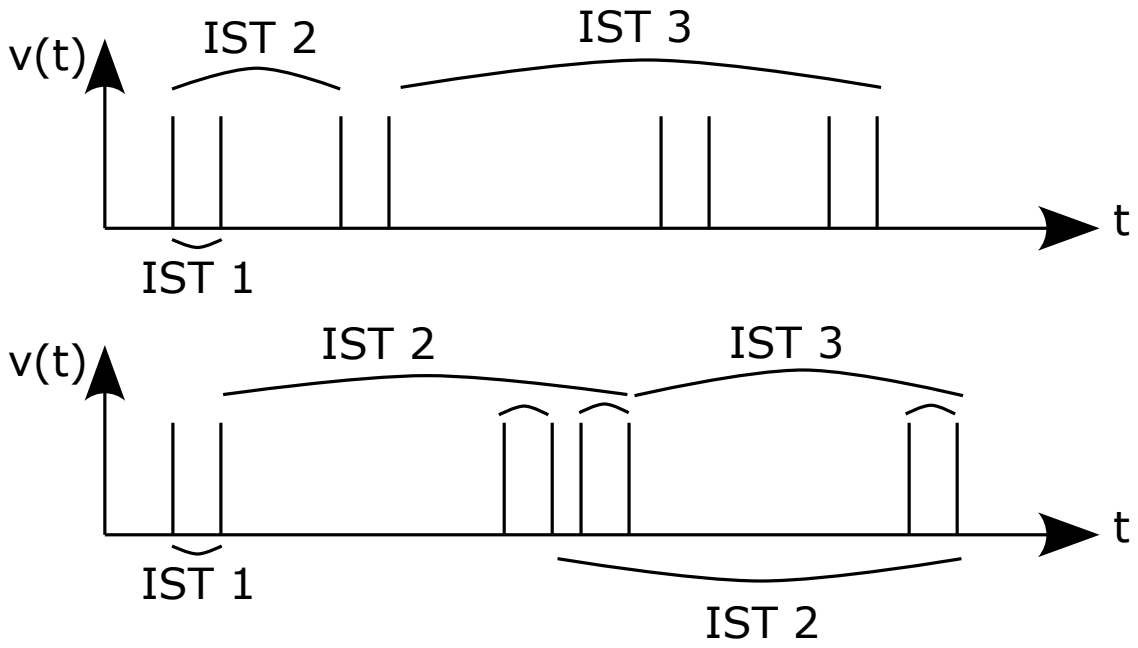


FIGURE 5.2 – Exemple de séquences respectant (en haut) et ne respectant pas (en bas) la contrainte sur les ISTs

faire sinon. Dans la fig 5.1, notre réseau de 3 neurones traite une séquence de 8 impulsions. En examinant la première ligne, nous observons la séquence envoyée au neurone N_1. Celle-ci est composée de quatre paires d'impulsions. Dans chaque paire, les impulsions sont séparées par un IST de IST_1, l'IST favori de N_1. Les deux premières paires et les deux dernières sont espacées entre elles de IST_2, IST favori du second neurone. Enfin, la deuxième et la quatrième sont espacées de IST_3. N_1 reçoit donc les quatre paires tour à tour, ce qui entraîne la transmission de quatre impulsions au neurone N_2. Lorsqu'il reçoit ces impulsions, le neurone N_2 les interprète à son tour comme deux paires, reçues avec le bon intervalle de temps, et envoie ensuite deux impulsions au neurone N_3, espacées par IST_3. Ce processus garantit que le neurone final décharge une fois en cas de réception de la séquence ciblée.

Une contrainte clé dans ce modèle est la nécessité que les ISTs successifs soient chacun au moins le double du précédent. Cette contrainte découle du risque de chevauchement entre plusieurs paires d'impulsions, ce qui transformerait la séquence de manière à ce qu'elle ne respecte plus la règle de conception définie précédemment. On peut voir en haut de la fig. 5.2 un exemple de séquence bien construite respectant la contrainte. En dessous, on peut voir la même construction de séquence, mais avec cette fois-ci le second IST choisi aussi grand que le troisième. Dans ce cas, on voit que la paire d'impulsion qui aurait dû être envoyée en troisième est maintenant la seconde paire. Cela change la méthode de construction de la séquence, et cette dernière ne pourra pas activer le réseau. Ainsi, nous avons choisi de concevoir un réseau d'exemple utilisé pour nos simulations avec 3 neurones, dont les ISTs sont respectivement 4.57×10^{-6}, 7.41×10^{-5} et 2.31×10^{-4}, vérifiant largement la contrainte précédente.

Les paramètres (C_m, g_L, τ_S) associés à chacun de ces neurones sont donnés dans le Tableau 5.1. Pour le dernier paramètre g_s, la même loi est utilisée pour tous les neurones. La conductance synaptique $g_s(t)$ varie dans le temps, influencée par les impulsions entrantes. Sa valeur est contrainte dans une plage allant de 0 à une valeur de saturation g_s^{max}, fixée à 100pS pour nos simulations.

TABLE 5.1 – Paramètres et métriques des neurones utilisés en simulation

Neurone	$C_m(F.cm^{-2})$	$g_L(S.cm^{-2})$	$\tau_S(s)$	$IST(s)$	$TW(s)$
Neurone 1	10^{-10}	10^{-5}	3×10^{-6}	4.57×10^{-6}	1.47×10^{-6}
Neurone 2	10^{-9}	10^{-5}	10^{-4}	7.41×10^{-5}	2.81×10^{-5}
Neurone 3	10^{-8}	10^{-4}	10^{-3}	2.31×10^{-4}	1.64×10^{-4}

5.2.2 Résultats

Dans nos travaux précédents du chapitre 4, nous avons montré que le TW de nos neurones n'est pas nul, ce qui implique qu'un léger décalage sur l'envoi d'une des impulsions pourrait quand même déclencher une impulsion dans le réseau. Ainsi, un neurone est capable de reconnaître un IST même en cas de faible différence dans le timing des impulsions. Cela peut permettre d'être très robuste pour éviter les MDs, mais peut entraîner l'apparition de FAs, dues à la reconnaissance de séquences proches de celle qui était ciblée. Dans cette partie, nous étendons cette étude à un réseau de neurones SLIF afin de fournir une évaluation complète et une quantification de cette sélectivité, ainsi que de déterminer les indicateurs de performance globaux du réseau.

Cette étude se concentre sur l'évaluation des probabilités de Fausses Alarmes (FA) et de MisDetections (MD).

MisDetections

Une Misdetection survient lorsque l'on rate la détection de la séquence que l'on ciblait. Habituellement, une MD vient du fait qu'entre la transmission de notre séquence et le moment où le réseau la reçoit, la séquence a été altérée, comme dans le chapitre 4. Nous allons nous placer dans un cas assez courant, celui dans lequel l'altération provient d'une désynchronisation fréquentielle entre l'émetteur et le récepteur dû aux défauts de fabrication des oscillateurs. Une des conséquences est l'accélération ou le ralentissement de la séquence par rapport au récepteur. Nous allons matérialiser cette désynchronisation par un jitter en fréquence, lui même représenté par un coefficient aléatoire par lequel nous multiplions tous les temps d'impulsion de notre séquence. En pratique, nous allons tirer un nombre d'une distribution normale, centrée sur 1, et avec un écart type de $\sigma = 0.1$, soit $X \sim \mathcal{N}(1, 0.1)$. S'il est inférieur à 1, la séquence sera considérée comme accélérée, sinon, comme ralentie.

Nous créons tout d'abord notre réseau à partir des trois neurones définis en Table 5.1, et la séquence associée. On va ensuite, à chaque itération, multiplier tous les temps d'impulsions par un jitter aléatoire. A chaque itération, nous vérifions si notre réseau est toujours capable de reconnaître cette séquence après l'avoir altérée. Si ça n'est pas le cas, on comptabilise une MD.

Nous avons pu observer que si le coefficient de jitter est proche de 1, les impulsions restent dans leur TW et le neurone de sortie atteindra quand même son seuil, alors que s'il est très éloigné, les ISTs reçus par nos neurones seront trop différents des favoris. Toutefois, il nous est possible d'élargir les TWs en modifiant la marge de seuil. Plus celle-ci sera élevée, plus le seuil sera bas, et donc facile à atteindre. Dans ce cas, nos neurones deviennent bien plus permissifs sur l'écart entre l'IST favori et l'IST reçu permettant d'atteindre la tension de seuil. Cela va donc améliorer la sensitivité, et diminuer la sélec-

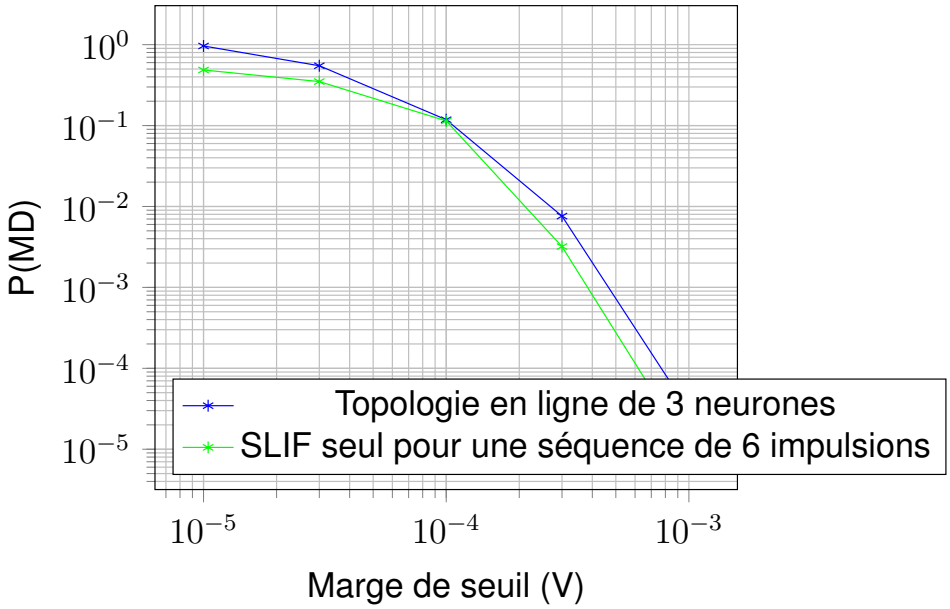


FIGURE 5.3 – Probabilité de Misdetection en fonction de la marge de seuil pour un jitter tiré selon $\mathcal{N}(1,\, 0.01)$.

tivité. C'est à dire qu'on va plus facilement ressortir une impulsion pour des séquences éloignées de celle ciblée, mais augmenter nos chances de ne pas rater la bonne séquence. Nous avons donc décidé de mesurer la probabilité de misdetection $P(MD)$ en fonction de cette marge de seuil. Par défaut, dans les travaux précédents nous utilisions sur les figures une marge entre l'amplitude maximale due à l'IST favori et la tension de seuil de $0.1mV$. Mais dans le cadre de ces simulations, nous allons tracer P(MD) en simulation pour 5 valeurs de marge de seuil : $10^{-5}V$, $3 \times 10^{-5}V$, $10^{-4}V$, $3 \times 10^{-4}V$, et $10^{-3}V$. Chaque point a été obtenu grâce aux résultats de 100000 itérations.

Nous pouvons observer en fig 5.3 que logiquement, une marge de seuil faible va réduire la sensibilité du neurone. Plus cette marge est faible, plus l'IST reçu va devoir être proche du favori pour entraîner une impulsion en sortie. De ce fait, on restreint le jitter à être plus proche de 1 pour avoir une reconnaissance fructueuse, et donc on va augmenter le nombre de misdetections. A l'inverse, avec une très grande marge de seuil, on autorise le neurone à décharger même pour un jitter éloigné de 1. Cela entraîne une drastique diminution du nombre de misdetections, au coût très certain de décharger pour des séquences qui ne seraient pas celle qui correspond aux ISTs de nos neurones, qui sont donc de fausses alarmes.

Fausses Alarmes

Séquences totalement aléatoires

Une fausse alarme correspond à une occurrence de reconnaissance de séquence, alors que la bonne séquence n'a pas été envoyée. Pour évaluer la probabilité de fausses alarmes, nous considérons des timings d'impulsions aléatoires dans une plage correspondant à la durée totale de la séquence, représentant ainsi la transmission d'une séquence totalement aléatoire utilisée afin de mesurer la susceptibilité du réseau à déclencher de fausses alarmes. Nous avons introduit différentes valeurs de marge de seuil pour quantifier les performances du système. La marge de seuil représente l'amplitude dans laquelle le neurone peut initier une impulsion vers les neurones suivants, et impacte le TW. Une marge plus grande

correspond à un seuil plus bas et à un TW plus élevé. L'analyse complète de ces métriques quantifie les performances du réseau à différents niveaux de sélectivité.

Si une impulsion doit se produire IST_i secondes après l'impulsion précédente, elle doit tomber dans une plage spécifique définie par la largeur temporelle TW_i dans la durée totale T_{tot}. Par conséquent, la probabilité de tirer l'impulsion dans cette fenêtre est calculée comme TW_i/T_{tot}. On en déduit la probabilité de FA, correspondant à la probabilité que chaque impulsion soit tirée dans le TW correct autour de l'IST ciblé.

$$P(FA) = \frac{\prod_{i=1}^{n} TW_i \times n!}{(T_{tot})^n} \tag{5.1}$$

Pour notre réseau à 3 neurones, nous avons déterminé la probabilité de FA pour $n_s = 2^n - 1 = 7$ impulsions avec $n = 3$. On enlève 1 parce que nous avons fixé la première impulsion de la séquence à $t = 0s$. Nous avons calculé la probabilité de FA (eq. 5.1) et observé que pour une $n_s = 7$ impulsions, la probabilité est $P(FA) = 3.25 \times 10^{-12}$, ce qui rend l'occurrence de FA trop rare pour être observée en simulation. En effet, sachant qu'un million d'itérations prennent environ 1 heure de simulation, il faudrait environ 100 ans pour en observer une.

D'une façon plus générale, la probabilité de FA, en considérant un tirage aléatoire de $n_s = 2^n - 1$ impulsions, pour un nombre variable de neurones et de marges de seuil, est illustrée dans la fig 5.4 pour les neurones décrits dans le tableau 5.1. Cette plage de valeurs de marge de seuil permet d'explorer différents niveaux de sensibilité et de sélectivité dans la réponse du réseau aux séquences d'impulsions. Le graphique montre une diminution rapide de la probabilité vers zéro lorsque nous réduisons la marge de seuil et donc, lorsque le TW est réduit. Cela est dû à la nécessité de tirer chaque IST dans la plage de TWs des neurones, centrée sur la valeur d'IST originale pour obtenir une fausse alerte. Le premier neurone a le TW le plus petit, donc si le timing de l'impulsion est en dehors de ce fin TW, le premier neurone n'émettra pas d'impulsion. Comme ce TW est relativement petit par rapport à la plage temporelle dans laquelle s'inscrit la séquence globale, même un léger décalage du timing d'une impulsion en dehors de ce TW rend l'occurrence de FA presque impossible, même si toutes les autres impulsions sont parfaitement choisies. Ensuite, si un deuxième neurone est ajouté, 3 impulsions doivent être bien placées dans la séquence au lieu d'une seule. La deuxième doit se trouver autour de IST_1 après le début, la troisième impulsion doit être placée autour de IST_2, et la dernière doit être IST_1 après l'avant-dernière. C'est pourquoi les fausses alertes sont plus rares lorsque nous utilisons plus de neurones.

Nous pouvons observer que les simulations et la théorie donnent presque les mêmes probabilités, mais la théorie est un peu moins précise que ce que nous avons simulé lorsque nous utilisons seulement un ou deux neurones. Cela est dû au fait qu'en pratique, pour une paire d'impulsions, lorsque l'une est tirée à l'extrême gauche du TW et l'autre à l'extrême droite de son TW, l'IST sera trop grand pour que le neurone émette une impulsion, même si elles sont toutes deux dans leur TW prédéterminé. Ce phénomène réduit le risque d'obtenir des FAs lorsque nous ajoutons des neurones et des impulsions.

On peut noter que, lorsqu'il y a 3 neurones et des marges de seuil élevées, le troisième TW dépasse la plage dans laquelle nous tirons nos timings d'impulsions. Par conséquent, nos simulations aboutissent à moins de FA que ce que nous aurions dû avoir d'après (5.1). Nous en déduisons donc que notre expression théorique agit comme une borne supérieure de la probabilité d'obtenir une FA en simulation.

De plus, certains points de simulation manquent. En effet, pour $n_s = 7$ impulsions et la

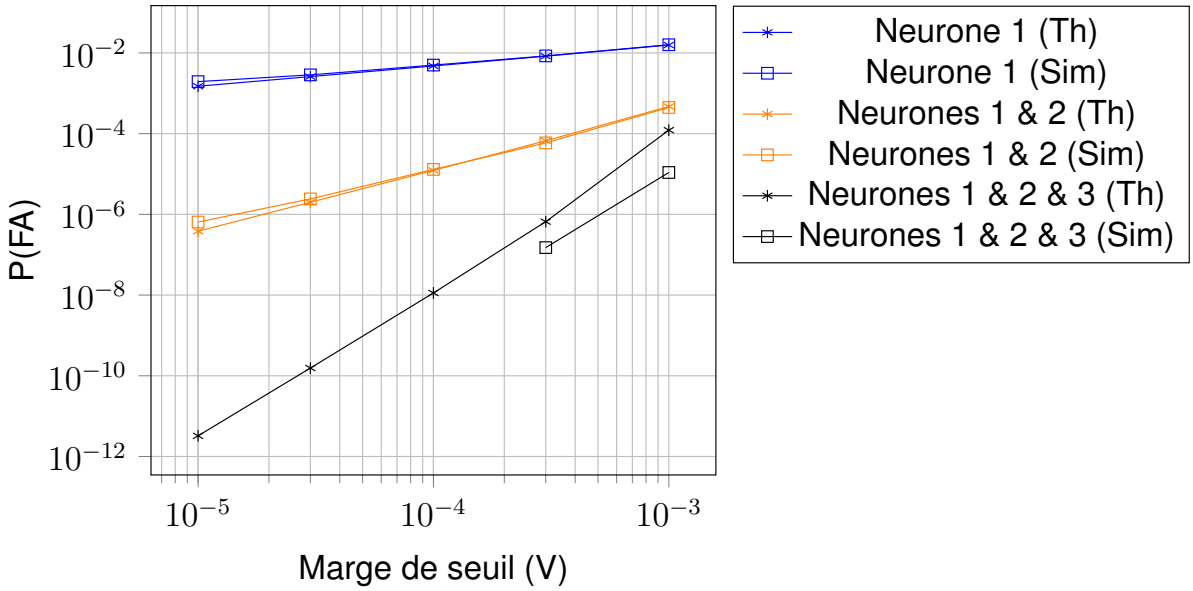


FIGURE 5.4 – Probabilité théorique et simulée de FA en fonction du nombre de neurones

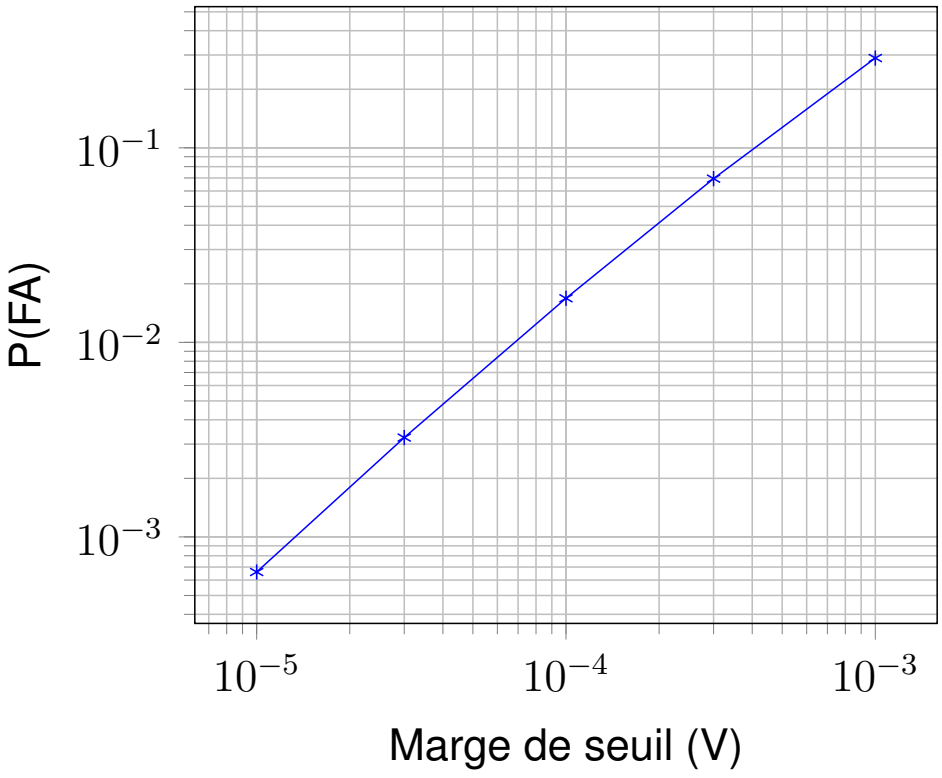


FIGURE 5.5 – Probabilité de FA obtenue en simulation en fonction de la marge de seuil

plus petite marge de seuil, une occurrence est extrêmement rare, avec $P(\text{FA}) = 3.25 \cdot 10^{-12}$. Nous avons testé et n'avons pas obtenu une seule FA en 10^8 itérations pour la marge de seuil de 10^{-4} avec 3 neurones, ce qui est cohérent avec l'approche théorique.

Cependant, les séquences non désirées pourraient ne pas être complètement aléatoires, mais plutôt conçues avec une logique similaire. En particulier, les ISTs pourraient être choisis dans différents ordres de grandeur. Nous définissons donc 3 ISTs différents dans chacune des 3 plages différentes (entre $2\mu s$ et $10\mu s$ pour la première plage, entre $20\mu s$

et $100\mu s$ pour la seconde, et entre $200\mu s$ et $1ms$ pour la dernière). Ensuite, on tire trois nombres aléatoires selon une distribution uniforme pour chacune des 3 plages. On va ensuite chercher dans notre base de paramètres l'IST enregistré le plus proche de chaque valeur tirée afin d'avoir des paramètres réalistes pour le simuler. Les codes sont construits en choisissant un IST dans chaque plage. Ainsi, $3^3 = 27$ codes différents peuvent être obtenus. Nous considérons l'un d'eux comme la séquence ciblée.

Ensuite, nous créons de nouvelles séquences en sélectionnant un autre ensemble d'ISTs. Pour chaque plage, soit on conserve celui de référence (avec une probabilité de 1/2), soit on en choisit aléatoirement un autre dans la plage correspondante.

Les résultats présentés dans la fig 5.5 montrent la probabilité de FA pour cinq valeurs distinctes de marge de seuil : $10^{-5}V$, $3 \times 10^{-5}V$, $10^{-4}V$, $3 \times 10^{-4}V$, et $10^{-3}V$, vérifiant la contrainte mentionnée précédemment. La figure illustre une augmentation de la probabilité à mesure que la marge de seuil augmente, lié à une diminution de la valeur du seuil. Cette observation est conforme aux attentes, car une marge de seuil plus grande permet au neurone de décharger pour des ISTs qui s'éloignent davantage de l'IST favori. Par conséquent, cette plage plus large permet de déclencher des FA pour un spectre plus large d'ISTs, entraînant une diminution des performances globales de notre réseau.

Ainsi, notre préférence s'oriente vers l'utilisation d'une valeur de marge de seuil plus faible, car elle améliore la sélectivité du réseau. Cependant, cela s'accompagne du compromis d'une augmentation de la probabilité de MD en présence de décalages de timing d'impulsions.

En résumé, augmenter la marge de seuil augmente notre sensibilité à la séquence cible, mais nous fait détecter un plus grand nombre de séquences différentes, diminuant grandement les misdetections mais augmentant le nombre de fausses alarmes.

Comme démontré précédemment, l'ajustement de cette valeur de marge de seuil permet de faire un choix stratégique entre minimiser les MDs et les FAs. Alternativement, un compromis peut être trouvé en choisissant une valeur intermédiaire, conduisant à des résultats modérés pour les deux métriques.

FA avec des séquences contraintes

Afin d'avoir un jeu de paramètres permettant un plus grand choix de codes différents, nous avons cherché à avoir plus de neurones différents par décade, dont les TWs ne se chevauchent pas. Nous avons ainsi réussi à en obtenir 6 par décade, dont les paramètres peuvent être retrouvés dans le tableau 5.2. Pour obtenir les paramètres une décade au dessus, il faut multiplier par 10 les valeurs de C_m et diviser τ_s par 10. Cela conduit à un IST et un TW 10 fois plus grands. Faire l'inverse permet de passer à la décade en dessous. Il nous est donc possible, en gardant le même modèle de réseau, de reconnaître 6^n séquences orthogonales, avec n le nombre de neurones et de plages dans le réseau. Pour cet exemple avec 3 neurones, cela nous permet d'utiliser 216 séquences différentes. Nous avons créé ces 216 récepteurs et pour chacun d'eux, nous avons présenté toutes les séquences sauf la leur, pour vérifier que les neurones ne répondront pas aux autres ISTs favoris de notre liste. Nous n'avons en effet obtenu aucune FA dans les simulations, ce qui implique que ces 216 séquences sont bien orthogonales par rapport à ce récepteur.

5.2.3 Conclusion

Dans cette partie, nous avons proposé une architecture en ligne de SNN basée sur des neurones SLIF pour la reconnaissance de motifs. Nous avons montré que ce type de SNN est capable de discriminer un motif basé sur des ISTs des autres et de n'émettre

TABLE 5.2 – Paramètres et métriques des 6 neurones utilisables pour la décade 10^{-6}

Neurone	$C_m(F \cdot cm^{-2})$	$g_L(S \cdot cm^{-2})$	$\tau_S(s)$	$IST(s)$	$V_T(V)$	$TW(s)$
Neurone 1	1.2507×10^{-16}	3.9550×10^{-11}	1.5849×10^{-6}	1.89×10^{-6}	-2.8796×10^{-2}	6.05×10^{-7}
Neurone 2	1.9822×10^{-16}	3.9550×10^{-11}	2.5119×10^{-6}	2.995×10^{-6}	-2.8796×10^{-2}	9.58×10^{-7}
Neurone 3	1.2507×10^{-16}	3.1416×10^{-12}	1.5849×10^{-6}	4.509×10^{-6}	-1.0698×10^{-2}	1.94×10^{-6}
Neurone 4	3.9550×10^{-16}	3.9550×10^{-11}	5.0119×10^{-6}	5.976×10^{-6}	-2.8796×10^{-2}	1.91×10^{-6}
Neurone 5	4.9791×10^{-16}	3.9550×10^{-11}	6.3096×10^{-6}	7.523×10^{-6}	-2.8796×10^{-2}	2.41×10^{-6}
Neurone 6	7.4955×10^{-16}	4.1416×10^{-11}	6.2623×10^{-6}	9.147×10^{-6}	-3.3082×10^{-2}	2.93×10^{-6}

une impulsion que pour la signature ciblée. Nous avons utilisé une topologie simple mais efficace qui consiste en une architecture totalement feed-forward. Nous avons également conjointement créé la séquence correspondante et montré qu'elle est effectivement celle qui fait décharger le réseau.

Nous avons étudié les performances de ce réseau en termes de FA et de MD, et discuté de l'impact de la marge de seuil des neurones sur ces métriques, ainsi que du compromis à réaliser. Actuellement, n neurones dans le réseau signifient que nous pouvons créer 6^n séquences orthogonales pour l'IoT, mais cela nécessite 2^n impulsions. Ce résultat peut être étendu. En effet, nous pouvons faire de même avec d'autres plages de temps et un nombre différent de neurones dans d'autres simulations. Cependant, plus d'impulsions signifie plus de consommation, c'est pourquoi il semble important d'étudier d'autres topologies qui nous permettraient d'utiliser moins d'impulsions pour le même nombre de neurones. Il pourrait également être intéressant de voir les métriques et les contraintes de l'utilisation d'un plus grand nombre de neurones afin de pouvoir adresser bien plus de 216 noeuds IoT comme cela serait le cas dans de réelles applications.

5.3 Étude de possibilités topologiques

5.3.1 Topologie en losange

Modèle du réseau

Le réseau de neurones SLIF en ligne offre une topologie simple pour la transmission des impulsions, mais nécessite un nombre élevé d'impulsions pour déclencher une réponse. En ajoutant un neurone, le nombre d'impulsions doit être multiplié par deux, augmentant ainsi la longueur temporelle des séquences, ce qui pourrait poser problème dans des applications réelles. Cela augmente également la consommation d'énergie du réseau, qui dépend linéairement du nombre d'impulsions traitées. L'objectif étant de minimiser la consommation d'une WuR, il est essentiel de réduire le nombre d'impulsions entrante par neurone dans le réseau.

Pour améliorer l'efficacité, nous avons augmenté le nombre de neurones par couche du réseau. C'est à dire que plutôt que d'avoir 3 étages de neurone unique, nous plaçons plusieurs neurones en parallèle, tous connectés à la couche suivante. Nous avons tout d'abord exploré la topologie la plus simple suivant cette idée, une topologie en losange composée d'un neurone d'entrée envoyant la séquence à deux neurones dans la première couche, eux-même connectés à un neurone de sortie, représenté en fig 5.6. Le but de cette configuration est d'évaluer s'il est possible de réduire le nombre d'impulsions nécessaires pour déclencher une réponse, ce qui permettrait de passer de 8 impulsions entrantes dans

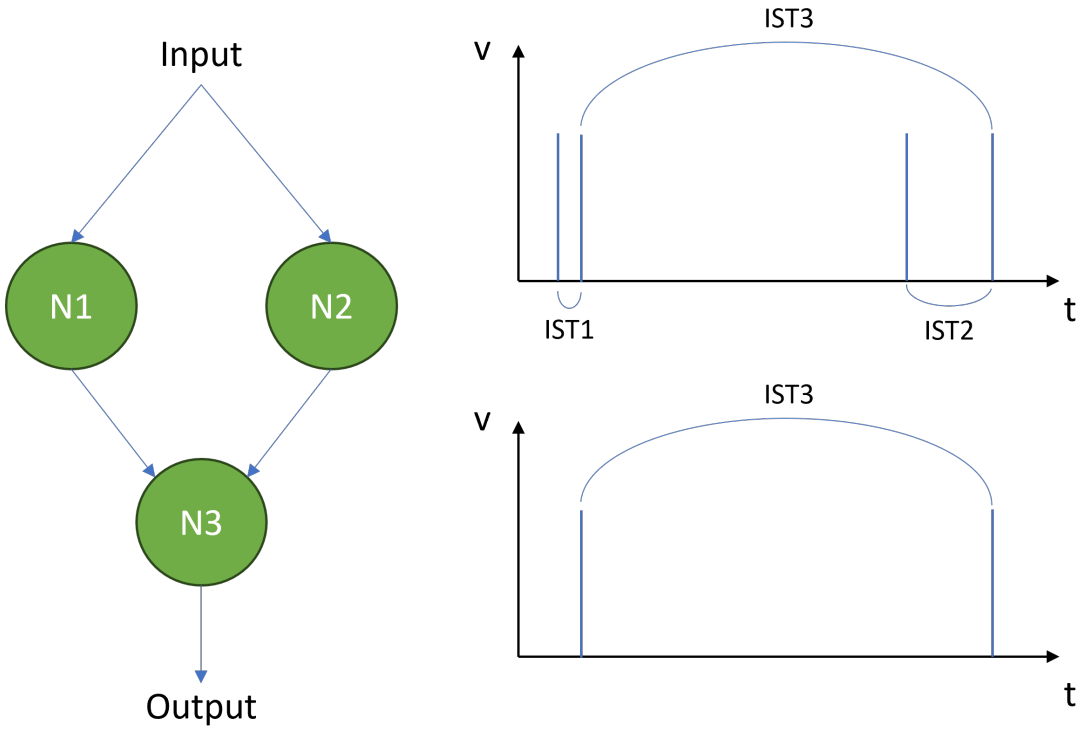


FIGURE 5.6 – Co-design d'un réseau en losange et de sa séquence associée

la topologie en ligne à seulement 4 dans la topologie en losange, tout en ayant toujours 3 neurones. Cette optimisation suggère que des arrangements plus complexes de neurones par couche pourraient offrir une utilisation plus efficiente des ressources neuronales. Il nous faudra donc déterminer si ces topologies sont de même niveau ou meilleures que la topologie en ligne, ou seulement plus efficiente.

On peut observer en fig 5.6 qu'on a effectivement un réseau de 3 neurones activé par des séquences codées à partir de 3 ISTs différents mais qui ne nécessite plus que 4 impulsions. Pour arriver à cela de la même manière que dans la section précédente, la séquence correspondant à une topologie doit être conçue conjointement. On peut retrouver en fig 5.6 les différents timings d'impulsion dont on a besoin pour déclencher le neurone de sortie. En partant du neurone de sortie et en remontant on peut déterminer cette séquence. On suppose que la première impulsion est envoyée à $t = 0$. On nomme les trois neurones N_i et leur IST correspondant IST_i. On peut observer que la seconde impulsion est envoyée à $t = IST_1$. On peut ensuite déterminer que l'écart entre la seconde et la quatrième doit être de IST_3. Cette dernière sera donc envoyée à $t = IST_1 + IST_3$. Enfin, la troisième doit être envoyée IST_2 avant la quatrième pour déclencher N_2, elle sera donc envoyée à $t = IST_1 + IST_3 - IST_2$

Il subsiste une contrainte sur les choix des ISTs pour éviter d'avoir un recouvrement des impulsions dans la séquence. Dans cet exemple, il faudra que IST_3 soit strictement plus grand que IST_2. Toutefois, il faut que IST_3 soit significativement plus grand car plus l'écart sera grand, plus le neurone N_2 aura du temps pour permettre à sa tension membranaire de redescendre à sa tension de repos. Si les deux valeurs d'IST étaient trop proches, la troisième impulsion arriverait juste après la seconde, ce qui créerait facilement une fausse alarme sur N_2. La tension de ce neurone n'atteindrait donc pas son seuil, et le réseau ne pourrait pas détecter la séquence. Pour nos simulations nous avons choisi

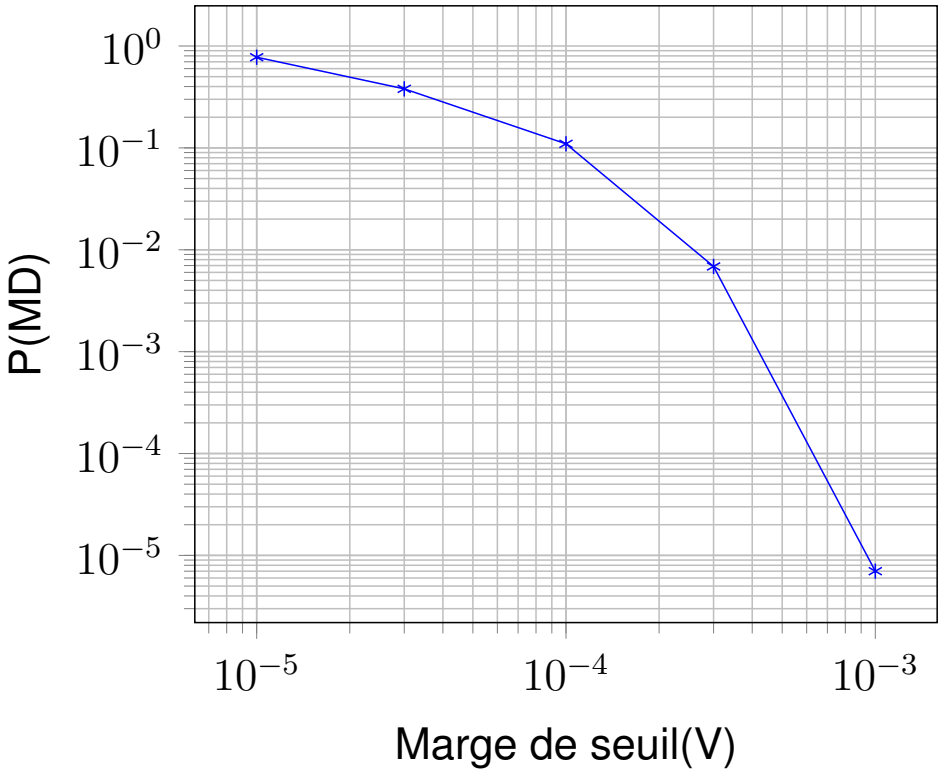


FIGURE 5.7 – Probabilité de MD en fonction de la marge de seuil pour un jitter tiré selon $\mathcal{N}(1, 0.01)$.

de conserver les mêmes neurones que pour celles de la topologie en ligne, qui respectent toujours la contrainte d'écart minimal.

Résultats

Afin de pouvoir réaliser des comparaisons avec la topologie en ligne, nous avons utilisé les mêmes N_1, N_2, N_3 que dans la section 5.2, dont les paramètres ont été donnés en table 5.1. Nous avons ensuite réalisé les mêmes études que dans la partie précédente.

L'étude des MDs a été réalisée en ajoutant un jitter fréquentiel déterminé de manière aléatoire à la séquence ciblée. Nous avons ensuite compté le nombre d'itérations pour lesquelles le système n'a pas reconnu la séquence. Cette expérience a été réalisée un million de fois pour chaque marge de seuil définies précédemment, afin d'obtenir quelques occurrences de MD pour la plus grande marge de seuil. L'évolution de la probabilité de MD en fonction de la marge de seuil, pour un réseau en losange est tracée en fig 5.7. Nos observations montrent une fois de plus une forte décroissance quand on s'approche des grandes marges. De la même manière qu'avec la topologie précédente, il est logique que lorsque notre réseau devient moins sélectif (car on élargit nos TWs), on reconnaisse plus facilement notre séquence comportant une désynchronisation fréquentielle. En effet, on envoie la séquence ciblée, accélérée ou ralentie. De ce fait, les timings d'impulsions réellement reçus seront légèrement avant ou après ceux définis initialement. Cela nous donne une séquence dont les nouveaux timings sont proches de ceux ciblés, et potentiellement, toujours dans le TW. Plus la marge de seuil est grande, plus le TW sera large, plus on aura de chance que nos timings soient dans cette fenêtre.

Si on compare ces résultats à ceux obtenus en fig 5.3, également pour 3 neurones,

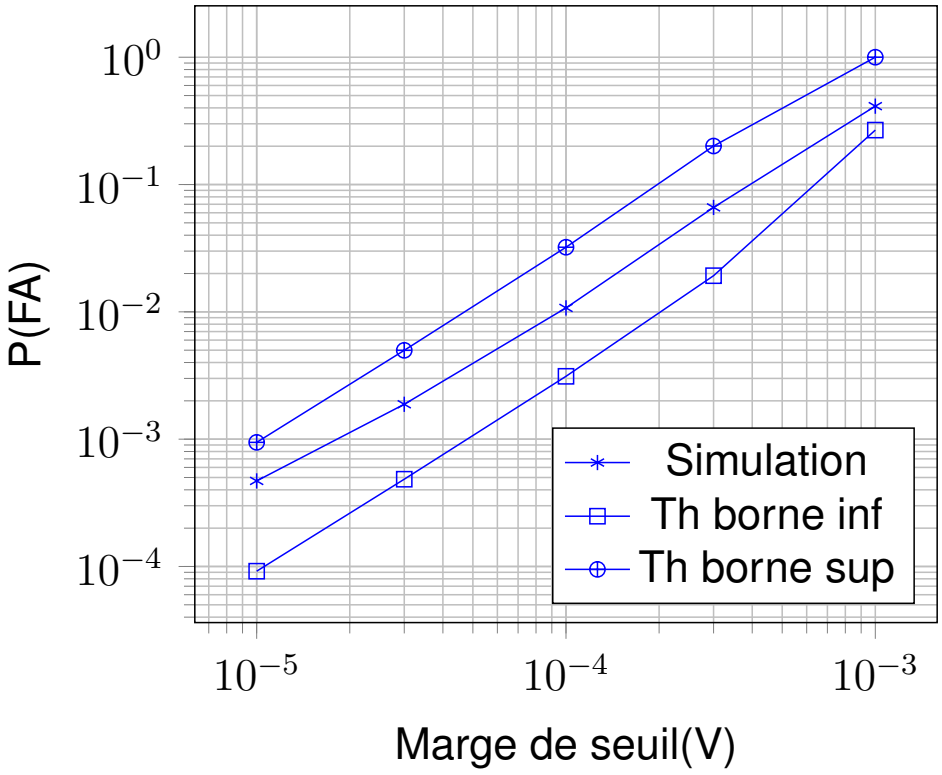


FIGURE 5.8 – Probabilité de FA en fonction de la marge de seuil

on remarque que les valeurs sont du même ordre de grandeur, mais sont 10 à 30% plus faibles pour le réseau en losange, à l'exception de probabilité pour une marge de $10^{-3}V$ qui est 4 fois moins élevée avec cette seconde topologie. On peut expliquer cette diminution de fréquence d'apparition de MD par le fait que nous ayons plus que 4 impulsions par séquence, ce qui limite plus l'impact du jitter que si l'on avait 8 impulsions par séquence. De plus, dans l'exemple de la ligne, chaque impulsion, à l'exception de la première, avait une contrainte à respecter pour être à la fois à une distance de IST_1 d'une impulsion, à IST_2 d'une autre, et à IST_3 d'une troisième. Par exemple, la quatrième impulsion de la séquence dans 5.1 doit respecter une distance de $IST_1 \pm \frac{TW_1}{2}$ avec le troisième, une distance de $IST_2 \pm \frac{TW_2}{2}$ avec la seconde impulsion, et une distance de $IST_3 \pm \frac{TW_3}{2}$ avec la huitième. Dans le cas du losange, chaque impulsion n'a que deux contraintes à respecter pour faire décharger le réseau. Ici, par exemple, la deuxième impulsion doit être à une distance de $IST_1 \pm \frac{TW_1}{2}$ de la première et à $IST_3 \pm \frac{TW_3}{2}$ de la quatrième. Il est donc plus probable que l'on ait une MD dans le cas de la topologie en ligne, pour un nombre de neurones équivalent. Enfin, si l'on souhaite minimiser la probabilité de MD, on voit sur ce graphique que la meilleure valeur est toujours obtenue pour une marge de $10^{-3}V$, avec cette fois ci un écart entre les performances pour 3×10^{-4} et $10^{-3}V$ plus important que dans la topologie précédente. Toutefois, une fois de plus, il nous faut étudier le compromis en P(FA) et P(MD)

La probabilité de FA a été déterminée en envoyant des codes totalement aléatoires à notre réseau en losange. Les résultats sont reportés sur la courbe "Simulation" de la fig. 5.8. On peut s'apercevoir que la probabilité de FA est largement supérieure à celle que nous avions obtenues pour la topologie en ligne. Plusieurs phénomènes expliquent pourquoi cette topologie, avec les neurones employés, est bien moins robuste aux FAs. La principale raison vient du choix des neurones. En effet, le neurone N_3 avait été choisi pour

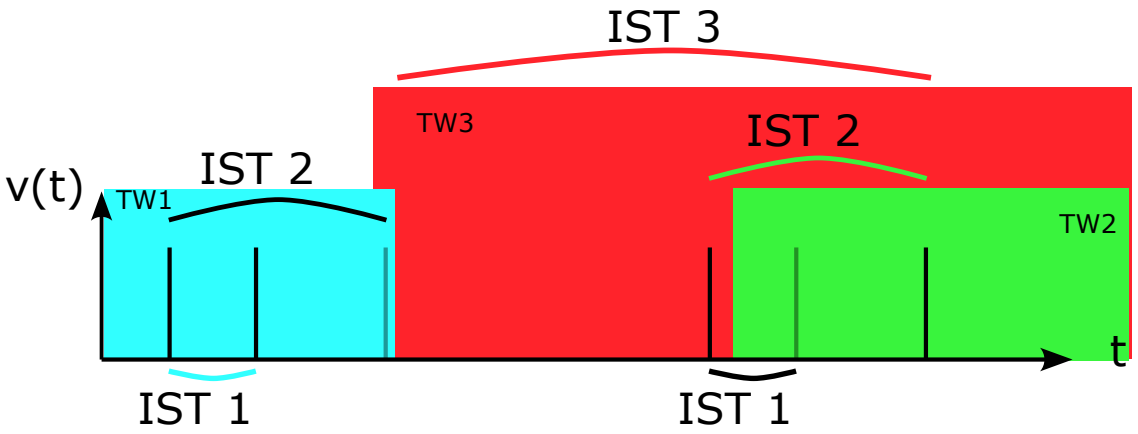


FIGURE 5.9 – Illustration des 4 séquences activant le réseau en losange

son large TW. Le problème qui en découle est que ce TW couvre totalement la fenêtre sur laquelle on tire nos timings d'impulsion. De ce fait, TW_3 n'est plus une contrainte car si N_3 reçoit deux impulsions, il va décharger quel que soit le temps qui les sépare. Ensuite, en observant plus en détail les séquences aléatoires faisant déclencher notre réseau, nous nous sommes aperçus qu'il y avait plus d'une séquence qui était reconnue par notre réseau. En effet, si N_3 reçoit deux impulsions séparées de IST_3, il va atteindre son seuil. Mais, cela peut arriver dans 4 cas puisque les neurones sont insensibles à l'ordre d'arrivée des impulsions, seulement à leur écart, et insensibles à la provenance des impulsions :

- Tout d'abord celui que nous avions prévu, une première paire séparée de IST_1, puis une autre séparée de IST_2, elles mêmes séparées de IST_3.
- Ensuite, si on inverse les deux paires, N_3 va bien recevoir une paire avec un IST de IST_3, et déclencher.
- Enfin, si les deux paires ont un IST de IST_1 (premier cas) ou IST_2 (deuxième cas) toutes les deux, le réseau va également reconnaître la séquence.

Nous avons illustré ce phénomène en fig. 5.9. La plage bleue représente la plage $TW1$, la verte représente $TW2$, et la rouge $TW3$. On peut voir la séquence correspondant au réseau en noir. En remplaçant la deuxième impulsion noire par la première grise, on voit que la première paire d'impulsions est toujours dans $TW1$. De même, en remplaçant la quatrième noire par la seconde grise, la seconde paire est bien dans $TW2$. Enfin, dans les 4 cas, l'écart temporel entre les deux paires sera compris dans $TW3$. Ces 4 configurations entraînent le déclenchement du neurone de sortie.

Nous nous sommes aperçus que le tout dernier cas évoqué, pour une séquence composée de deux paires d'IST_2, est même le plus probable en simulation. Cela est dû au fait qu'IST_2 étant à peu près 10 fois plus grand qu'IST_1, la probabilité d'être dans son TW est également de l'ordre de 10 fois plus grande. De ce fait, plus la marge de seuil va être réduite, plus la différence entre nos valeurs théoriques et nos valeurs de simulations seront éloignées. Pour vérifier cela, nous avons évalué théoriquement cette probabilité de FA à l'aide de l'équation (5.1) en fig. 5.8. On retrouve bien les deux éléments pressentis. Tout d'abord, la probabilité de FA en simulation est supérieure à celle théorique. Ensuite, on voit bien en fig. 5.8 que plus la marge de seuil est petite, plus la théorie s'éloigne de la simulation. Si la marge de seuil est grande, les TWs seront assez larges pour couvrir les autres ISTs ce qui rend possible les quatre cas mentionnés ci dessus. Sinon, les cas commencent à se dissocier et les TWs ne se superposent plus, ce qui accentue la probabilité de FA pour des cas non pris en compte dans la formule. C'est pour cela que nous l'utilisons

comme borne inférieure de notre probabilité d'erreur Nous avons ensuite calculé la somme des probabilités d'apparition de chacune des quatre séquences afin d'avoir une borne supérieure de P(MD), toujours à l'aide de l'équation (5.1). Cette probabilité est supérieure à la vraie probabilité de tirer aléatoirement une séquence proche de celle faisant déclencher le système, car pour les valeurs hautes de marge de seuil, les TWs donnant une des quatre séquences se chevauchent. On peut toutefois remarquer que nos valeurs en simulation sont bien comprises entre les deux courbes de probabilité théorique.

En comparant ces résultats avec ceux de la topologie précédente, on peut voir que l'on a ici beaucoup plus de FAs que dans le cas "Neurones 1 & 2" de la fig 5.4. On retrouve ici le fait que pour tirer des timings proches de ceux des 4 séquences, il ne faut respecter que deux critères par rapport aux ISTs comme expliqué pour les MDs. De ce fait, il est beaucoup plus probable de tomber sur une séquence amenant le réseau à déclencher une impulsion en sortie. De plus, TW_2 étant une décade au dessus de TW_1, la probabilité d'avoir N_2 qui envoie deux paires à N_3 est de l'ordre de 10 fois à celle d'avoir une des trois autres.

En conclusion, dans cette partie, nous avons pu voir les performances offertes par cette seconde topologie, et les comparer avec la topologie en ligne. Nous avons pu observer de meilleures performances en termes de résistance aux MDs, notamment parce qu'on a moins d'impulsions qu'avec le réseau en ligne. Par contre, nous avons pu voir que ce second réseau était beaucoup moins efficace quand il s'agit d'éviter d'avoir des FAs. Même si l'on compare avec le cas à 2 neurones, la probabilité de FA reste de l'ordre de 10^2 à 10^3 supérieure pour le cas du losange. Celui vient des deux raisons évoquées au dessus : le fait que 4 séquences déclenchent le réseau, et que le choix d'un N_3 ayant un TW particulièrement grand par rapport à son IST a un impact encore plus négatif sur les performances du losange par rapport à la ligne. Nous avons également vu que lorsque plusieurs neurones d'une couche sortent sur le même neurone de la couche d'après, cela augmente le nombre de séquences activant le réseau. De ce fait, si l'on veut augmenter le nombre d'adresses disponibles pour notre réseau en rajoutant des ISTs différents sur une même couche, il faudra choisir la topologie de manière à ne pas permettre qu'il y ait plusieurs séquences d'activation.

Il nous aurait été possible de palier au fait que 4 séquences différentes font déclencher le réseau, en rajoutant le phénomène de période réfractaire à nos neurones. Pour rappel, ce phénomène empêche la tension membranaire d'un neurone de quitter sa valeur de repos, pendant une période imposée, après avoir atteint sa tension de seuil. Dans notre cas, si chaque neurone a une période réfractaire de IST_3, il n'est alors plus possible que N_3 atteigne son seuil grâce à deux paires provenant du même neurone de la couche précédente. Il nous resterait quand même deux séquences différentes reconnues par le réseau, mais aucun mécanisme capable de nous faire descendre à une seule. Pour palier au fait que le TW de N_3 est trop large pour nous permettre d'être sélectifs, nous avons choisi de refaire ces simulations avec des neurones provenant cette fois de la table 5.2 afin de montrer l'impact du choix des paramètres neuronaux sur les performances.

Il est à noter que contrairement aux neurones précédents, les neurones de la table 5.2 ont tous un rapport IST/TW du même ordre de grandeur. Il n'y a donc pas de neurone qui pourrait drastiquement augmenter une des deux probabilités d'erreur. Dans cette nouvelle série de simulations, nous avons choisi de tirer aléatoirement notre réseau. N_1 et N_2 sont choisis aléatoirement parmi les 6 de la table, tout en étant forcément différents, et N_3 est choisi parmi les 6, potentiellement les mêmes paramètres que N_1 ou N_2, mais décalé d'une décade. Avec cette méthode, la contrainte sur les ISTs est forcément respectée.

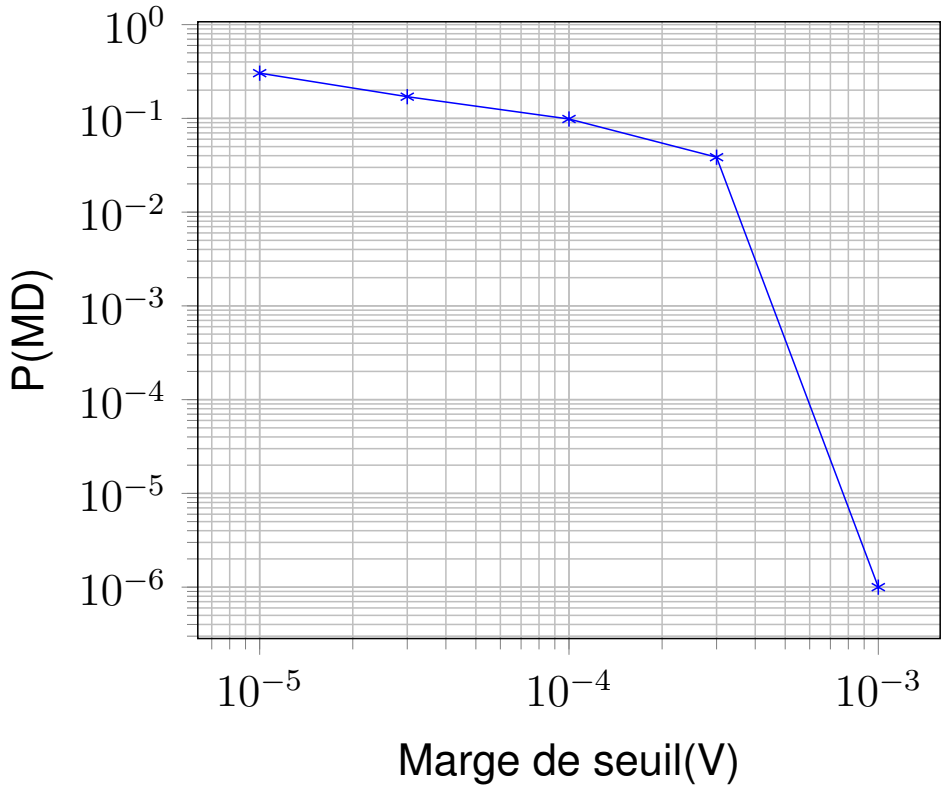


FIGURE 5.10 – Probabilité de MD en fonction de la marge de seuil pour un jitter tiré selon $\mathcal{N}(1,\, 0.01)$.

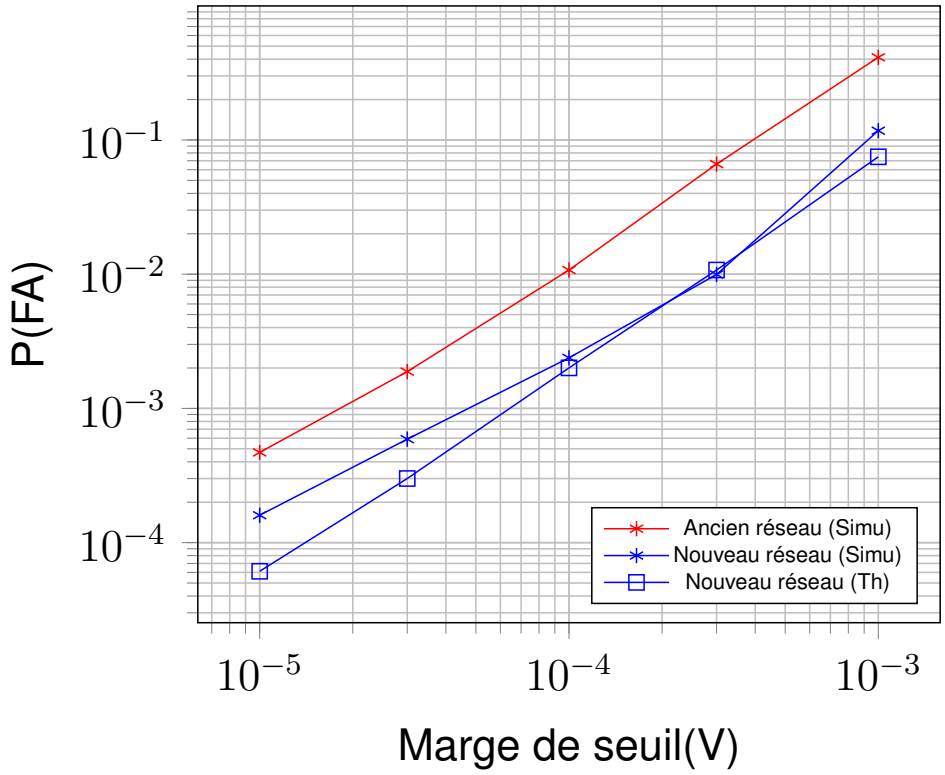


FIGURE 5.11 – Probabilité de FA en fonction du seuil de marge

Les résultats de nos simulations permettant de déterminer la probabilité de MD pour cette deuxième itération de topologie en losange sont affichées sur la fig 5.10. On peut noter que cette courbe montre des performances moins bonnes que sur la courbe 5.7. Cela s'explique justement par le fait que nous avons choisi des neurones ayant un TW plus faible, comparé à leur IST. Dans ce cas, chaque neurone sera plus sélectif sur l'IST reçu, et donc sera moins susceptible de déclencher lors de tirages de jitter plus loin de 1. Toutefois, ces résultats restent dans le même ordre de grandeur que précédemment.

La probabilité de MD a été mesurée en envoyant des séquences à un grand nombre de réseaux différents. Pour chaque réseau on envoie la séquence altérée par un jitter aléatoire. Pour cela, on tire une valeur d'après une distribution $\mathcal{N}(1,\, 0.01)$ et on multiplie tous les instants d'impulsions de la séquence par cette valeur. Ensuite, on envoie ces nouvelles séquences et on compte le nombre de fois où celle-ci n'est pas reconnue. La probabilité de FA a été déterminée en suivant la même expérience que précédemment, on tire des timings d'impulsion selon une loi uniforme sur une durée égale à celle de la séquence ciblée. On peut voir sur la fig 5.11 trois courbes correspondant aux performances en simulation du réseau avec les nouveaux paramètres, de celui avec les anciens paramètres, et la courbe théorique pour les nouveaux paramètres. On peut tout d'abord confirmer que le changement de neurones, pour des nouveaux ayant des TWs plus fins, a permis de diminuer la probabilité de FA. En effet, pour la courbe théorique, avoir TW_1 et TW_2 dans la même décade, TW_3 seulement une décade au dessus des deux autres, et avoir des TWs plus faible en comparaison avec leur IST favori, a permis de réduire la probabilité d'occurrence d'une FA. La probabilité théorique a été calculée avec l'eq. (5.1) et ne prend pas en compte le fait que 4 séquences permettent de déclencher le neurone de sortie. Toutefois, bien qu'elle soit réduite par le fait de ne compter qu'une séquence, elle est également revue à la baisse car, comme on a pu le voir dans le chapitre 4, si une paire d'impulsions voit chacun de ses timings à l'extrémité opposée de son TW respectif, alors l'équation aurait compté une FA, alors qu'en simulation il n'y en aurait pas eu.

On peut donc en conclure que ce nouveau jeu de neurones nous aura permis de diminuer la probabilité de FA, au prix d'une augmentation du nombre de MD, grâce à des neurones plus sélectifs. Le choix des paramètres doit donc se faire en fonction de la tâche à accomplir, de sa criticité, et de la forme que peuvent prendre les variations de signaux. S'il est vraiment important de ne rater aucune communication, mais qu'un réveil par erreur n'est pas critique, on va préférer un circuit plus petit, avec potentiellement moins d'impulsions dans la séquence. Si au contraire on veut s'assurer que chaque réveil ne se fait que lorsqu'il est réellement demandé, et qu'on peut sans problème renvoyer des requêtes si les précédentes ont été manquées, on va vouloir favoriser un plus grand réseau, avec une séquence forcément unique. On rappelle qu'un sigma de 10^{-2} est supérieur à la réalité et que dans des cas réalistes, il parait très invraisemblable que notre récepteur reçoive autant de séquences sur des temps aussi courts, qui respecteraient la temporalité que l'on a imposé pendant nos simulations.

5.3.2 Topologie multi-couche

Les deux topologies montrées précédemment comportaient toutes les deux peu de neurones, et étaient activées par des séquences comportant très peu d'impulsions. Grâce à cela, on a réussi à montrer que l'on avait très peu de MDs, et des séquences assez courtes, donc une consommation très faible. Toutefois, même si nos conditions de simulations sont plus dures que les cas réels, la probabilité de FA reste très supérieure à celle de MD.

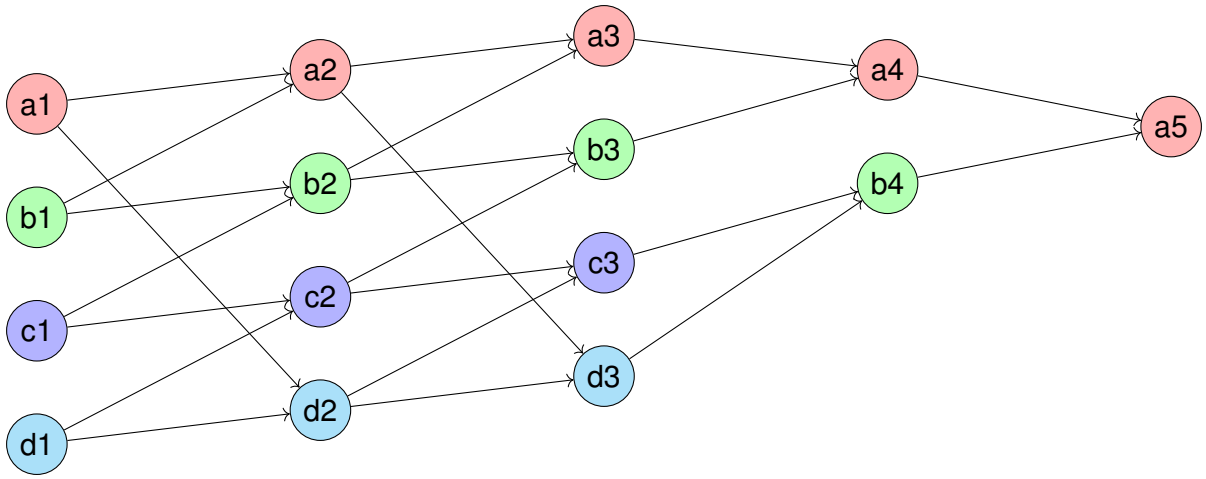


FIGURE 5.12 – Exemple de topologie multi-couche avec trois couches complètes

Nous allons cette fois-ci présenter un réseau plus complexe, dont le but sera d'être plus robuste aux FAs que les précédents. Sa consommation sera donc plus importante, mais il permettra de limiter les réveils du noeud IoT associé, limitant donc sa consommation, bien plus grande que celle de la WuR. Il nous faudra donc estimer si l'augmentation du nombre d'impulsions à envoyer est compensée par la diminution du nombre de fois où le noeud sera réveillé par erreur.

Modèle de réseau

Pour satisfaire nos objectifs pour ce réseau et obtenir une meilleure sélectivité, nous avons opté pour une topologie plus grande et plus complexe. Afin d'éviter d'avoir plusieurs séquences d'activation, nous avons créé des couches plus grandes, dans lesquelles la majorité des neurones ont plusieurs sorties. Le but est d'intriquer les neurones entre eux, afin que chaque neurone d'une couche n soit connecté à plusieurs neurones de la couche $n+1$. Grâce à cela, on rajoute des contraintes sur chaque impulsion pour faire déclencher tout le réseau.

Nous avons appelé cette topologie la topologie multi-couche. Sur la figure 5.12, on compte 3 couches de quatre neurones, et deux couches plus petites. Cela inclut la couche d'entrée, celles cachées dans le réseau, puis les trois neurones de sortie qui rappellent la topologie en losange, sans le neurone d'entrée précédent. On a donc sur cette image la représentation d'une topologie multi-couche à 5 couches au total. Pour faire marcher ce réseau parfaitement, il nous faut cependant 4 entrées différentes, branchées sur $a1$, $b1$, $c1$ et $d1$ respectivement. Nous avons choisi nos neurones parmi ceux de la table 5.2, et chaque lettre de a à d correspond à un jeu de paramètres de la table. Pour ne pas avoir à se soucier des contraintes sur le choix des différents ISTs, nous avons choisi de prendre une décade d'ISTs différente pour chaque couche, correspondant à un chiffre différent dans les noms des neurones. Ainsi, IST_a, IST_b, IST_c et IST_d correspondent à 4 ISTs différents. Ensuite, il est possible d'obtenir l'IST favori d'un neurone du réseau avec la formule :

$$IST_{xi} = IST_x \times 10^{i-1} \tag{5.2}$$

où x est la lettre parmi a, b, c et d, et i le numéro de la couche.

Afin de déterminer la séquence d'activation de ce réseau, il nous faut regarder les séquences d'activation de chaque neurone et redescendre couche par couche pour déterminer

quel signal il nous faut envoyer à chaque neurone de la première couche. Chaque neurone à l'exception de ceux de la première couche est relié à deux neurones de la couche précédente. Pour qu'un neurone déclenche et envoie une impulsion à la couche suivante, il doit recevoir deux impulsions séparées de son IST favori. Afin de contraindre au maximum la séquence, et limiter au maximum les FAs, nous avons déterminé qu'il serait intéressant de faire en sorte que chacune des deux impulsions provienne d'un neurone différent. Par exemple, pour faire décharger le neurone $a2$, ce dernier recevra une impulsion provenant de $a1$, puis une autre provenant de $b1$, qui arrivera IST_{a2} après la première. En général, chaque neurone N_{xi} recevra d'abord une impulsion du neurone N_{xi-1}, puis recevra une autre impulsion de l'autre neurone auquel il est connecté après un délai de IST_{xi}. Grâce à ce mécanisme, il n'est désormais plus possible que le neurone de sortie, ici $a5$, réponde positivement à plusieurs séquences d'activation différentes.

Dans cet exemple à 5 couches comme représenté en fig 5.13, nous avons besoin que les neurones de la couche 3 déchargent chacun une fois, pour que les deux neurones de la quatrième couche reçoivent une paire d'impulsions chacun, et renvoient une impulsion chacun au dernier neurone $s_4(t)$, qui va transformer la paire qu'il a reçu en une dernière impulsion $s_5(t)$, qui signifie que la séquence a bien été détectée. Sachant que les neurones de la troisième couche reçoivent chacun une paire d'impulsions, la deuxième couche doit recevoir deux paires d'impulsions de la sortie de la première couche. Finalement, chaque neurone de la première couche doit recevoir quatre paires d'impulsions pour permettre à l'ensemble du réseau de pouvoir détecter cette séquence. En résumé, il nous faut donc avoir 4 entrées différentes envoyant chacune une séquence différente de huit impulsions à la première couche, pour faire fonctionner cette topologie. On a donc 4 fois plus d'impulsions que précédemment, pour 5 fois plus de neurones.

Nous avons construit la fig 5.13 en représentant les timings d'impulsions en sortie de chaque couche de neurones. La couleur de chaque impulsion correspond à la couleur sur la fig 5.12 du neurone qui l'a tirée dans la couche correspondante. Par exemple pour la troisième ligne de la figure, la première impulsion est rouge, donc elle provient du neurone $a3$. Après une durée IST_{a4}, une impulsion (verte) est tirée par $b3$. De même, les impulsions bleu et cyan sont envoyées par $c3$ et $d3$ respectivement et sont séparées d'un temps IST_{b4}. On peut noter que la ligne en dessous nous indique que ces deux paires sont espacées de IST_{a5}. A partir de ces graphiques, on peut retrouver la séquence que doit envoyer chaque neurone d'entrée à ceux de la première couche. En effet, chaque impulsion de la ligne $s_1(t)$ est engendrée par un neurone de la première couche, dont l'IST est simplement IST_a, IST_b, IST_c ou IST_d selon s'il est rouge, vert, bleu ou cyan respectivement. Par exemple pour le neurone d'entrée relié à $a1$, il suffit de déterminer les timings d'impulsions dessinés en rouge. Chacun de ces timings correspond au second de la paire sensée déclencher $a1$. Le premier de chacune de ces paires sera donc IST_a plus tôt.

Ici, en utilisant la table 5.2, on peut utiliser 6 neurones différents par couche. Cela nous donne un nombre de configurations possibles de $\binom{6}{4}^3 \times \binom{6}{2} \times \binom{6}{1} = 15^4 \times 6 = 303750$. Il nous est donc possible d'adresser 303750 noeuds différents, avec le même nombre de séquences différentes. C'est beaucoup plus que les 216 que l'on avait avec la topologie en ligne, même si l'on avait que 3 neurones. On a cependant 5 fois plus de neurones et 4 fois plus d'impulsion par séquence. On a donc approximativement 20 fois plus d'impulsions opérées par notre réseau par rapport à la topologie en ligne, et le double si l'on compare au réseau en losange. On souhaite donc avoir au moins le même rapport entre la probabilité de FA de cette nouvelle topologie et les précédentes.

Un tableau résumant les principales différences entre les topologies étudiées peut être

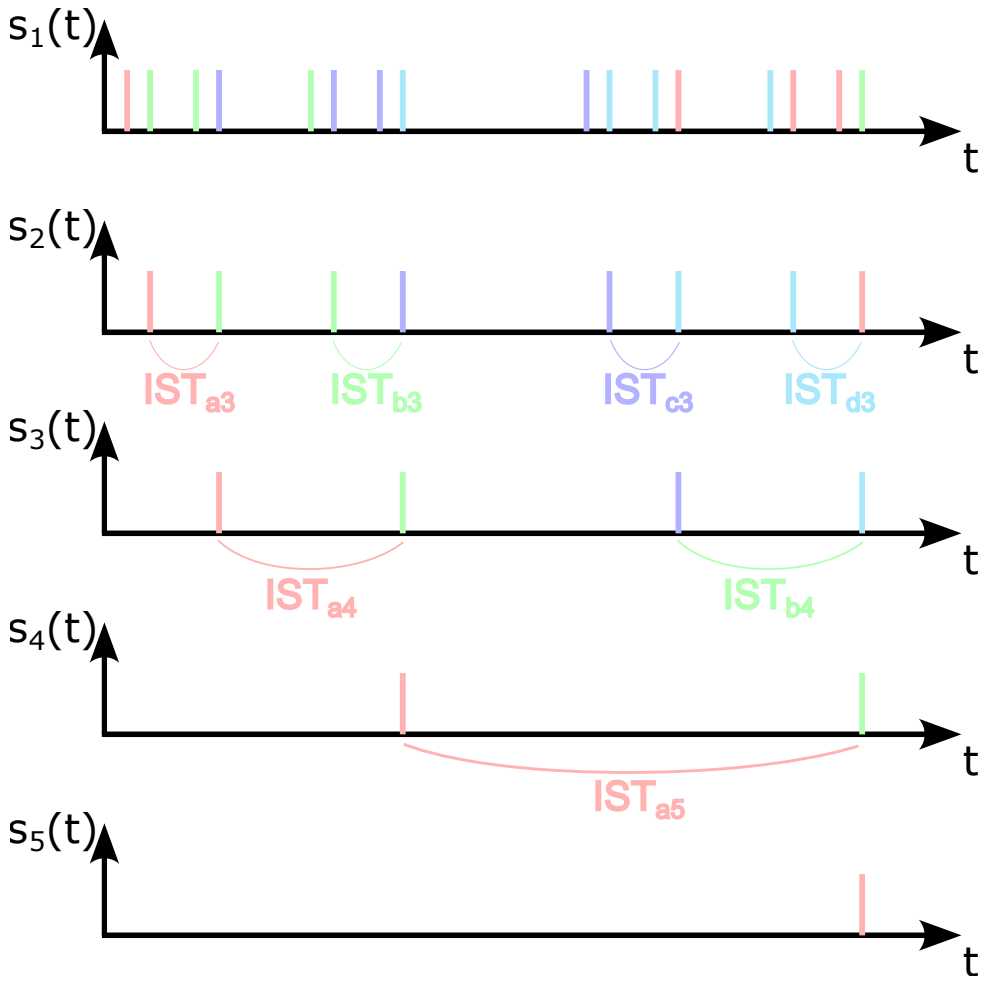


FIGURE 5.13 – Timings d'impulsions en sortie de chaque couche de neurones

retrouvé en table 5.3.

TABLE 5.3 – Paramètres et métriques des 3 topologies étudiées

Topologie	Nombre neurones	Nombre séquences	Nombre impulsions par séquence
Ligne	3	1	8
Losange	3	1	4
Multi-couche	15	4	8

Performances

Nous démarrons l'évaluation des performances par la détermination de la probabilité de MD. Nous avons réalisé ces simulations en utilisant un réseau dont les neurones ont été choisis en partie aléatoirement à chaque itération. A chaque fois, on choisit quatre des six neurones de la table 5.2 auxquels on assigne les lettres a, b, c et d, puis nous construisons notre réseau à l'aide de ces quatre jeux de paramètres, que l'on adaptera d'une décade à chaque passage à la couche supérieure. Ensuite, on reproduit la même expérience, on

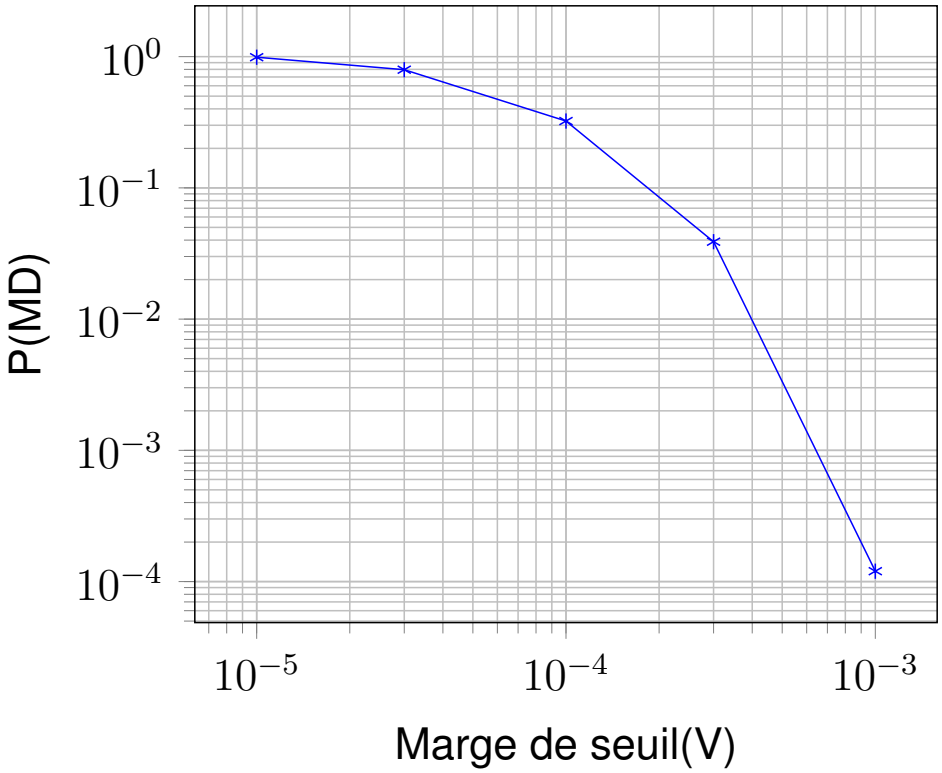


FIGURE 5.14 – Probabilité de MD en fonction de la marge de seuil pour un jitter tiré selon $\mathcal{N}(1,\, 0.01)$.

va construire la séquence correspondant au réseau, puis lui ajouter un jitter fréquentiel, provenant toujours d'un tirage suivant une loi normale $\mathcal{N}(1,\, 0.01)$.

Les résultats tracés en fig 5.14 montrent une probabilité de MD plus élevée que ceux obtenus pour les deux précédentes topologies. Cela s'explique par le fait que l'on ait beaucoup plus d'impulsions à chaque itération, 32 ici pour 8 auparavant, et des temporalités s'étalant sur 5 décades. De fait, il n'est plus possible que les TWs se superposent pour des grandes valeurs de marge de seuil, ce qui augmente la sélectivité de notre réseau, mais implique d'avoir une plus grande P(MD). Au maximum, pour une marge de seuil de $1mV$, la probabilité de MD est multipliée par 10. Néanmoins, celle-ci reste relativement faible, surtout dans des applications où l'on veut économiser de l'énergie, et où une MD n'est pas critique.

Afin de pouvoir vérifier si l'évolution de la probabilité de MD était compensée par une diminution des FAs, nous avons réalisé les mêmes simulations que dans les parties précédentes. Nous avons donc envoyé à chacun des 4 neurones d'entrée une séquence différente composée chacune de 8 impulsions dont les timings ont été tirés selon une loi uniforme entre $t = 0$ et le timing de la dernière impulsion de la séquence ciblée. Nous avons fait tourner ces simulations 100000 fois, puis analysé les résultats obtenus. Nous n'avons obtenu aucune occurrence de FA, même pour la marge de seuil à $10^{-3}V$. Nous avons validé ce résultat en traçant la courbe de probabilité théorique de FA à l'aide de la formule en eq. (5.1). Cette courbe peut être observée en fig 5.15.

On remarque que la probabilité d'occurrence d'une FA dans nos simulations est, pour le cas favorable aux FAs, de l'ordre de 10^{-70}. Sachant que nos simulations sont composées de 10^5 à 10^6 itérations, il nous est pratiquement impossible d'observer une FA. Pour les topologies précédentes, l'estimation théorique descendait au mieux à une probabilité de

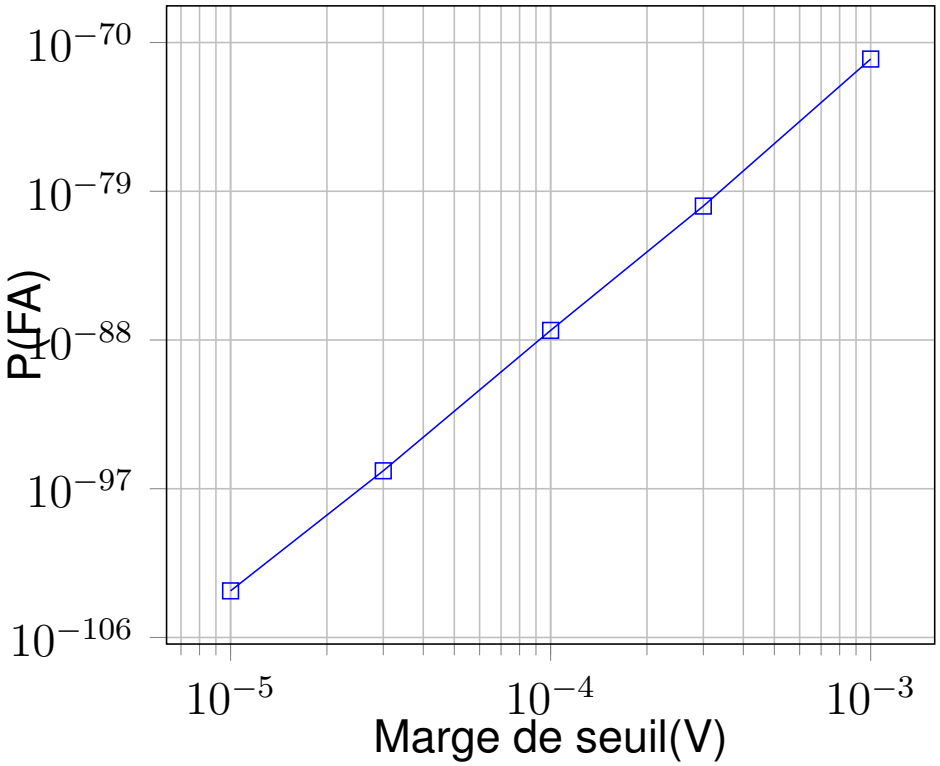


FIGURE 5.15 – Probabilité théorique de FA de la topologie multi-couche en fonction du seuil de marge

FA de 10^{-12} avec un réseau en ligne de 3 neurones et pour une marge de seuil de $10^{-5}V$. Cela valide notre hypothèse selon laquelle, en complexifiant notre réseau et en considérant des séquences plus longues par rapport aux ISTs utilisés, nous allions drastiquement diminuer la probabilité de FA. Ces résultats s'expliquent par le fait que l'on envoie 4 fois plus d'impulsions, sur une fenêtre temporelle 100 fois plus grande. Il devient donc très improbable de réussir à placer correctement des impulsions dans des TWs d'une largeur de l'ordre de $10^{-6}s$, lorsque l'on tire des timings dans une plage large de $0.1s$. Un de nos objectifs était de réaliser un réseau qui serait beaucoup plus robuste aux FAs que les topologies précédentes, sans toutefois devenir imperméable à toutes les séquences. Si on compare les performances observables en fig 5.10 avec celles en fig 5.14, on peut observer un facteur 10 entre les probabilités de MD des deux topologies. Cependant, si on compare maintenant les probabilité de FA entre la fig 5.11 et la fig 5.15, ce facteur devient 10^{70} à 10^{100} et cette fois ci en faveur de la topologie multi-couche. Cela vient largement compenser la faible réduction de performances en termes de MD. Notre objectif étant de limiter la consommation de notre réseau, si celui ci devient extrêmement sélectif au point de ne plus subir de FA, quitte à manquer quelques communications, cela devrait réduire la consommation globale, malgré le fait que chaque séquence contienne 4 fois plus d'impulsions. En effet, la puissance d'un micro-contrôleur classique est de l'ordre du mW alors que celui de nos neurones est de la centaine de pW. Si on utilise 15 neurones et 4 fois plus de séquences, on multiplie la consommation par approximativement 60. Mais, la consommation des micro-contrôleurs est de l'ordre de 10^5 à 10^6 supérieure à celle de la WuR. Dans ce cas, la réduction de la probabilité de FA contrebalance largement la complexification du réseau.

Enfin, afin de vérifier que chaque neurone permet la création d'une séquence tota-

lement orthogonale aux autres, au moins sur des marges de seuil inférieures ou égales à $0.1mV$, nous avons construit 1000 réseaux parmi les 303750 possibles, et leur avons envoyé 10000 séquences différentes de celle ciblée. Nous n'avons pas pu tester les $303750 \cdot 303749$ combinaisons pour des raisons de temps de calcul trop long. Encore une fois, nous n'avons pas obtenu la moindre occurrence de FA.

Cela nous permet de valider que nous avons réussi à implémenter un réseau plus complexe répondant à notre objectif initial, réaliser une topologie de réseau très robuste aux FAs, tout en restant assez robuste aux MDs. Nous avons pu vérifier qu'augmenter le nombre de neurones, le nombre d'impulsions par séquence, la longueur temporelle de la séquence, et les intrications entre neurones, ont permis à notre réseau de devenir beaucoup plus sélectif sur la séquence qui le fait décharger. Notre objectif est d'utiliser ce réseau de neurones dans le domaine de l'IoT. Dans ce contexte, nous aurons besoin que la consommation de la WuR soit la plus faible possible, afin que sa batterie tienne le plus longtemps possible. Toutefois, il n'est pas prévu que son utilisation soit un élément critique d'une quelconque application. Dans ce cas, on veut faire en sorte que notre appareil se réveille quand on communique avec lui, mais ce n'est pas très grave si de manière sporadique, il est nécessaire d'envoyer deux fois la séquence pour que le réseau réveille le noeud correspondant. A l'inverse, chaque FA résultera en un gâchis de consommation qui aurait pu être évité. C'est pourquoi cette dernière topologie semble être la plus adaptée pour réaliser une WuR à base de neurones SLIF.

5.4 Conclusion

Notre étude a porté sur l'utilisation de la dynamique du neurone SLIF pour la création d'un réseau de neurones à impulsions capable de réaliser une fonction de reconnaissance de signature. Nous avons réalisé le design de topologies de réseaux ainsi que la création de séquences adaptées aux réseaux considérés. Dans un premier temps, nous avons étudié le co-design de la séquence et du réseau sur une topologie très simple, dans laquelle les neurones étaient placés en ligne, dans une configuration totalement feed-forward. Cela nous a permis de comprendre comment construire la séquence en se basant sur la topologie et les paramètres de chaque neurone utilisé. Nous avons pu observer les différentes contraintes qui se posaient sur le choix des différents neurones, principalement sur le choix de l'IST à employer. Nous avons ensuite réalisé plusieurs expérimentations pour pouvoir mesurer les performances de ce réseau en termes de probabilité de MD et de FA. Enfin, nous avons mis au point un jeu de paramètres de neurones, afin d'avoir un choix de neurones dont les ISTs se trouvent dans la même décades, et dont les TWs se chevauchent pas. Cela nous a permis de définir le nombre maximal de séquences orthogonales disponibles par décades. Dans le cas d'un réseau en ligne de 3 neurones, cela permet d'avoir 216 différents réseaux et séquences disponibles pour adresser des noeuds IoT.

Nous avons ensuite réalisé deux topologies différentes, sensées chacune optimiser une des deux performances mesurées. Dans le but de minimiser la probabilité de MD, nous avons designé la topologie dite en losange. Cette topologie composée d'une couche à deux neurones, reliés eux mêmes à un neurone de sortie sur la deuxième couche. Cette topologie nous a permis d'observer plusieurs limites de notre système. Tout d'abord, il ne nous était pas possible de différencier lequel des deux neurones de la couche intermédiaire provenait chaque impulsion reçue par le neurone de sortie. A cause de cela, ce neurone ne pouvait différencier s'il avait reçu deux impulsions du même neurone ou de deux neurones différents, et dans ce dernier cas, lequel des deux neurones était responsable de chaque

impulsion. En conséquent, 4 séquences différentes activaient le réseau. De plus, le neurone de sortie avait été initialement choisi pour illustrer un neurone au TWs très large, ce qui engendre une très faible sélectivité. Ces deux facteurs ont fait que notre réseau était extrêmement peu sélectif, et que sa probabilité de FA était assez élevée. Néanmoins, du fait de la faible sélectivité, notre neurone était extrêmement robuste en termes de MD. En changeant de neurones, nous avons vu qu'avec des neurones choisis plus méticuleusement pour avoir un TW plus fin en comparaison à son IST, le réseau était un peu moins robuste aux MDs, mais subissait moins de FAs.

Enfin, de manière à se placer dans un contexte plus proche de l'application dans le domaine de l'IoT, nous avons réalisé un réseau plus robuste aux FAs, afin d'éviter que notre réseau ne réveille trop souvent le noeud IoT. Pour cela nous avons mis au point une topologie plus complexe, nommée topologie multi-couche, composée de bien plus de neurones que les précédentes. Nous avons ainsi étudié comment construire chacune des 4 séquences à partir de ce réseau dont les neurones sont enchevêtrés. Nous avons ensuite mesuré les performances de cette topologie en termes de MD et de FA. Nous avons pu voir que ce réseau était moins performant que le précédent en termes de MD, mais d'un coefficient assez peu élevé. A l'inverse, la probabilité de FA était quasi inexistante, tombant de 10^{-4} environ à un ordre de grandeur de 10^{-100}. Nous avons donc montré que complexifier le réseau, rajouter des neurones, les intriquer et allonger la séquence permettait d'obtenir de bien meilleures performances de sélectivité, sans toutefois beaucoup amoindrir les chances de MDs.

Cette topologie nécessite 4 entrées envoyant chacune une séquence différente pour être activée, et utilise 15 neurones lorsqu'on utilise 3 couches intermédiaires, comme dans nos simulations. Cela augmente donc la consommation du réseau à chaque réception, comparé aux topologies précédentes. Cependant, la probabilité de FA est très fortement réduite ce qui peut participer à la réduction de consommation d'énergie, surtout si le récepteur a tendance à transformer fréquemment du bruit en un signal. Dans ce cas, notre réseau sera capable de ne pas se réveiller inutilement, contrairement à un réseau classique qui pourrait être activé plus souvent par des FAs, augmentant ainsi la consommation d'énergie.

Un tel réseau sera donc parfaitement adapté pour la création d'une WuR basée sur un SNN de neurones SLIF une fois que l'on sera capable de reproduire le phénomène de saturation synaptique avec des composants ultra faible puissance.

Chapitre 6

Conclusion

L'objectif de cette thèse était de développer et d'évaluer des architectures de réseaux de neurones à impulsions adaptées aux WuRs pour l'IoT, afin d'améliorer l'efficacité énergétique et la robustesse dans des environnements contraints en énergie. En s'appuyant sur des principes bio-inspirés, ce travail visait à proposer des solutions neuromorphiques pour le traitement en temps réel et la détection de signaux dans des contextes de faible consommation d'énergie.

6.1 Résumé des Contributions

Notre première contribution, présentée dans le chapitre 3 a été de réaliser un SNN capable de répondre aux contraintes du projet UWAKE. Il était question de recevoir un signal et de détecter s'il correspondait bien à la séquence cible ou non. Les neurones que nous avions à notre disposition sont inspirés du modèle Morris Lecar, et pouvaient très facilement être adaptés en LIF. Il était également prévu que le signal puisse être parallélisé et que nous puissions utiliser les mécanismes de délai synaptique et de période réfractaire. A partir de ces informations, nous avons designé une topologie de neurones qui reproduit le fonctionnement d'un corrélateur [51, 52]. Nous avons montré que ce réseau était capable de reconnaître une séquence spécifique dans un train d'impulsion, et de discriminer les autres séquences. Nous avons étudié l'impact de plusieurs paramètres, le type de code, la tension de seuil et la durée d'un bit, sur les performances en termes de probabilité de MD et de FA au travers de plusieurs scénarios de simulation.

Finalement, les contraintes que nous avions ont changé : il n'était plus possible ni de paralléliser le signal, ni de mettre en place un mécanisme de délai synaptique. Notre topologie précédente nécessitait que les impulsions arrivent de manière synchronisée aux différents neurones pour permettre une reconnaissance de toute la séquence. Nous avons donc dû chercher une autre solution, qui puisse reconnaître un signal série sans avoir besoin de synchroniser l'arrivée des impulsions entre les neurones.

Nous avons donc exploré différents modèles de neurones pour pallier à ces nouvelles contraintes. Sans trouver de solution bio-inspirée et réalisable en termes de modèle de neurone, nous avons étudié avec l'aide d'un neuroscientifique le modèle de synapses saturantes [50]. Nous nous sommes aperçus que ce phénomène, issu de l'effet des dendrites sur les cellules pyramidales, permettait une réponse dépendant du Timing Inter-Impulsions (IST). C'est à dire qu'il peut avoir des réponses différentes à des signaux temporels différents, et donc de les discriminer grâce à cet IST. Nous avons réalisé une étude en profondeur de ce modèle et de ses mécanismes afin de réaliser un système composé d'un neurone LIF

et d'une synapse saturante capable de reconnaître des séquences temporelles composées de deux impulsions [53, 54]. Nous avons ensuite étendu cette étude à des séquences de plus de deux ISTs et évalué l'évolution des performances du réseau en fonction du nombre d'ISTs dans la séquence. Nous avons également regardé l'évolution des performances dans l'espace des paramètres principaux du modèle, pour plusieurs types de séquence d'activation. Nous avons validé la possibilité d'utiliser ce modèle SLIF dans le cadre des WuRs pour reconnaître une séquence basée sur des ISTs.

Notre dernière contribution a été l'étude de l'utilisation de ce modèle pour la création d'un SNN. Nous avons commencé par créer un réseau très simple composé de quelques neurones composé de couches d'un unique neurone à chaque fois [55]. Nous avons travaillé sur l'élaboration de séquences en co-design avec la topologie choisie, et vérifié que celle-ci était bien reconnue. Nous avons testé ce réseau sur plusieurs scénarios différents afin de pouvoir quantifier en simulation les probabilités de MD et FA pour des versions à peu de neurones de cette topologie.

Nous avons dans un second temps proposé une topologie qui nous permettrait de réduire la consommation en diminuant le nombre d'impulsions requises pour activer le neurone de sortie, tout en gardant le même nombre de neurones. Notre topologie en losange nous a permis de diviser par deux le nombre d'impulsions pour un nombre constant de neurone. Malheureusement, nous avons rencontré un certain nombre de contraintes avec cette topologie. Nous nous sommes d'abord rendu compte que ce réseau était activé par 4 séquences différentes plutôt qu'une. Nous avons également observé qu'avoir un neurone avec un très grand TW altérait encore plus qu'avant les performances de tout le réseau. Nous avons grandement diminué la probabilité de MD, déjà assez faible, mais largement augmenté celle de FA. En conséquence, le réseau aurait réveillé le noeud IoT bien plus souvent que prévu, entraînant une augmentation de la consommation de ce noeud malgré le fait que l'on ait divisé par deux le nombre d'impulsions, et donc la consommation du SNN. Nous nous sommes donc finalement focalisés sur un réseau plus complexe capable de diminuer drastiquement les FAs.

Cette dernière topologie est composée de plus de neurones et nécessite plus d'impulsions pour être activée, ce qui augmente la consommation théorique du réseau. Toutefois, nous avons pu montrer que la probabilité de FA était devenu quasiment nulle, sans pour autant altérer énormément la probabilité de MD. C'est pourquoi nous avons considéré l'utilisation de cette topologie dans un milieu bruité afin de limiter au maximum les réveils intempestifs du noeud associé.

6.2 Impact et Limitations

Dans le cadre du projet UWAKE, nous avons réussi à implémenter un réseau neuromorphique capable d'être utilisé pour reconnaître une séquence. Celui-ci pourra être couplé au module de récupération d'énergie RF [56, 57] pour être utilisé comme WuR. Le projet devrait donc pouvoir être mené à bien et les résultats sont encourageants. La consommation du système complet devrait être de l'ordre du nW comme estimé dans le sujet, ce qui est inférieur aux solutions de WuR actuelles dont la puissance est plutôt autour de $100\mu W$ [11, 58].

De plus, nous avons proposé le modèle SLIF pour palier à l'absence de délai synaptique, mais il n'était pas possible de réaliser des synapses saturantes avec la technologie très basse puissance de nos collaborateurs de l'IEMN. Toutefois, il nous faudrait estimer si la réalisation de ces synapses avec des composants plus consommateurs nous permettrait

quand même d'économiser de l'énergie par rapport au système actuel de parallélisation du signal.

Chapitre 7

Perspectives Futures

7.1 Prolongements Immédiats

Au cours de ces trois années nous avons exploré un certain nombre d'idées, mais il en reste au moins autant que nous n'avons pas eu le temps d'étudier. Dans le prolongement des dernières contributions, nous aurions voulu coupler à la fois les séquences à plusieurs ISTs successifs pour déclencher un neurone, et l'utilisation d'un réseau de plusieurs neurones. Cela aurait pu nous permettre de rendre les séquences plus complexes sans avoir à ajouter de neurones, et nous permettrait d'améliorer les performances pour une même taille de réseau

Ensuite, il serait aussi possible de rajouter un mécanisme d'apprentissage dans notre système. Dans le modèle actuel, la conductance synaptique est capée à g_s^{max} qui est une constante. Nous avons pensé à faire varier g_s^{max} en fonction de l'activité synaptique, de la manière que la méthode d'apprentissage STDP. De cette manière, on pourrait diminuer g_s^{max} lorsque l'activité est trop importante pour que certains neurones deviennent plus sélectifs, ou au contraire l'augmenter si l'on est dans des milieux bruités qui empêchent le réseau de décharger à cause de la variabilité de l'environnement.

7.2 Amélioration des Architectures SNN

Dans un second temps, nous aurions voulu complexifier notre modèle SLIF en rajoutant d'autres mécanismes biologiques afin de pouvoir utiliser et améliorer les performances du réseau. Par exemple, nous pourrions rajouter une période réfractaire, plus spécifiquement sur la topologie en losange, pour éviter qu'il soit possible que les séquences faisant décharger deux fois un des deux neurones de la première couche n'activent le réseau. Cela aurait obligé chacun de ces neurones à décharger une fois pour faire décharger le neurone de sortie, ce qui aurait diminué la probabilité de fausses alarmes. Nous n'avons pas gardé cette idée au premier abord car cela implique que si du bruit fait décharger un neurone peu de temps avant la réception de la séquence cible, la période réfractaire nous aurait fait rater la détection. Toutefois ce cas est très improbable et il nous aurait fallu mesurer si l'augmentation de la probabilité de MD contrebalançait la diminution de celle de FA.

De plus, nous avons montré qu'il était possible de réaliser des topologies répondant à certaines contraintes. Nous avons diminué le nombre d'impulsions pour réveiller le réseau sans augmenter le nombre de neurones, puis nous avons réalisé un réseau diminuant fortement la probabilité de FA. Toutefois, nous n'avons sûrement pas trouvé le réseau le plus optimal dans chacun de ces cas. Il pourrait être intéressant d'utiliser des méthodes

d'optimisation d'ANN pour déterminer des topologies optimales, sans avoir un trop grand nombre de neurones ou d'impulsions dans chaque séquence, afin de continuer à avoir pour objectif une consommation très faible.

Enfin, nous avons étudié des topologies capables de réveiller un noeud IoT ayant une signature unique. Cependant, il arrive que l'on veuille réveiller un groupe de noeuds en même temps. Il serait donc intéressant d'étudier si la reconnaissance de plusieurs séquences différentes à travers plusieurs neurones de sortie serait envisageable, et comment cela altérerait les performances par rapport à l'utilisation de plusieurs WuRs.

7.3 Nouvelles Applications

Assez logiquement, les avancées dans le domaine des réseaux de neurones à impulsions découlent des avancées en neurosciences computationnelles. Le cerveau reste l'organe le moins bien compris par la science, et des découvertes sont réalisées régulièrement. La découverte de nouveaux modèles de neurones, de synapses, ou de nouveaux phénomènes bio-inspirés pourraient permettre la mise en place de nouveaux mécanismes dans les réseaux de neurones à impulsion, et permettraient sûrement l'utilisation des SNNs dans de nouvelles applications. Par exemple, si le fonctionnement du cerveau était totalement compris, il serait sûrement possible de relier un SNN à un cerveau humain pour réparer des lésions ou des manques dans le cerveau, ou reconnecter des membres dont on aurait perdu l'usage.

7.4 Technologies Futures

Le développement des puces neuromorphiques a commencé il y a quelques années [14, 59] mais celles ci restent encore rares. Pourtant, celles ci sont très importantes pour le développement du domaine, car elles permettent de tester les différences entre la simulation sur ordinateur, et la simulation sur circuit. Le progrès sur la réalisation à grande échelle de ces puces permet de démocratiser ce genre d'architecture, et de pousser la recherche encore plus loin dans cette direction.

De plus, la recherche sur les ces puces pourrait permettre d'encore plus optimiser la consommation et la puissance de ces circuits, ce qui pourrait leur permettre de remplacer les circuits classiques dans certains domaines comme l'IoT.

7.5 Défis Restants

De nombreux progrès restent à faire quant aux réseaux de neurones à impulsions, autant sur l'aspect théorique que sur l'aspect matériel. En effet, les SNNs ont toutes les qualités pour devenir un standard dans les années à venir si la recherche continue leur étude. Cependant, il faudrait pour cela avoir une compréhension bien meilleure de tous les mécanismes neuronaux afin de mettre en place des procédés de reconnaissance et d'apprentissage bien plus efficaces en termes de consommation d'énergie.

Le développement d'outils logiciels pour la simulation de neurones à impulsions pourrait également faire avancer la recherche dans ce domaine, car les outils actuels comme Brian2 [36] ou SNNTorch [60] restent encore très récents et peu intuitifs.

Enfin, l'étude de la scalabilité des architectures de SNN pourrait également permettre l'utilisation de cette technologie dans un plus grand nombre de domaines.

L'augmentation exponentielle du nombre d'objets connectés dans notre société entraîne des besoins énergétiques sans précédent, et ces circuits très basse puissance pourraient être une solution à envisager si l'on doit continuer à vivre dans cette société surconsommatrice dont les ressources ne sont pas infinies.

Contributions

Ce travail a conduit aux publications suivantes :

[51] Guillaume Marthe, Claire Goursaud, and Laurent Clavier. Wake-up radio receiver based on spiking neurons for detecting activation sequence. In 2023 IEEE Wireless Communications and Networking Conference (WCNC), 2023

[53] Guillaume Marthe, Claire Goursaud, Romain Cazé, and Laurent Clavier. On exploiting the synaptic interaction properties to obtain frequency-specific neurons. In 2023 IEEE 16th Dallas Circuits and Systems Conference (DCAS), pages 1–6, 2023. Best Paper Award

[52] Guillaume Marthe, Claire Goursaud, Laurent Clavier. Récepteur Wake-up radio basé sur des réseaux de neurones à impulsions pour la détection de séquence. CoRes 2023-8emes Rencontres Francophones sur la Conception de protocoles, l'evaluation de performances et l'experimentation de Reseaux de communication. Pages 1-4, 2023

[54] Guillaume Marthe, Claire Goursaud. Exploitation des propriétés de saturation synaptique pour obtenir un neurone à fréquence spécifique. Gretsi 2023, Aug 2023, Grenoble, France

[55] Guillaume Marthe, Claire Goursaud, and Laurent Clavier. Enabling low-power signature recognition for the IoT with SLIF neurons. In 2024 European Signal Processing Conference (EUSIPCO), 2024.

Un article de journal est aussi en cours de soumission.

Bibliographie

[1] J Alec. Industrial iot : Unleashing potential of connected factories. Industrial Automation and AI, 2017.

[2] Eastern Peak. Iot in agriculture : 9 technology use cases for smart farming (and challenges to consider). Eastern Peak Blog, 2020.

[3] Halil Yetgin, Kent Tsz Kan Cheung, Mohammed El-Hajjar, and Lajos Hanzo Hanzo. A survey of network lifetime maximization techniques in wireless sensor networks. IEEE Communications Surveys & Tutorials, 19(2) :828–854, 2017.

[4] Pei Huang, Li Xiao, Soroor Soltani, Matt W. Mutka, and Ning Xi. The evolution of mac protocols in wireless sensor networks : A survey. IEEE Communications Surveys & Tutorials, 15(1) :101–120, 2013.

[5] Nafiseh Seyed Mazloum and Ove Edfors. Performance analysis and energy optimization of wake-up receiver schemes for wireless low-power applications. IEEE Transactions on Wireless Communications, 13(12) :7050–7061, 2014.

[6] Dora Spenza, Michele Magno, Stefano Basagni, Luca Benini, Mario Paoli, and Chiara Petrioli. Beyond duty cycling : Wake-up radio with selective awakenings for long-lived wireless sensing systems. In 2015 IEEE Conference on Computer Communications (INFOCOM), pages 522–530, 2015.

[7] Ilker Demirkol, Cem Ersoy, and Ertan Onur. Wake-up receivers for wireless sensor networks : benefits and challenges. IEEE Wireless Communications, 16(4) :88–96, 2009.

[8] Michele Magno and Luca Benini. An ultra low power high sensitivity wake-up radio receiver with addressing capability. In 2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pages 92–99, 2014.

[9] Michele Magno, Vana Jelicic, Bruno Srbinovski, Vedran Bilas, Emanuel Popovici, and Luca Benini. Design, implementation, and performance evaluation of a flexible low-latency nanowatt wake-up radio receiver. IEEE Transactions on Industrial Informatics, 12(2) :633–644, 2016.

[10] Ilias Sourikopoulos, Sara Hedayat, Christophe Loyez, François Danneville, Virginie Hoel, Eric Mercier, and Alain Cappy. A 4-fj/spike artificial neuron in 65 nm cmos technology. Frontiers in Neuroscience.

[11] F. Danneville, C. Loyez, K. Carpentier, I. Sourikopoulos, E. Mercier, and A. Cappy. A sub-35 pw axon-hillock artificial neuron circuit. Solid-State Electronics, 153 :88–92, 2019.

[12] Mingzhe Chen, Ursula Challita, Walid Saad, Changchuan Yin, and Mérouane Debbah. Artificial neural networks-based machine learning for wireless networks : A tutorial. IEEE Communications Surveys & Tutorials, 21(4) :3039–3071, 2019.

[13] Timothy O'Shea and Jakob Hoydis. An introduction to deep learning for the physical layer. IEEE Transactions on Cognitive Communications and Networking, 3(4) :563–575, 2017.

[14] Mike Davies et Al. Loihi : A neuromorphic manycore processor with on-chip learning. IEEE Micro, 38(1) :82–99, 2018.

[15] Wulfram Gerstner, Werner M. Kistler, Richard Naud, and Liam Paninski. Neuronal Dynamics : From Single Neurons to Networks and Models of Cognition. Cambridge University Press, 2014.

[16] Wolfgang Maass. Networks of spiking neurons : The third generation of neural network models. Neural Networks, 10(9) :1659–1671, 1997.

[17] Michael Hopkins, Garibaldi Pineda García, Petrut Bogdan, and Stephen Furber. Spiking neural networks for computer vision. Interface Focus, 8(4), August 2018.

[18] Peter Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. Frontiers in Computational Neuroscience, 9, 2015.

[19] Xu Zhang, Z Xu, C Henriquez, and S Ferrari. Spike-based indirect training of a spiking neural network-controlled virtual insect. 12 2013.

[20] Rodney Douglas Giacomo Indiveri, Elisabetta Chicca. Memory and information processing in neuromorphic systems. Proceedings of the IEEE, 103(8) :1379–1397, 2015.

[21] Denis Fize Simon Thorpe and Catherine Marlot. Spikenet : Real-time visual processing with one spike per neuron. Neurocomputing, 38-40 :1049–1054, 2001.

[22] Markus Meister Tim Gollisch. Rapid neuronal computation in the retina using relative spike times. Nature, 451 :720–724, 2008.

[23] Ilyass Hammouamri, Ismail Khalfaoui-Hassani, and Timothée Masquelier. Learning delays in spiking neural networks using dilated convolutions with learnable spacings, 2023.

[24] Louis Lapicque. Recherches quantitatives sur l'excitation électrique des nerfs traitée comme une polarisation. J. Physiol. Pathol. Gen, 9 :620–635, 1907.

[25] Anthony N Burkitt. A review of the integrate-and-fire neuron model : I. homogeneous synaptic input. Biological cybernetics, 95(1) :1–19, 2006.

[26] Sophie Deneve Romain Caze. Robust transmission in neural circuits with unreliable synapses. Frontiers in Computational Neuroscience, 7 :91, 2013.

[27] Alan L. Hodgkin and Andrew F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. The Journal of physiology, 117(4) :500–544, 1952.

[28] Peter Dayan and L.F. Abbott. Theoretical Neuroscience : Computational and Mathematical Modeling of Neural Systems. MIT Press, 2001.

[29] Apostolos P. Georgopoulos, Andrew B. Schwartz, and Robert E. Kettner. Neuronal population coding of movement direction. Science, 233(4771) :1416–1419, 1986.

[30] Mu-ming Poo Guo-qiang Bi. Synaptic modifications in cultured hippocampal neurons : dependence on spike timing, synaptic strength, and postsynaptic cell type. Journal of Neuroscience, 18(24) :10464–10472, 1998.

[31] Wulfram Gerstner Abigail Morrison, Markus Diesmann. Spike-timing-dependent plasticity in balanced random networks. Neural computation, 21(2) :250–260, 2008.

[32] Matthew Cook Peter U Diehl. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. International Joint Conference on Neural Networks (IJCNN), pages 1–8, 2015.

[33] Andrei Lavric and Andrei Petrescu. Reinforcement learning for autonomous navigation of a spiking neural network controlled robot. 2017 IEEE International Conference on Robotics and Automation (ICRA), pages 123–130, 2017.

[34] Xiaofei Liu and Fang Li. Energy-efficient spiking neural network with reinforcement learning for sensor networks. IEEE Transactions on Neural Networks and Learning Systems, 26(11) :2642–2650, 2015.

[35] Răzvan V. Florian. Reinforcement learning through modulation of spiking activity in spiking neural networks. Neural Computation, 19(6) :1468–1502, 2007.

[36] Marcel Stimberg, Romain Brette, and Dan Goodman. Brian 2, an intuitive and efficient neural simulator. eLife, 8, 08 2019.

[37] Gerard J. M. Smit Liang Liu, Eric Klumperink and Bram Nauta. A cmos wake-up receiver for wireless body-area networks. IEEE Journal of Solid-State Circuits, 50(1) :320–330, 2016.

[38] Kaushik Roy Priyadarshini Panda. Energy-efficient spiking neural networks for reconfigurable architectures. IEEE Transactions on Neural Networks and Learning Systems, 27(9) :1907–1919, 2016.

[39] Thomas Barbier, Céline Teulière, and Jochen Triesch. Unsupervised Learning of Spatio-Temporal Receptive Fields from an Event-Based Vision Sensor. In 29th International Conference on Artificial Neural Networks,, pages 622–633, Bratislava, Slovakia, 2021.

[40] Rodrigo Alvarez-Icaza et al. Paul A. Merolla, John V. Arthur. A million spiking-neuron integrated circuit with a scalable communication network and interface. Science, 345(6197) :668–673, 2014.

[41] Tsung-Han Lin et al. Mike Davies, Narayan Srinivasa. Loihi : A neuromorphic manycore processor with on-chip learning. IEEE Micro, 38(1) :82–99, 2018.

[42] Mirko Prezioso, Farnood Merrikh-Bayat, Brian D Hoskins, Gianluca C Adam, Konstantin K Likharev, and Dmitri B Strukov. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. Nature, 521(7550) :61–64, 2015.

[43] Abhronil Sengupta and Muhammad A Alam. Proposal for an all-spin artificial neural network : Emulation of synaptic behavior with magnetic tunnel junctions. IEEE Transactions on Electron Devices, 63(7) :2963–2970, 2016.

[44] Amir Tavanaei, Mohammad Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony S Maida. Deep learning in spiking neural networks. Neural networks, 111 :47–63, 2019.

[45] Sara Hedayat, Ilias Sourikopoulos, Christophe Loyez, Francois Danneville, Laurent Clavier, Virginie Hoel, and Alain Cappy. Experimental investigation of stochastic resonance in a 65nm cmos artificial neuron. In 2017 International Conference on Noise and Fluctuations (ICNF), pages 1–4, 2017.

[46] Eric Hunsberger and Chris Eliasmith. Spiking deep networks with LIF neurons. CoRR, abs/1510.08829, 2015.

[47] Senthilkumar Meyyappan, R. Karthik, and I. Raja. An efficient approach for increasing power optimization in mobile ad-hoc networks. International Journal of Engineering Research and Technology (IJERT), 3, 02 2014.

[48] A. L. Hodgkin and A. F. Huxley. Currents carried by sodium and potassium ions through the membrane of the giant axon of loligo. The Journal of Physiology, 116(4) :449–472, 1952.

[49] Etay Hay, Sean Hill, Felix Schürmann, Henry Markram, and Idan Segev. Models of neocortical layer 5b pyramidal cells capturing a wide range of dendritic and perisomatic active properties. PLOS Computational Biology, 7(7) :1–18, 07 2011.

[50] Cazé RD. All neurons can perform linearly non-separable computations. F1000Research, 2022.

[51] Guillaume Marthe, Claire Goursaud, and Laurent Clavier. Wake-up radio receiver based on spiking neurons for detecting activation sequence. In 2023 IEEE Wireless Communications and Networking Conference.

[52] Guillaume Marthe, Claire Goursaud, and Laurent Clavier. Récepteur Wake-up radio basé sur des réseaux de neurones à impulsions pour la détection de séquence. In CoRes 2023-8emes Rencontres Francophones sur la Conception de protocoles l evaluation de performances et l experimentation de Reseaux de communication, Cargese, France, May 2023.

[53] Guillaume Marthe, Claire Goursaud, Romain Cazé, and Laurent Clavier. On exploiting the synaptic interaction properties to obtain frequency-specific neurons. In 2023 IEEE 16th Dallas Circuits and Systems Conference (DCAS), pages 1–6, 2023.

[54] Guillaume Marthe and Claire GOURSAUD. Exploitation des propriétés de saturation synaptique pour obtenir un neurone à fréquence spécifique. In 29° Colloque sur le traitement du signal et des images, number 2023-1366, pages p. 1121–1124, Grenoble, Aout 6 - Sept 9 2023. GRETSI - Groupe de Recherche en Traitement du Signal et des Images.

[55] Guillaume Marthe, Claire Goursaud, and Laurent Clavier. Enabling low-power signature recognition for the iot with slif neurons. In 2024 32nd European Signal Processing Conference (EUSIPCO), pages 1471–1475, 2024.

[56] Jesus Argote-Aguilar, Florin-Doru Hutu, Guillaume Villemaud, Matthieu Gautier, and Olivier Berder. Efficient association of low and high rf power rectifiers for powering ultra-low power devices. In 2022 29th IEEE International Conference on Electronics, Circuits and Systems (ICECS), pages 1–4, 2022.

[57] Jesus Argote, Muh-Dey Wei, Florin Doru Hutu, Guillaume Villemaud, Matthieu Gautier, Olivier Berder, and Renato Negra. Low-input-power sub-ghz rf energy harvester for powering ultra-low-power devices. pages 1–4, 11 2023.

[58] Patrick P. Mercier, Benton H. Calhoun, Po-Han Peter Wang, Anjana Dissanayake, Linsheng Zhang, Drew A. Hall, and Steven M. Bowers. Low-power rf wake-up receivers : Analysis, tradeoffs, and design. IEEE Open Journal of the Solid-State Circuits Society, 2 :144–164, 2022.

[59] Paul A. Merolla et Al. A million spiking-neuron integrated circuit with a scalable communication network and interface. Science, 345, 2014.

[60] Jason K Eshraghian, Max Ward, Emre Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spi-

king neural networks using lessons from deep learning. Proceedings of the IEEE, 111(9) :1016–1054, 2023.

Table des figures

Liste des tableaux

FOLIO ADMINISTRATIF

THESE DE L'INSA LYON, MEMBRE DE L'UNIVERSITE DE LYON

NOM : **MARTHE** DATE de SOUTENANCE : **12/11/2024**
(avec précision du nom de jeune fille, le cas échéant)

Prénoms : **Guillaume, Sébastien**

TITRE : **Neurones à impulsion pour les communications sans fil**

NATURE : **Doctorat** Numéro d'ordre : **2024ISAL0094**

École Doctorale : **160 Électronique, Électrotechnique et Automatique de Lyon**

Spécialité : **Traitement du Signal et de l'Image**

RÉSUMÉ :
Dans le contexte de l'Internet des Objets, l'un des plus grands défis réside dans la gestion énergétique. Les radios à réveil (Wake-up Radio) permettent aux dispositifs de rester en veille tout en consommant très peu d'énergie, se réveillant uniquement lors de la réception de signaux spécifiques. Dans cette thèse, nous proposons d'utiliser les réseaux de neurones à impulsions (SNNs) comme WuR. Le rôle du réseau de neurones sera de reconnaître la séquence d'activation du noeud concerné dans un flux de bits, afin de le réveiller. Nous présentons tout d'abord les hypothèses et modèles de neurones, de réseau et de signaux utilisés pour notre étude. La première contribution est de montrer la pertinence de ces réseaux. Notre seconde contribution a été l'étude et la proposition du modèle Saturating Leaky Integrate and Fire pour le conception d'une WuR. Dans cette partie, nous proposons d'utiliser un phénomène bio-inspiré appelé Interaction Synaptique afin de produire un filtre temporel dépendant de l'Inter-Spike Timing. Nous étudions les paramètres de ce modèle afin de comprendre comment adapter cette plage d'Inter-Spike Timings. L'originalité de cette contribution est de proposer un nouveau moyen de reconnaître dans le domaine de l'analogique des séquences temporelles. Par la suite, différentes topologies de réseaux de neurones SLIF ont été explorées, notamment une topologie en ligne, en losange et réseau multi-couches, afin de comprendre comment le réseau répond aux séquences d'impulsions. Cette thèse établit ainsi les bases pour des recherches futures sur l'utilisation des réseaux neuromorphiques dans les dispositifs IoT à faible consommation d'énergie, notamment dans les WuRs. Les travaux accomplis ouvrent la voie à la conception de réseaux de neurones capables de traiter des signaux temporels complexes tout en maximisant l'efficacité énergétique, répondant ainsi aux exigences des applications IoT.

MOTS-CLÉS : **Réseau de neurones à impulsion, neurone à impulsion, Wake-Up Radios, détection de séquence d'activation, saturation synaptique, interaction synaptique, système faible puissance, intégration temporelle**

Laboratoire(s) de recherche : **CITI Lab**

Directeur de thèse : **Claire GOURSAUD**

Président du Jury : **Bernard GIRAU**

Composition du Jury :
Bernard GIRAU, Anne JULIEN-VERGONJANNE, Jean MARTINET, Céline TEULIERE