PROJET SCIENTIFIQUE COLLECTIF

Rapport Intermédiaire

EPIGÉNÈSE SYNAPTIQUE DANS L'ESPACE DE TRAVAIL NEURONAL GLOBAL

Groupe BIO 001

Anne-Sophie MIGEON
Camille DEMARRE
Gauthier GUINET
Jean-Stanislas DENAIN
Paul JACOB
Valentin VILLECROZE

Tuteur
Guillaume DUMAS
Coordinateur
Yves MECHULAM

Table des matières

INTRODUCTION

L'objectif de ce PSC est de simuler des réseaux de neurones biologiquement plausibles et d'étudier leur performance sur diverses tâches d'apprentissage. L'exigence de réalisme biologique nous mène à choisir des modèles d'apprentissage issus des neurosciences computationnelles plus que ceux propres à l'intelligence artificielle. Par conséquent, il n'y a pas de rétropropagation dans nos réseaux : nous nous sommes appuyé sur des modèles de plasticité synaptique. L'évolution du poids des connexions entre neurones au cours de l'apprentissage structure peu à peu le réseau jusqu'à lui permettre de résoudre certaines tâches.

Le travail que nous avons effectué se divise en deux parties :

— Un **travail de recherche bibliographique**, pour mieux comprendre les différents modèles mathématiques de plasticité synaptique et les mécanismes biologiques qui sous-tendent les phénomènes qu'ils décrivent.

— Un **travail informatique de simulation** et d'optimisation des modèles et de leur performance sur diverses tâches. Notre principal objectif de simulation est le calcul d'une **fonction XOR**, exemple canonique dans la littérature : c'est une tâche classique mais non triviale, une bonne preuve du bien fondé d'un modèle.

Dans ce rapport nous décrivons tout d'abord la manière dont nous nous organisons afin de mener ce projet à bien, puis nous présentons les avancées que nous avons effectuées jusqu'à présent dans le travail bibliographique comme dans les simulations informatiques.

1 ORGANISATION DU TRAVAIL

1.1 Répartition des rôles et planning

Notre travail se décompose en deux grandes parties : un état de l'art très approfondi sur les connaissances actuelles des phénomènes d'apprentissage du cerveau humain et un travail de simulation informatique de ces mécanismes d'apprentissage. Nous nous sommes donc réparti le travail selon ces différentes tâches : Camille et Anne-Sophie se chargent du travail de recherche bibliographique et d'état de l'art (chacune se spécialisant plus particulièrement sur un type d'apprentissage par renforcement ou de type hebbien) ; Jean-Stanislas et Paul se chargent de la simulation informatique des synapses selon le modèle proposé par Izhikevich [1] ; Valentin et Gauthier se chargent d'implémenter et d'étudier le test du XOR.

Cette répartition est celle que nous avions prévue au départ. L'avancée dans notre projet nous a permis de la revoir. En effet, le test du XOR n'intervient que dans un second temps puisqu'il faut avant cela proposer une architecture fonctionnelle de synapses. Nous avons donc constitué un unique groupe pour les simulations informatiques qui a avancé sur la mise en œuvre de l'architecture synaptique proposée par Izhikevich nécessaire à l'implémentation ultérieure du test de la fonction XOR.

Pour résumer notre planning dans les grandes lignes, nous avons utilisé le mois de septembre pour cerner plus précisément notre sujet en mettant en évidence l'importance des deux types d'apprentissage, et nous avons commencé à nous familiariser avec les outils que nous allions utiliser (Brian notamment). Les premières simulations effectuées à partir du papier publié par Eugène Izhikevich ont été intégrées dans le travail de notre tuteur et présentées lors d'un Symposium de l'Institut Pasteur, ce que nous détaillons plus loin. Nous avons ensuite eu une réunion de coordination en présence de Yves Mechulam et Guillaume Dumas le 24 octobre, ce qui nous a permis de faire le point sur la marche à suivre après le symposium : continuer la recherche bibliographique, et améliorer les simulations informatiques.

1.2 GitHub

Nous avons utilisé l'outil *GitHub* pour favoriser l'interaction du groupe en ce qui concerne les simulations informatiques. En effet, il s'agit d'une plateforme sur laquelle nous pouvons partager nos codes informatiques et travailler de manière coordonnée sur un même pro-

gramme informatique.

1.3 Poster

Nous avions comme échéance intermédiaire de participer au Symposium de l'Institut Pasteur " *Neural networks - from brains to machines and vice versa*" qui s'est déroulé les 11 et 12 octobre. Ce fut pour nous une belle opportunité pour présenter notre projet, ainsi qu'une motivation supplémentaire pour progresser rapidement. Pour l'occasion nous avions donc préparé un panneau (Annexe B - Fig.11) dans lequel nous avions inséré nos premiers résultats ainsi que l'objectif de notre travail. La présentation de ce panneau nous a permis d'échanger sur le sujet avec des chercheurs exerçant dans ces domaines. Ce contact privilégié avec le monde de la recherche fut très instructif : en effet, l'esprit de collaboration ainsi que l'émulation positive nous permit d'avoir de nombreux avis et conseils constructifs pour mieux progresser par la suite. De plus, il y avait de nombreuses conférences pendant le Symposium. Elles furent l'occasion de découvrir et approfondir des aspects du fonctionnement des réseaux de neurones que nous n'avions pas encore eu le temps d'étudier à ce stade de nos recherches bibliographiques. Par ailleurs, les techniques employées sont souvent récurrentes : ainsi nous avons pu retrouver l'usage de la porte logique XOR lors d'autres travaux de recherches comme test permettant de valider l'efficacité du réseau. Ce fut donc une belle opportunité pour retirer des éclairages pour le travail de recherche bibliographique qui a suivi.

2 TRAVAIL DE BIBLIOGRAPHIE

Un des premiers aspects importants dans notre projet est de bien définir le cadre d'étude et comprendre ce qui a été fait pour avancer. Nous avons donc eu une partie bibliographie, ou état de l'art, assez importante au vu du nombre de papiers publiés en rapport avec notre objectif. Nécessaire pour bien comprendre ce qui est mis en jeu, elle a mobilisé une grande partie de notre recherche. Nous nous sommes particulièrement intéressés dans un premier temps aux deux types d'apprentissage du cerveau humain : **l'apprentissage par renforcement** (reinforcement learning) et **l'apprentissage de type hebbien** (hebbian learning). Ce travail de recherche bibliographique nous permettra d'intégrer les deux types d'apprentissage dans nos simulations.

2.1 Généralités sur le fonctionnement des synapses

Les neurones sont liées par des synapses : plus précisément, la synapse est le lieu où l'axone d'un neurone (dit présynaptique) rencontre la dendrite d'un second neurone le plus souvent distinct (neurone post synaptique). Lorsque le neurone présynaptique émet un potentiel d'action, ce signal est transmis au neurone postsynaptique via la synapse. Nous nous restreindrons ici à l'étude des synapses chimiques (les *gap junctions*, synapses électriques, sont moins fréquentes). Chez ces dernières, le signal électrique présynaptique mène à la libération de *neurotransmetteurs* dans l'espace séparant les deux neurones. Ces molécules induisent une cascade de signalisation dans la membrane postsynaptique, qui mène à la dépolarisation du second neurone : on parle de potentiel postsynaptique (cf image ci-dessous, issue de [2]).

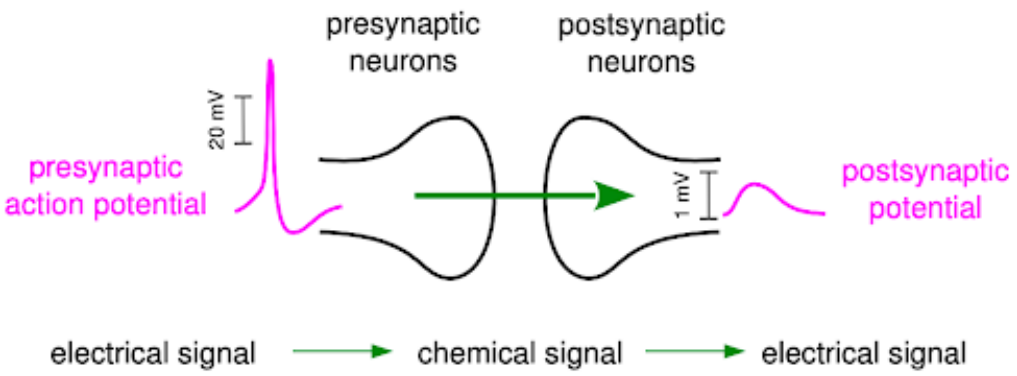

FIGURE 1 – Fonctionnement d'une synapse chimique (image issue de [2])

L'intensité du potentiel synaptique en réponse à un potentiel d'action du neurone post-synaptique varie parmi les synapses : on parle de *poids synaptique*, bien qu'il n'y ait pas de consensus autour d'une définition rigoureuse de cette notion. Le poids synaptique peut évoluer au cours du temps, et ce selon plusieurs mécanismes. Nous nous intéresserons ici à la plasticité de long terme, c'est à dire aux changements du poids synaptique qui demeurent après plusieurs jours. La potentiation (resp. dépression) de long terme LTP (resp. LTD) désigne l'augmentation (resp. diminution) durable du poids synaptique, le plus souvent chez les synapses excitatrices. LTP et LTD sous-tendent de nombreux processus cognitifs de mémorisation et d'apprentissage. Ces phénomènes sont des exemples de ce qu'on appelle l'apprentissage hebbien.

2.2 Apprentissage Hebbien (Hebbian learning)

On appelle apprentissage hebbien le renforcement des synapses dans le cerveau selon la formule de Hebb ([3]) : *« cells that fire together, wire together »*. L'idée est que des neurones dont l'activité est simultanée ou quasi-simultanée voient la synapse qui les relient se renforcer. Inversement, le poids d'une synapse reliant deux neurones dont les décharges nerveuses sont asynchrones aura tendance à diminuer. Intuitivement, les deux neurones dont l'activité est proche encodent des phénomènes corrélés, et l'augmentation du poids synaptique constitue l'apprentissage de cette corrélation. L'apprentissage hebbien permet donc d'encoder les régularité statistiques dans l'environnement.

La LTP a été mise en évidence pour la première fois par Bliss et Lømo en 1973 ([4]), dans l'Hippocampe des lapins. En 1997, Markram et collaborateurs ([5]) ont introduit la notion de

STDP (Spike Time Dependent Plasticity). En contrôlant précisément le temps séparant les potentiels d'actions de deux neurones, ils ont observé que le renforcement de la synapse varie selon le temps qui sépare les impulsions des deux neurones. Plus précisément : la synapse reliant le neurone 1 au neurone 2 verra son poids augmenter si le neurone 1 spike juste avant (tau < 10 ms) le neurone 2, et diminuer si le neurone 2 spike juste avant le neurone 1(cf image ci-dessous, également issue de [2]).

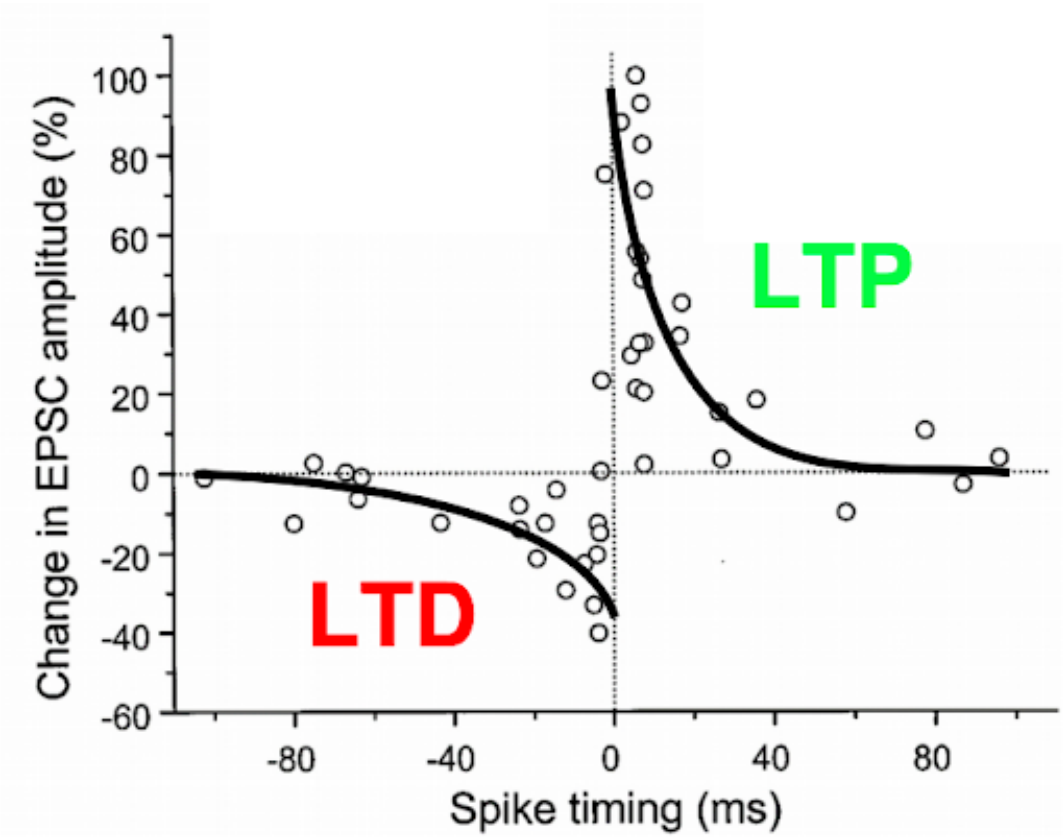

FIGURE 2 – Renforcement du poids synaptique en fonction de la durée séparant les impulsion présynaptiques et postsynaptiques (image issue de [2])

Ces résultats sont expérimentaux, mais notre travail bibliographique a aussi porté sur les mécanismes biologiques sous-jacents en nous basant sur différentes sources ([6], vidéos 2, 3 et 5 de [7]).

L'apprentissage hebbien a donc des conséquences sur l'architecture du cerveau. Nous nous sommes intéressés à l'évolution de l'architecture neuronale suite au phénomène de renforcement, comme l'expliquent notamment Lindsay et Fusi dans leur papier de novembre 2017 [8] sous l'expression *"rich get richer"* détaillant l'effet amplificateur du renforcement des

synapses selon l'apprentissage de type hebbien. A n de pouvoir simuler ce type de plasticité, nous nous sommes également intéressés aux modèles mathématiques de la STDP proposés par Markram, Izhikevich et Gerstner ([2], [9], [10]). La principale source que nous avons utilisé dans ce but est l'article Scholarpedia écrit par Wulfram Gerstner et Jesper Sjöström ([10]).

2.3 Apprentissage par renforcement (Reinforcement learning)

L'apprentissage par renforcement consiste à privilégier les actions qui procurent la plus grande récompense possible. Au contact d'un environnement donné, un individu peut apprendre de diverses manières, et notamment en tâtonnant, c'est-à-dire en essayant une action, en faisant des erreurs et en apprenant de ces erreurs. Ce type d'apprentissage est très utilisé en intelligence arti cielle, et on peut marquer les débuts algorithmiques du reinforcement learning avec le « Temporal Difference Learning algorithm » de Richard Sutton en 1988. La rencontre entre neurobiologistes et chercheurs en intelligence arti cielle fait ensuite ressortir les similarités existant entre les types d'apprentissage des modèles computationnels et le fonctionnement du cerveau humain. Différent de l'apprentissage supervisé et de l'apprentissage non supervisé, tous deux utilisés en machine learning, l'apprentissage par renforcement repose sur deux grands principes : la recherche par trial-and-error (essais et erreurs) et la notion de delayed reward (récompense retardée). Cette dernière notion implique que le choix fait d'une action n'impacte pas seulement la récompense immédiate mais aussi d'autres futures récompenses étant donné les états suivants.

En intelligence arti cielle, on dé nit un cadre théorique pour l'apprentissage par renforcement, comptant un agent, un environnement avec éventuellement un modèle de cet environnement, des états caractérisant la relation entre l'agent et l'environnement, un signal de récompense selon l'état, plusieurs stratégies différentes pour l'agent, et une valeur pour les états.

On distingue ainsi généralement la valeur et la récompense due à un état. Alors que la récompense est immédiatement perçue lorsque l'agent arrive sur un nouvel état, sa valeur est différente : elle caractérise les récompenses possible en partant de cet état sur le long terme. Elle est donc plus beaucoup plus dif cile et donc centrale dans le reinforcement learning . Ainsi, la majeure partie des modèles computationnels traite de la manière d'approximer cette valeur.

FIGURE 3 – Paradigme général de l'apprentissage par renforcement (image issue de [14])

On peut nalement se demander à quoi correspondent les éléments indiqués plus hauts (agent, l'environnement) dans le contexte des neurosciences. Cloisonner très précisément l'agent et l'environnement peut s'avérer dif cile selon l'échelle qui est étudiée [11].

Dans notre travail, on peut considérer que l'agent correspond à l'ensemble des neurones, l'environnement aux signaux qui lui sont envoyés, et la récompense à la dopamine. Cette récompense est plus particulièrement le résultat d'une comparaison entre ce que le cerveau prédit de l'environnement et ce qu'il perçoit effectivement, selon l'hypothèse du codage prédictif, notamment dans le réseau du mode par défaut, qui correspond aux régions cérébrales actives lorsque le réseau est au repos, mais actif [12]. L'importance de la dopamine se manifeste notamment dans des maladies présentant des troubles liés à la dopamine : maladie de Parkinson, syndrome de la Tourette, schizophrénie [6]. Il est donc normal que la dopamine joue un rôle central dans notre simulation.

Du point de vue expérimental, l'in uence de la quantité de dopamine sur la sensibilité de la potentiation STDP a été étudiée par Zhang et ses collaborateurs [15]. En général la présence de dopamine dans l'environnement renforce la LTP et permet un renforcement synaptique pour des impulsions moins proches dans le temps.

Eugene Izhikevich a proposé un modèle de plasticité synaptique mêlant STDP et modulation par la dopamine ([1]), à la frontière entre reinforcement learning et hebbian learning . C'est ce modèle que nous avons étudié plus en détail, et que nous avons utilisé dans nos simulations.

2.4 Modèle de Izhikevich

Dans Solving the distal reward problem through linkage of STDP and dopamine signaling (2007), Eugene Izhikevich propose un modèle de plasticité mêlant apprentissage hebbien et apprentissage par renforcement. L'idée générale est qu'une récompense sous forme d'un in ux de dopamine puisse renforcer l'effet de l'apprentissage hebbien décrit par le modèle STDP.

Plus précisément, si le neurone 1 a une impulsion juste avant le neurone 2, l'augmentation du poids de la synapse entre le neurone 1 et le neurone 2 :`

— est une fonction décroissante de la durée qui sépare les deux impulsions (c'est ce que prévoit le modèle STDP).
— est une fonction croissante de la concentration en dopamine dans le milieu.

Ce modèle est particulièrement intéressant dans la mesure où il est très robuste :
— le réseau neuronal est capable de déterminer quels sont les neurones présentant des motifs d'activité responsables des récompenses, même quand de nombreux autres neurones “distracteurs” sont actifs dans la durée séparant le stimulus de la récompense.
— le relâchement global de dopamine, bien que non ciblé, ne renforce que les synapses qui sont régulièrement associées à une récompense, et n'a en moyenne pas d'effet sur les autres synapses.

Dans son article, Izhikevich propose une modélisation mathématique plus rigoureuse de son modèle, en introduisant des variables auxiliaires et en précisant les équations différentielles suivies par les différents paramètres. C'est cette modélisation que nous avons utilisée au cours de nos simulations, et une part importante de notre travail a été de répliquer les résultats obtenus par Izhikevich dans un langage et avec un formalisme différent.

3 SIMULATIONS BRIAN

3.1 Cadre général

Pour nos simulations nous avons souhaité adopter un point de vue qui n'est pas celui de l'intelligence arti cielle. En effet, il s'agit pour nous de simuler le comportement d'un ensemble de neurones reliés par des synapses, de telle sorte que les variables d'état des neurones et synapses soient régies par les équations d'un modèle donné, en l'occurrence celui d'Izhikevich (2007) ([1])

Pour cela nous utilisons le logiciel Brian2 , auquel on accède via Python, dont nous avons utilisé notamment les fonctions suivantes :

- les instances de la classe NeuronGroup sont des groupes de neurones dont on précise les équations différentielles qui régissent l'évolution des différents paramètres (eg potentiel de membrane, conductance des canaux ioniques).
- les instances de la classe Synapse sont des synapses entre neurones : on précise suivant le modèle de synapse employé l'in uence d'une activité présynaptique sur le potentiel post synaptique.

Nous avons choisi d'utiliser le logiciel Brian2 puisque son interface de programmation a l'avantage d'ajouter une couche d'abstraction et donc de garantir une certaine plausibilité biologique par rapport à des bibliothèques purement mathématiques (eg Numpy, MATLAB). Elle est de plus optimisée pour des simulations de réseaux de neurones. Cependant, à cause de cette abstraction il est plus dif cile de déboguer les codes.

Une simulation en Brian2 est généralement de la forme suivante (celle que nous avons choisie de prendre pour nos simulations) :

- Dé nition de la valeur des paramètres qui resteront constants au cours de la simulation : temps de simulation, constantes de temps des équations différentielles, potentiels ioniques d'équilibre, densité synaptique, etc. (cf Annexe A.1)
- Création et initialisation de groupes de neurones et de synapses en les reliant, en précisant les équations différentielles suivies par les paramètres de ces neurones et synapses. (cf Annexe A.2)
- Création d'objets Brian2 a n d'enregistrer et de suivre l'activité de certains neurones

au cours de la simulation : il s'agit des SpikeMonitor et StateMonitor.

— Lancement de la simulation et af chage de graphiques permettant de visualiser l'évolution du réseau.

3.2 Simulations effectuées et voies d'amélioration envisagées

Renforcement de synapse

Notre première véritable simulation avait pour objectif de mettre en évidence le renforcement d'une synapse au sein d'un réseau de neurones. Pour cela, nous choisissons au préalable une synapse dans le réseau (celle qu'on souhaite renforcer), et on initialise son poids à 0. Les neurones du réseau sont choisis de telle sorte qu'ils émettent des potentiels d'action de manière aléatoire avec une fréquence de 1Hz. Lorsqu'une impulsion du neurone postsynaptique a lieu dans un intervalle de 10ms après un potentiel d'action présynaptique, de la dopamine est relâchée dans l'ensemble du réseau avec un délai de l'ordre de la seconde. On s'intéresse au poids de la synapse choisie.

L'idée est la suivante : l'avènement de spikes quasi simultanés décrit ci-dessus n'aura initialement lieu que rarement (environ une fois toutes les quelques minutes étant donnée la fréquence d'impulsion). Cependant, quand il advient, le poids de la synapse croît du fait de la libération de dopamine. Cette augmentation augmente la probabilité qu'une impulsion présynaptique donne lieu à un potentiel d'action du neurone postsynaptique : le poids de la synapse devrait donc monter au cours du temps. Comme seule la synapse d'intérêt est associée avec une telle régularité à la libération de dopamine, son poids augmente plus que celui des autres synapses. Cette simulation a donc permis de renforcer la synapse choisie au départ.

Le schéma de l'Annexe B - Fig.5 est issu de l'article d'Izhikevich : on observe bien que le poids de la synapse renforcée augmente au cours de la simulation, jusqu'à dépasse de loin les autres poids synaptiques.

Finalement, nous avons réussi à répliquer ce résultat d'Izhikevich en Brian2 . Le schéma de l'Annexe B - Fig.7 montre l'évolution du poids de notre synapse et un graphique de l'évolution des paramètres de la synapse d'intérêt. L'Annexe B - Fig.6 montre l'histogramme de la distribution des poids des différentes synapses à la n de la simulation. On observe bien que la synapse 0 est renforcée, et que les autres synapses ne le sont pas, ou très peu (synapse 0.3).

Conditionnement

Notre seconde tâche avait pour objectif d'observer des phénomènes de conditionnement : on considère un réseau composé de n groupes contenant chacun m neurones. On appelle S1 le premier groupe. Tous les neurones ont une activité stochastique, avec une impulsion toutes les secondes environ. À ce bruit s'ajoute une activité supplémentaire : tous les cinquièmes de seconde environ, on choisit un des n groupes, et tous les neurones de ce groupe émettent un potentiel d'action. L'idée est que chaque groupe de neurone encode un stimulus différent. Le stimulus associé au groupe S1 est le stimulus conditionnant : il donne lieu à une libération de dopamine dans l'ensemble du réseau.

Le principe de la simulation est le suivant : le mécanisme de STDP couplé à la libération de dopamine aura tendance à renforcer le poids des synapses reliant les neurones de S1 aux autres neurones du réseau. Par conséquent, au bout d'un certain temps, le stimulus conditionnant permettra à l'ensemble du réseau d'accroître son activité : selon l'analogie utilisée par Izhikevich, les neurones du réseau “écouteront plus attentivement” les neurones du groupe S1. On af che en n le raster plot de la simulation, ie un graphique qui montre quand chaque neurone a un potentiel d'action. Dans le schéma de l'Annexe B - Fig.8, issu de Izhikevich (2007), on observe bien l'effet escompté.

Nous avons programmé en Brian2 le réseau correspondant. Les simulations effectuées jusqu'à présent ne répliquent pas le résultat obtenu par Izhikevich. Nous voyons plusieurs voies d'améliorations. Nous avons tout d'abord véri é la plausibilité des valeurs numériques des paramètres de notre simulation. Nous envisageons une autre hypothèse, que nous sommes en train de tester : la constante de temps de la libération de dopamine peut être trop élevée. Si la dopamine est relâchée trop lentement, la récompense n'est plus spéci que à un certain événement, mais impacte plein d'événements, et l'apprentissage n'a pas lieu. Il est également possible que nos ordinateurs n'aient pas la puissance nécessaire pour pousser la simulation assez loin pour pouvoir observer l'effet attendu. Notre objectif est de lancer les simulations sur le cluster de l'Institut Pasteur, a n de pouvoir mener les calculs en parallèle et gagner en puissance. Notre raster plot est à l'Annexe B - Fig.9.

Calcul d'une fonction XOR

XOR est une fonction $\{0,1\}$ £ $\{0,1\}$! $\{0,1\}$ dé nie par la table de vérité suivante :

FIGURE 4 – Table de valeurs de la fonction XOR (image issue de [13])

Il s'agit ici d'apprendre à calculer la fonction XOR . Concrètement, l'architecture que nous avons choisie est la suivante (voir Annexe B - Fig.10). Les deux entrées ainsi que la sortie sont des groupes de deux neurones, un pour 0 et un pour 1. Ces trois groupes sont tous connectés à un groupe central. Les neurones de ce groupe central sont connectés par des synapses dont la plasticité est régie par le modèle de Izhikevich. Lorsque le réseau renvoie la bonne réponse, c'est à dire le XOR des deux entrées, de la dopamine est relâchée. Le XOR est une tâche classique en intelligence arti cielle, “la tâche la plus simple parmi les tâches non triviales”. [2]

Le principe est similaire à celui de l'expérience précédente, mais plus complexe. En effet, le groupe de neurone favorisé varie selon le contexte, selon la valeur des entrées. Nous avons implémenté le réseau mais les résultats ne sont pas encore probants. En en parlant avec notre tuteur nous avons décidé que la priorité était d'aboutir à une réplication satisfaisante de la tâche de conditionnement, qui constitue une brique élémentaire nécessaire pour l'apprentissage du XOR. Nous avons donc choisi dans un premier temps de concentrer notre ré exion sur la simulation du conditionnement pour ensuite nous intéresser plus en profondeur au test du XOR.

RÉFÉRENCES

[1] Solving the Distal Reward Problem through linkage of STDP and Dopamine Signaling , Eugene Izhikevich, 2018

[2] Synaptic Plasticity : Spike-timing dependent plasticity (STDP) , a course by Michael Gaupner

[3] The Organisation of behaviour , Donald Hebb, 1949

[4] Long-lasting potentiation of synaptic transmission in the dentate area of the anaesthetized rabbit following stimulation of the perforant path , Bliss and Lømo, 1973

[5] Regulation of Synaptic Ef cacy by Coincidence of Postsynaptic APs and EPSPs , Markram et al. , 1997

[6] From reinforcement learning models to psychiatric and neurological disorders , Maia and Frank, 2011

[7] Neuroscience: Exploring the Brain , Bear, Connors and Paradiso, éd Lippincott, Williams and Wilkins

[8] Hebbian Learning in a Random Network Captures Selectivity Properties of the Prefrontal Cortex , Grace W.Lindsay et al., 2017

[9] Neuronal Dynamics , Wulfram Gerstner et al., CUP, 2014

[10] Spike Timing dependent plasticity , Scholarpedia article, Sjöström and Gerstner

[11] Cellular mechanisms of brain function , a course by Carl Petersen

[12] Dark Control: A Uni ed Account of Default Mode Function by Control Theory and Reinforcement Learning , Bzdok, Dumas and Dohmatob, 2017

[13] The revenge of Perceptron!Learning XOR with TensorFlow , Claude Coulombe, 2017

[14] Reinforcement learning , Sutton and Barto, Second Edition, MIT Press, Cambridge 2018

[15] Gain in sensitivity and loss in temporal contrast of STDP by dopaminergic modulation at hippocampal synapses , Zhang et al., 2009

A EXTRAITS DU CODE

Ci-dessous des extraits de notre code pour la première tâche d'intérêt : le renforcement d'une synapse. Le reste du code est disponible sur le GitHub du PSC.

A.1 Initialisation des paramètres qui resteront constants pendant la simulation

```
# Parameters
simulation_duration = 10* second

## Neurons
taum = 10*ms
Ee = 0*mV
vt = -54*mV
vr = -60*mV
El = -74*mV
taue = 5*ms

## STDP
taupre = 20*ms
taupost = taupre
gmax = .01
dApre = .01
dApost = -dApre * taupre / taupost * 1.05
dApost *= gmax
dApre *= gmax

## Dopamine signaling
tauc = 1000*ms
taud = 200*ms
taus = 1*ms
epsilon_dopa = 5e-3
```

A.2 Création de groupes de neurones et synapses

On ne montre ici qu'un extrait, la création des synapses dont la plasticité est régie par le modèle d'Izhikevich. On remarquera que les équations différentielles sont codées telles quelles dans le programme.

```
## STDP section
epsilon = 0.1 # sparseness of synaptic connections

synapse_stdp = Synapses(neurons, neurons,
                  model='''mode: 1
                        dc/dt = -c / tauc : 1(clock-driven)
                        dd/dt = -d / taud : 1(clock-driven)
                        ds/dt = mode * c * d / taus : 1(clock-driven)
                        dApre/dt = -Apre / taupre : 1(event-driven)
                        dApost/dt = -Apost / taupost : 1(event-driven)''',
                  on_pre='''ge += s
                         Apre += dApre
                         c = clip(c + mode * Apost, -gmax, gmax)
                         s = clip(s + (1-mode) * Apost, -gmax, gmax)
                         ''',
                  on_post='''Apost += dApost
                         c = clip(c + mode * Apre, -gmax, gmax)
                         s = clip(s + (1-mode) * Apre, -gmax, gmax)
                         ''',
                  method='euler'
                  )
synapse_stdp.connect(condition = 'i!=j', p=epsilon)
if not((0, 1) in zip(synapse_stdp.i, synapse_stdp.j)):
    synapse_stdp.connect(i=0, j=1)
```

B RÉSULTATS ET IMAGES

B.1 Renforcement d'une synapse

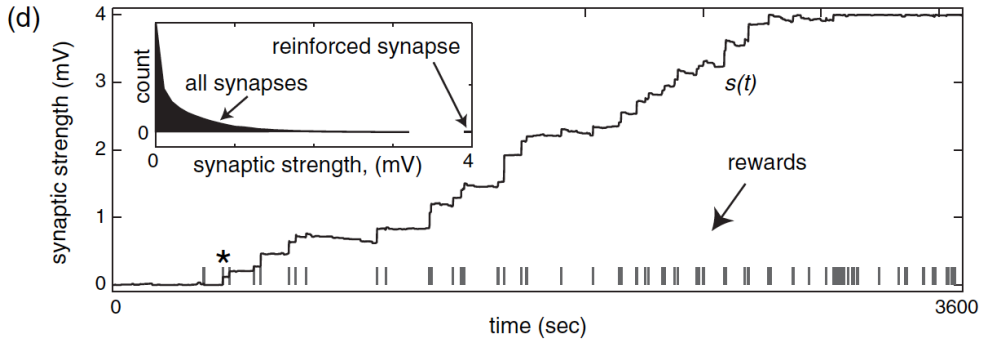

FIGURE 5 – Renforcement du poids synaptique : résultats d'Izhikevich [1].

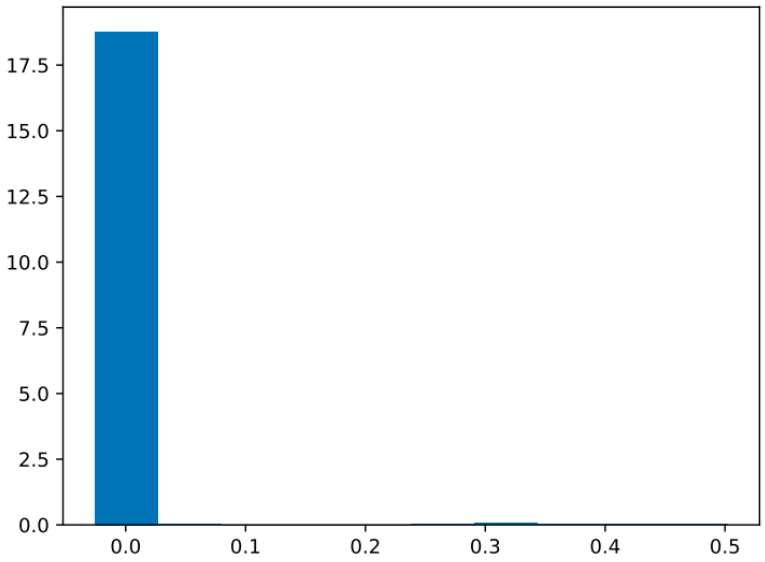

FIGURE 6 – Renforcement du poids synaptique : histogramme des poids des différentes synapses. On aperçoit à peine la synapse renforcée en queue de distribution.

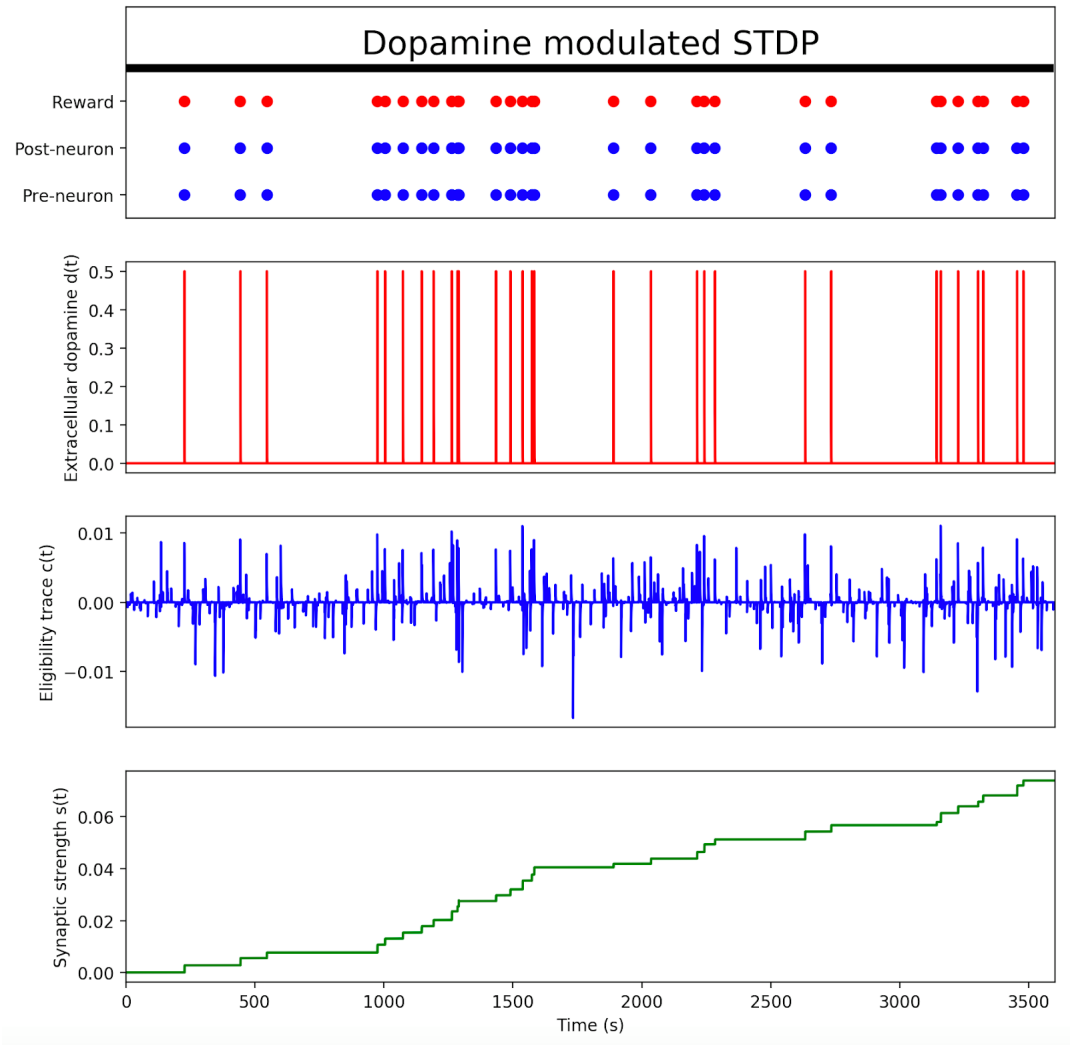

FIGURE 7 – Renforcement du poids synaptique : évolution des paramètres de notre simulation. On observe l'augmentation progressive du poids de la synapse renforcée.

B.2 Conditionnement

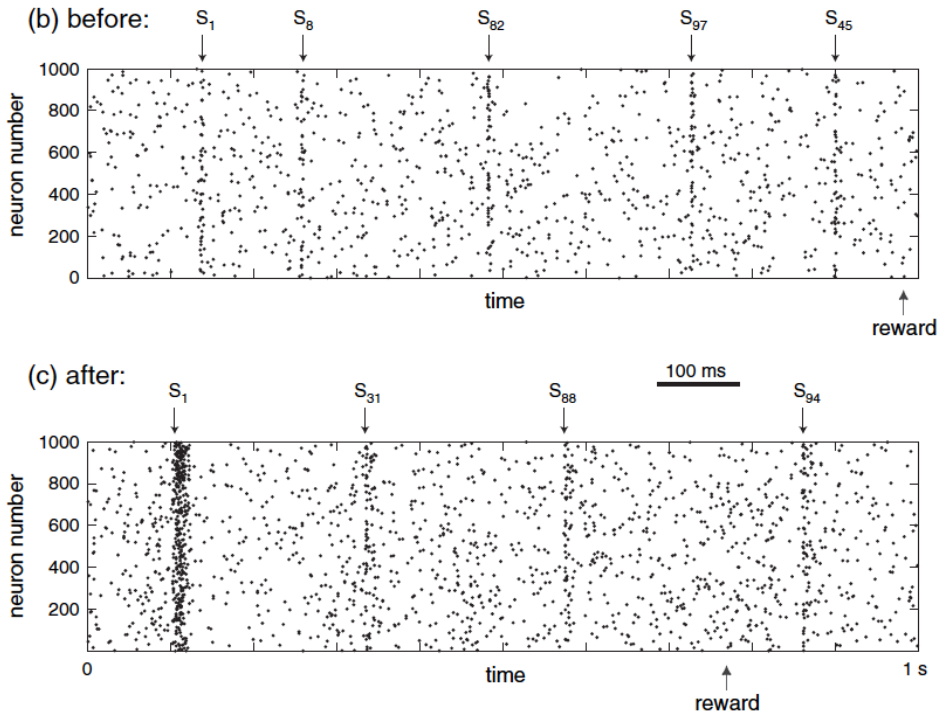

FIGURE 8 – *Raster plot* d'Izhikevich [1] pour la tâche de conditionnement, au début et en fin de simulation.

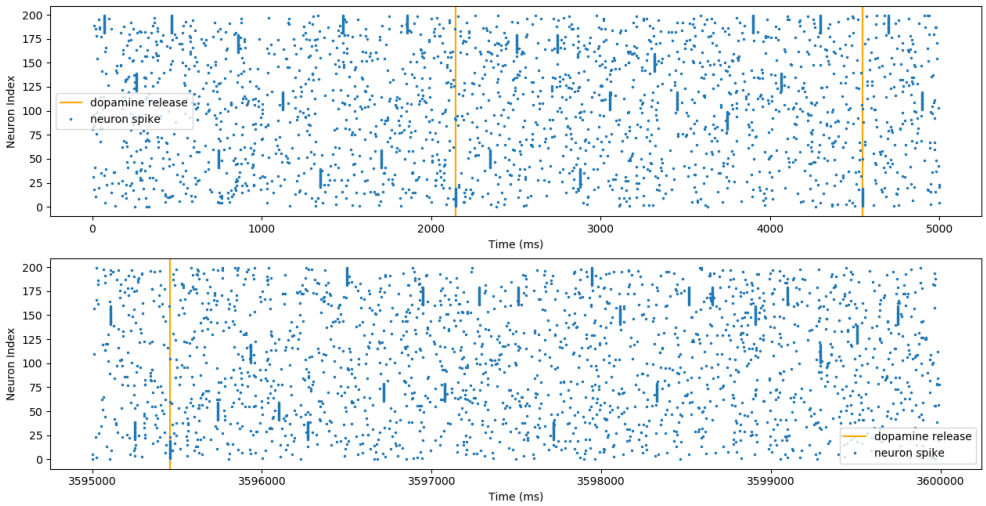

FIGURE 9 – Notre *Raster plot* pour la tâche de conditionnement.

B.3 Calcul du XOR

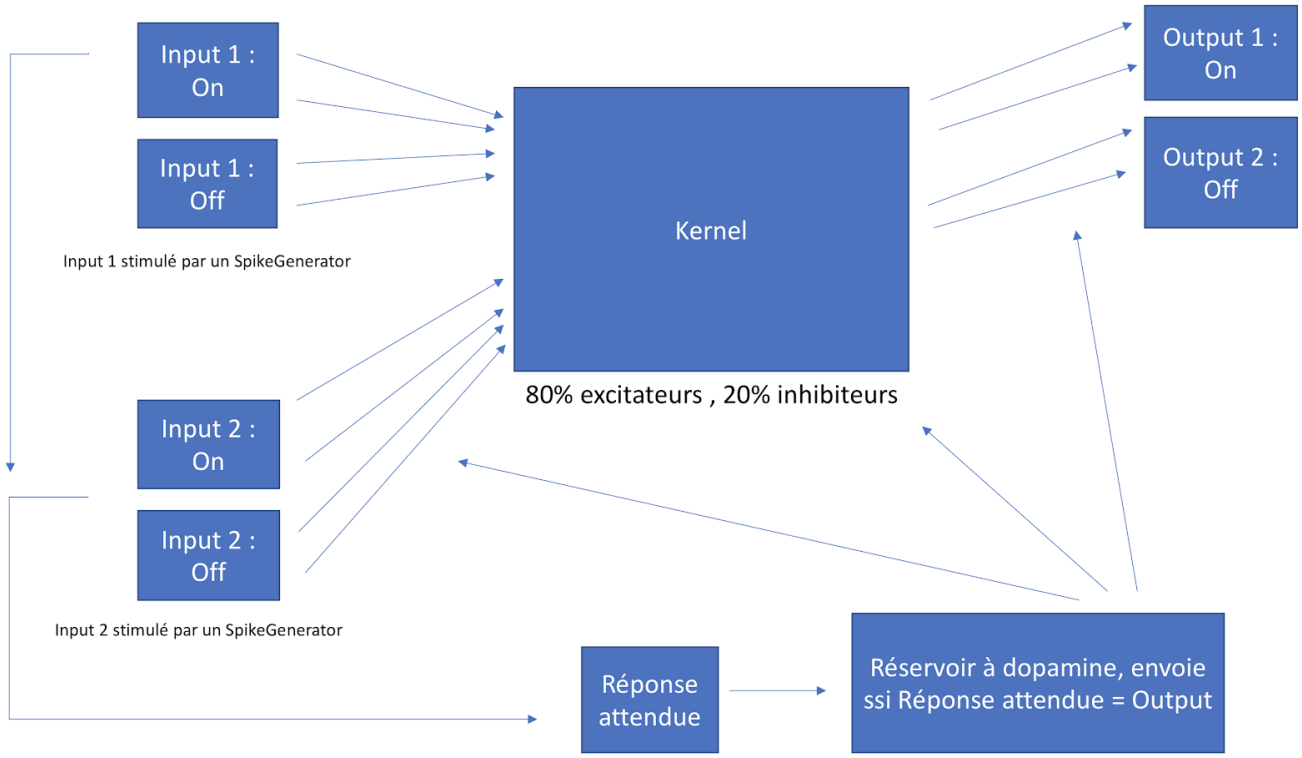

FIGURE 10 – Architecture de notre simulation pour l'apprentissage du XOR.