Oui Bonjour 15 personnes encore je crois sur ce week-end. Mais on peut commencer tranquille et on commencera, on attaquera dans le gros du cours un peu plus tard et déjà bonne année. Ça va ça été les vacances. Ouais, c'est un peu. C'est un peu chiant qu'on ait ce format là parce que on a coupé un peu cette fois, ça fait 3 semaines. Du coup c'est. Ouais mais pour du coup on va essayer de faire vite fait entre nous qui un petit blocage. Redéfinir quelques trucs et faire en sorte que on soit OK pour pour partir sur la suite, soit en gros aujourd'hui on va. On va passer au-delà du de ce qu'on a fait pour l'instant. C'est dire ce que j'ai appelé Tabular RL jusqu'ici on retenait tout le temps, donc on stockait tout dans des énormes tables. Sauf que dans la vraie vie, bah ça ça marche pas. Et donc là on va passer sur des des modèles qui peuvent prendre en en entrée des États, des actions continues. Euh mais avant ça, en fait, je veux que ce soit bien clair, tout le monde. Du coup, on va commencer par. De. De ce qu'on a fait jusqu'à jusqu'à maintenant. Et pour ça, je vais. Je vais vous poser des petites questions et je vais vous demander tous de prendre un petit papier et de décrire chacun pour vous ou sur votre PC. Ça marche plus ? Je vous donne à chaque fois une trentaine de secondes pour réfléchir à la question et ensuite on répond ensemble et on voit si. Alors pour commencer, on va faire vraiment les. On vous parle de enfin vous Notez tout ce que vous que ce que je vous dis bien à l'esprit quand je vous parle de de MDP je suis volontairement très très flou mais MDP on va beaucoup parler au début et c'est un peu fondamental pour qu'on fait pour la théorie notamment la thématique derrière. Du coup pour MDP on a plein de. On a plein de choses qu'on a défini ensemble pendant les premiers cours notamment et. Et donc j'aimerais que vous notiez un peu là tout de suite. Bah comment est-ce que vous définiriez un des faits ? C'est quoi les quantités qui sont importantes ? À quoi est-ce qu'elles correspondent ? Et Bah comment est-ce que ? À quoi est-ce que vous parlez plus tard ? C'est le genre de question que je vais vous mettre là à au partiel France, parce que c'est ça qui est vraiment important quand on fait du RP, c'est savoir de quoi on parle après un milliard d'algorithmes, mais l'important c'est de comprendre ce que fait l'algorithme. Donc pour vous dire, la. Qu'est-ce que c'est ? Ok, parce que quelqu'un peut peut répondre à cette question. Oui, je préfère le tout seul, ouais. Ouais, qu'est-ce que c'est ? Qu'est-ce que c'est ? Donc se se doit sachant ce moi c'est ce que tu m'as dit mais du coup faut être plus précis. Et là, j'ai mis un. Qu'est-ce que c'est ? La probabilité de l'élément précédent en connaissant l'élément. Non ? Quand on les fait comme ça, c'est peut-être la policie. Ouais. C'est de transition. Ok, pas mal, c'est un des points plus importants dans dans 1DP. Un autre d'autres choses. Ouais. Excellent donc là on rajoute les les quantités. Constance, on a la récompense. On a donc qui appartient à toujours ensemble grand-père qui est défini de récompenses, un ensemble d'actions à. Et appartient à l'ensemble des actions possibles du grand A et t'as parlé d'autre chose comment ? Ouais et l'ensemble des États qu'on a, on a des ici. La suite des États s ? Qui appartient à l'ensemble de n'hésitez pas encore. Ce qui manque encore ? On a parlé de policier, on a dit que c'était proche de ça, mais pas exactement ça. C'est important, hein ? R, c'est un peu le truc que vous voulez apprendre la police. Donc si vous ne savez pas ce que ça, c'est compliqué quand même. Je vous aide. On la plaie. Pie dans dans nos cours précédents. C'est quoi la policier ? Ouais, je dirais que c'est un peu la manière dont on va s'y prendre pour. Parcourir, atteindre un but précis, ouais. Et concrètement, ça veut dire quoi ? C'est la manière de s'y prendre, c'est notre comportement, notre behavior. On pourra dire en anglais aussi et concrètement, on la définit comment cette politique ? Par nous, elle parle de quel ensemble elle va, dans quel ensemble et. Espérance maximale des gains, c'est une autre valeur qui est importante. Et ça s'appelle comment ? L'espérance maximale des gains. La. L'espérance de de bien pardon là je me suis trompé parce que j'ai oublié. L'espérance du bien sachant sachant pire non sachant l'État ouais sachant s donc là on a défini la value function OK j'ai été sachant pardon avait-il sachant s. Ouais donc V c'est une fonction et après en état notre étable, OK ? Merci maintenant. Et bon, on définira juste après. De prendre l'action par rapport à. Exactement et pourquoi est-ce que on appelle ça une police ou une politique ou une un comportement ? Ouais en gros c'est la police, c'est ce qu'on met dans le robot quoi. C'est un robot, on me donne une observation, on me donne un État dans lequel tu te trouves. Bah en fait ce qu'on attend du robot c'est de lui dire Bah en fait moi j'ai envie de faire ça quoi, j'ai envie de bouger mon bras, j'ai envie de d'appuyer sur une touche de mon clavier pour gagner le jeu. J'ai envie de faire XY, donc c'est ça la police ? La police ici, elle va de quel ensemble, à quel ensemble ? Sur un des films, mathématiquement. Ce qu'elle prend on ? Qu'est-ce qu'elle prend en entrée ? L'état ? Ouais. Et elle va dans quoi ? La récompense ? Ouais, elle va dans les actions. Ouais disons il y a plusieurs, il y a plusieurs manières de le faire, mais disons que dans les actions et donc à un État s, elle a ceci. L'action que je vais prendre, il y a une 2e version. Si on a une politique qui est stochastique. Ouais. Non, ce sera. 0 un de ? Ce sont des scènes, mathématiquement, c'est un peu plus chiant, mais dans l'idée. Dans l'idée, dans l'idée, c'est ça. Et elle associe à. Elle associe donc. Pour tout État va associer l'ensemble des probabilités de prendre l'action a sachant que je suis dans l'État est. Ou alors on la définit comme prenez en entrée l'État et l'action et en ordonnant. Dans l'idée, c'est ça, on est dans un État, on prend une action. Comment est-ce qu'on fait pour prendre une décision ? Quoi ? Donc on est exactement dans ce cas-là. Ok ouais, petit détail sur les sur les propriétés de transition, rigoureusement dans le MDP on a une on prend à la fois. Enfin on conditionne à la fois par l'État et par l'action. Vous avez déjà expliqué en pratique en R, on le fait pas beaucoup, mais parfois on peut prendre une décision. On peut prendre le chemin pour aller vers l'État s 2. Mais en fait l'action a 0A une probabilité non nulle de nous faire retourner à en 0 quand même. Donc il y a même si on a fait l'action, on n'est pas certain de de d'arriver sur l'État, donc c'est un juste une. Une variabilité au problème dans certains cas. Ok, c'est pas mal, c'est pas mal, c'est pas mal parce que il y a d'autres choses qui vous reviennent à l'esprit pour le cours un. Non, ouais. Ouais, c'est bien ce que ça a coûté de ma soeur. Je transition. C'est égal à. Ouais. C'est le. Probabilité de au moins un virgule. Exact. Ça, c'est clair pour tout le monde, pas ce qu'on fait quand on écrit ça. Tu m'as dit ? Encadré en rouge. C'est pas. Et pourquoi est-ce qu'on fait cette hypothèse là ? Par exemple, si vous avez un réseau de neurones, qui qui va, qui va, qui va modéliser votre modèle, si votre réseau de neurones il devait prendre en entrée tous les États qui précédaient, parce qu'en dans la vraie vie. La, l'action que vous faites maintenant, elle dépend quand même beaucoup de l'action que vous avez fait il y a un an, quand vous êtes inscrit à des frites par exemple. Donc vous pouvez pas dire vous pouvez pas dire que c'est indépendant alors que là c'est ce qu'on dit. On dit à l'instant T je m'en fous de tout ce qui s'est passé avant. Je me je m'occupe uniquement de l'État précédent. Donc. Et donc c'est une. C'est une très forte, c'est une très forte hypothèse. Donc ce qu'on suppose c'est que en fait toutes les, toutes les. Quand on voit toutes les informations, elles se propagent à travers les États. On a pas besoin d'aller chercher tous les États, les États, mais c'est c'est bien de garder à l'esprit que certes on fait ça tout le temps, mais c'est pas du tout évident et a priori c'est trop dans la. Ok. C'est pas mal. C'est pas mal, je vais juste vous remettre. Une slide. Hop là. Donc on se on se remet rapidement dans le. Alors un rapide résumé de donc on a vu le cas de mathématiques du MDP et maintenant en R on simplifie un peu les choses avec ces histoires de action environnement du vin et de et donc ça on a déjà défini jusqu'à ce qu'il y a une notion qui vous paraît pas clair là-dedans ou non. Très bien. Ça c'est très bien, on va avancer un peu quand même, on va pas refaire tout court. Alors ? Ensuite, on avait. On a vu policier value function. On a vu, on a pas fini le gain ici rapidement, ça c'est le gain. J'étais là, non ? Ouais, samedi. Exactement hop, j'ai perdu mon stylo. Ouais, le lien c'est la somme sur quoi ? Je me permet de mettre 1C à la place, c'était y a 0. C'est la somme, la somme pondérée de toute récompense qu'on arrive. Ok ? Bon, ça doit être clair pour tout le monde, hein. Si si vous êtes pas au cas-là dessus, vous allez avoir du mal dans la suite. Ensuite on a parlé de donc policier value function, ça c'est bon. On a réussi à programmer nos premiers algorithmes de R avec son appelle du Dynamic programming et ce que quelqu'un peut me rappeler un peu, c'est quoi le Dynamic programming et qu'est ce qu'on a fait ? Prenez une minute là pour faire la feuille blanche là donc vous écrivez sur un papier ou sur votre sur votre passé parce que vous vous rappelez de cette séance là et moi je vous redonne rapidement les points clés et les trucs qu'il faut vraiment retenir de ce de ce cours. Ok, est-ce que tu peux me faire un petit résumé du cours ? Programme. Déjà, c'est quoi le dernier programmé ? Programmation dynamique, oui. Pas du tout. Ben la programmation dynamique, c'est vraiment complètement décorée de ce qu'on fait en réel. C'est de base ça. C'est juste une manière de résoudre le problème. Ouais. On a tous les. Ouais. Et donc on va itérer sur chaque bêta alors que y a y a ça dans ce qu'on fait plus tard. Mais le le la programmation dynamique c'est beaucoup plus générale que ça. Vous devez me donner une définition de la programmation dynamique qui sans parler de RL, c'est ça que je disais ça c'est un paradigme très très particulier. Mais aux adultes déjà le voir vous avez beaucoup d'informatique normalement. Donc la programmation dynamique, ça doit vous parler dans tout ce qui est contenté de la ? On s'appelle de la Récursion et tout ça. On parle souvent de programmation dynamique. Vous avez déjà fait de la réception, non ? Ouais. Alors c'est quoi la formation dynamique ? Ouais exactement, c'est ça. Et donc maintenant. Donc je je répète, l'approbation dynamique, c'est les méthodes qui permettent de résoudre un problème complexe en le séparant en plein de petits problèmes. C'est le le principe du devis and Conquer donc. Pour résoudre un problème de RL où on a un un modèle qui est de sortir dans la terrasse et Ben on peut brancher ce problème là c'est très gros problème là qui est en fait quand même très difficile, c'est pas évident. Même les humains ont du mal alors qu'ils voient tout va bien. Eh Ben on peut résoudre ce problème là en faisant le programme dynamique, c'est-à-dire en séparant notre gros problème en plein de petits problèmes de de, de de styles. Bah je suis dans les cases 0 3 où est-ce que je dois aller quoi et ce que je retiens dans toutes les cases pour pour faire circuler l'information. Ok, et qu'est-ce qu'on a vu du coup comme algorithme qui permettait de de résoudre nos problèmes de RL avec de la programmation dynamique ? Les slides, non, vous pouvez les voir. On va mieux ta région ? Très bien. Et qu'est-ce que ça ? Qu'est-ce qu'on fait dans le Val d'Itération ? Ouais donc pour répéter on calcule toutes les valeurs à l'instant T et on les propage à l'instant T +1 pour avoir plus d'informations. Et pour faire ça, on se base sur une équation en particulier. Ouais, melman ? Qu'est-ce qui peut me réciter belman ? Pareil ? Belman vous connaître par cœur ? Le daming promis, c'est fondé sur l'équation de Bellman. Qui s'est écrit comment ? Non mais oui, je vous remercie. J'aimerais que vous me donniez l'équation de Bellman, sauf si j'ai mal écrit il y a bien 2. Alors comment est-ce qu'on écrit l'équation de Bellman ? Allez, vous devriez l'avoir sous les yeux, là vous l'avez dans vous l'avez dans les slides. Pas très bavard aujourd'hui je trouve. Alors belman on écrit quoi en fonction de quoi ? Déjà l'idée, l'idée générale, c'est quoi ? Ouais ouais, on lit le l'État et l'action. Ouais et quel est ce qu'elle a ? C'est quoi l'intérêt de la ? On a dit qu'on voulait propager l'État était +1 sur l'État t parce que on savait des choses quand on allait plus loin dans le jeu et on voulait propager sur ce qu'on, sur ce qu'on a fait avant. Donc théoriquement, quoi ? Comment est-ce qu'on va déjà ? C'est sûr. Quel ? Quel ? Quelle quantité de question de Batman ? Ouais donc on va écrire quelque chose en fonction de quelque chose de St +1. En fonction de quelque chose de St. Ouais, j'entends un truc qui m'a plu, ouais. C'est l'espérance de Ouais mais donc qu'est-ce qui ? Qu'est-ce qu'elle a déjà ? C'est quoi la l'équation de Bellman ? Elle porte sur quoi de base ? Cousin quoi ? Du coup ouais Ouais, y a un gamin dans le Store, ouais. 2R, 2T. Espérance de RDT plus. Plus gamma. Fois beaucoup. Gamin devait de ? Okay ? Euh. Chant. Sachant est égal, hop là. Ok, donc ça Pour rappel, on l'avait démontré. Et c'est le genre de truc que je pourrais vous donner vous donner à au partiel. J'irai pas beaucoup plus loin en termes de maths parce que En bref, c'est pas grand chose, mais si vous avez compris ça. Eh Ben je pense que vous avez de quoi. Avoir la bonne mathématique. C'est on arrive à manipuler, on sait qu'on parle, ça c'est une probabilité, c'est une variateur, ça c'est pas une valloire. La politique elle est trop facile, donc c'est une probabilité ou pas. C'est un communiste. Ok ça, ouais, ça se. C'est pas compliqué, mais il faut vraiment faire l'effort à cette fois de demander aux que ça c'est quoi ? Pourquoi ce que je peux écrire une espérance ? Pourquoi est-ce que la formule de Céline, elle me permet de faire telle ou telle opération ? Ok, bon, on va quand même essayer d'avancer. Ça se passe donc quand même programmé. On a vu attacher de ça. On peut propager de +1 jusqu'à SOK et donc on a plusieurs algorithmes derrière qui vont permettre de faire cette opération là pour résoudre le problème, on avait vu le value valuation. OK on avait vu policy Edition. On avait vu. J'ai plus le nom en tête, un truc directement avec la fonction Q. Où il y a des histoires pour. En gros on termine, on regarde tous les États du labyrinthe et une fois qu'on A la fin, bah en fait on peut propager la fin jusqu'à toutes les étapes précédentes. Ouais, ça vous revient à l'esprit un peu. Ok donc ça c'est des algorithmes de programmation dynamique parce que à chaque fois les banques on retient tous les États possibles et on on les propage de l'un vers l'autre à partir de de schémas de du MDP que vous avez montré pour trouver la revue function ou la ou la fonction Q sur. Tous nos sur tous nos États et donc rapidement. Dernier point, qu'est-ce qui peut me définir la fonction Q, du coup, hop, ça va être important aujourd'hui. On a défini la value function et l'action value function. Je ne sais pas exactement la même chose, c'est proche, mais. Ouais et Ouais, ça prend en entrée l'État et l'action. Tu peux les faire directement comme ça. Bah tu peux tu peux normalement la somme tu l'as là non ? Qui peut remplacer la somme ? Dans le premier coup hein, j'avais fait une fiche avec les définitions comme ça. Donc c'est quoi la la ? On a, on a parlé de cul learning, et cetera, si vous savez pas ce qu'on sait que dessus ça va être compliqué. C'est quoi le cul là ? C'est quoi l'action value option si vous voulez ? Et donc du coup l'action Valley fonction, elle associe à quoi donc là associée à SC et a ? Pour. Pourquoi pas ? Ça a un peu à voir parce que justement quand on va faire du cul learning, on va utiliser le maximum de de possibilités. En pratique, jusqu'ici c'est ça cette fonction là c'est habitable parce que on avait. Une table avec ici toutes les tous les pardons, tous les États possibles. Hop, et ici toutes les actions possibles. Et donc là, à chaque fois, j'avais une valeur de Q, et cetera. Et justement, quand on fait, quand on dit qu'on fait du cul learning, c'est ça qu'on fait, on apprend la table. Mais par contre ma question c'est c'est quoi en fait ? Plus de s un à un, c'est quoi ? C'est le Max de tous les États. À partir de quel on peut arriver, ça changerait ça exactement. On est pas loin la moyenne, non ? C'est très proche de ça, hein ? La value fonctionne, c'est quoi ? Ça sert juste à dire à quel point est-ce que quand je me trouve dans un État c'est bien et ça ça nous aide à prendre des décisions. Parce que si on sait quel État à côté de nous est le meilleur, bah on va vouloir aller vers cet État, c'est logique. OK la queue fonction c'est la même chose. Sauf que ici on on on paramètre par l'État et l'action. Et du coup, si je dois le définir ? Ça vaut quoi si on s'inspire de ça ? Espérance, ouais. De. Ouais et le gain c'est ça qui nous intéresse en fait. À la fin ce qu'on veut maximiser, ce que notre agent il veut maximiser c'est son gain. C'est la somme de ces récompenses. Ok donc c'est l'espérance du gain sachant mon état S et l'action a. C'est à dire, je suis dans l'état S et Ben je vais regarder grâce à ma cultivable, je suis en état s ? Je vais regarder la. Hop. Donc là c'est grand s hein, je suis dans l'état s, je regarde la la ligne qui correspond à l'état s dans ma table et là juste à dire Ah bah si je fais l'action a 3 et bah j'ai la meilleure que. Enfin j'ai la meilleure valeur de Q sur ma table. Donc je prends cette action là parce que ça maximise mon gain à la fin, ça maximise l'espérance du mal. OK. C'est clair pour tout le monde ça, franchement ça faut le il faut le savoir. Il faut l'ai déjà dit OK. Bon euh. Et bien maintenant il est temps de, il est temps de d'avancer sur le cours d'aujourd'hui. Ce que vous avez une question d'abord sur tout ce qu'on vient de dire. Avant, avant soit clair. Vous avez dit quand vous avez pris l'équation de base, vous avez dit valu pour les citer région et que ça sert enfin c'est 3 bannières de faire ça sert à quoi par rapport à l'expression de balmont ? J'utilise tout le monde, c'est un peu plus, je vais vérifier tout dispositif. Learning c'est plus tard qu'on l'a vu, on a vu la séance 3 donc en fait dès que t'as itération dans le dans le dans le nom de l'algorithme, ton investisseur on avait valu et policiers. Les 2 ? Et donc ta question C'est pourquoi est-ce que ça que ? Pourquoi est-ce que ça utilise belman ? Ouais donc dans. Dans le policier donc non plus simple nation, on cherche à partir d'une Poly donnée à trouver la value function et pour ça on itère en. En propageant nos États de de l'un vers l'autre. OK, c'est à dire si ton l'état final de ton labyrinthe là ou t'as une récompense de 10. Il est là OK ? Donc là je sais que R grand t égal 10 parce que je viens de terminer le labyrinthe. Mais juste avant je savais rien sur mon modèle OK et avant je me situais à cet état là. Ok. Donc là RT moins un. Il va les 0 parce que j'étais toujours dans le labyrinthe et que j'ai j'ai, j'ai perdu même même en général, on met moins un parce que on veut minimiser le. On va minimiser le temps dans le labyrinthe OK donc là ce que je dis c'est que bah là si je fais une une approximation de base parce que je connais rien, je récupère juste la récompense parce que c'est la seule chose que je connais. Donc la value function à s ici hein, c'est moins un. Donc en pratique, quand on quand on va prendre la décision d'aller. Terminer le labyrinthe, on se dit Ah super chouette. En fait, j'ai une récompense de 10 qui arrive juste après. Ok et donc là je vais rétro propager ce que je sais sur l'état d'après, sur ma sur ma sur mon état s ici et et donc ça et et donc ça pour faire ça, pour lier St +1 avec St et Ben on utilise bellman parce que c'est c'est à ça que sert bellman, ça sert à relier les 2. Donc là tu vois ce que dit. Atman c'est que V de S ici ça vaut donc là si je le remplace dans l'intérieur donc ça vaut RT, c'est moins un mois d'accord ? Hop enfin ce qu'on enlève l'espérance, hein, parce que il a entendu l'espérance, ça veut dire qu'il faut faire un il faut le faire plein de fois pour avoir une approximation. Ok, mais si en première approximation on dit que on apprend aussi une espérance par la valeur exacte, OK, on pourra dire que c'est moins un plus gamma fois V 2T+1 qu'on connaît parce que on vient de on. On on vient d'y arriver parce qu'on connaît la récompense au dernier moment. Et comme la, quand on arrive à l'État terminale, ici le gain en fait c'est juste la récompense finale. D'accord, donc la value function quand je suis à l'État final c'est 10 d'accord OKYA plus d'expérience, puisqu'on est déterministe. On a, on a terminé quoi, donc plus. Yama fois 10. Donc en fonction de ton gamma, ça va prendre une certaine valeur. Et donc là, grâce à BELMAN, je vais pouvoir dire que mon V ? De s ici je sais pas, il va être environ égal à 9 fois. Si on prend, si on prend-on a 1OK. Donc là on a itéré parce que la première fois on avait moins un et 10 et la 2e fois on propage l'un vers l'autre. Et là maintenant j'ai 9 et 10 et on sait en fait. Du coup maintenant que je le remets plus proprement ici, j'ai une value function de 9 ici j'ai une value function de 10 ici par le même par le même processus je vais voir une value function de 9OK et au coup d'après. Là j'ai toujours moins un parce que je j'étais pas arrivé au bout du labyrinthe mais à l'itération d'après, et Ben hop là cette fois je vais avoir combien ici ? Gamma égale à un c'est okay donc là à chaque fois que j'y terre. Je vais propager plus loin dans le labyrinthe la connaissance que j'ai sur la fin du labyrinthe. Donc c'est pour ça qu'on dit que c'est du Dany Programming et qu'on utilise bellmann. C'est que un on change le gros problème en plein de petits problèmes, juste que je vais regarder entre 2 États successifs comment ce que je fais et Ben et 2 j'utilise belman parce que Batman c'est précisément ce qu'il sert à lier ce qu'on sait sur les plages quelques-uns avec les. Bien compris le SV de St +1, on sait ce que c'est, c'est pas. C'est comment dire là quand au au premier coup tu vois si j'ai moins un et moins un ? C'est +1 j'ai pas le choix, faut que je j'utilise la valeur que j'ai là donc en pratique je vais rien changer ici. Ok. Donc donc je j'utilise la valeur que je connais mais c'est pas forcément la vraie valeur. Et en fait ce que ce qu'on a montré dans le premier cours on va pas le refaire, mais c'est que c'est une approximation. Donc en fait on est garanti que ce truc là enfin ce ce valuation là on converge vers la vraie valeur à la fin. OK donc donc oui non. En pratique quand tu le code tu utilises pas la vraie valeur, ce que tu la connais pas, sinon t'auras pas le code pour vous. Par contre tu sais qu'à la fin si tu le fais suffisamment d'étapes, et Ben tu vas arriver à ta ta étape de valeur de de telle édition et c'est et d'ailleurs c'est les donc le policier par le value interrogation. Vous aurez une preuve de confiance, c'est tout le reste. Donc là aujourd'hui on va commencer à faire des neurones et compagnie. Là vous oubliez toute forme de forme de preuve ou quoi de convergence vers un optimum ? Ok, là on va chercher dans des espaces de dimension énorme des solutions à peu près OK. On verra que par super bien hein, mais quand on fait ça il y a un petit côté matheux, on trouve la solution et et on est satisfait quoi. C'est aussi pour ça que en R, quand les gens sont arrivés avec des réseaux de neurones, tout le monde a dit Bah Attendez nous on a des preuves que notre truc il converge vous qu'est ce que vous arrivez après la dietman ? Qui a qui a sorti les qqn et qui ont gagné sur des Jeux Atari alors que personne savait comment faire ? Ils ont dit, Ah, c'était peut-être intéressant, quoi. Bon bref, donnée programmée ensuite, on a vu les premières middle, les. Ah oui, c'est ça les premières méthodes modèle Free. Avec quelqu'un peut me citer des des algorithmes qu'on a fait assez classés de la séance Moda Free. Il y a plusieurs mots clés là que que j'ai noté, moi. Modèle Free, c'est quoi déjà ? Référence à quoi on termine ? Apprendre les itérations comment apprendre des itérations, apprendre des itérations ? Ouais, ouais, de manière non explicite. Ouais, ça peut ça, ça peut rentrer dans la définition. Ouais, je vais rentrer un peu plus spécifique. Ouais. Ouais. Exactement. Modèle Free, ça veut dire qu'on ne modélise pas l'environnement, c'est-à-dire tout ce qu'on a en entrée, c'est un État et des actions et nous on fait juste dans cet État là. Je fais telle action, mais je vais pas chercher à prédire ce qui va se passer dans l'environnement OK ? Alors que dans la vraie vie on le fait tous. On a déjà parlé mais. Donc modèle Free ouais modèle Free c'est les c'est la majorité des méthodes de l parce que en RL il y a souvent notre notre, notre environnement. Il est beaucoup trop compliqué pour qu'on puisse le modéliser si notre si notre agence est un robot vous pouvez pas modéliser le monde entier. Donc vous avez simplement modéliser le la mécanique de votre robot à vous et vous allez faire du modèle Free dans la mesure où le modèle il va faire seulement considérer son État à lui et il va pas chercher à prédire l'utilisateur dans dans 3 secondes, il va juste regarder l'utilisateur à l'instant T et faire en fonction. Très bien donc modèle Free et on a vu. Ouais est ce que c'était dans ce type de modèle ? Mais on avait le justement le le poids gamma qui permettait de prendre plus ou moins en compte les événements passés. Ouais bah le gamin il est là. On a déjà écrit dans le gain. Et donc oui, dans les modèles Free, on utilise aussi le gamin. Je sais pas si t'avais un truc plus spécifique en tête mais. Un facteur qui pouvait faire varier le le modèle. Garder plus ou moins en compte les actions précédentes pour faire ton choix sur l'action future ? Ouais. Donc c'est en effet le cycle c'était plutôt le Alpha du coup je pensais quand quand on a parlé d'instrumental mine ? Ouais c'est quand Alpha est égal à 1/2. On prenait en France, ça c'est le Alpha et l'Incrémenter donc tu pourras regarder dans le groupe. Et en effet, là c'est aussi un moyen de c'est un moyen. Ben j'aimerais vous faire dire le mot en question. C'est un moyen de d'arbitrer un un compromis qui a beaucoup en R et que j'aimerais que vous faire dire quand on quand Alpha égale 0, on oublie tout ce qui se passe avant. Ouais, je peux faire un rapide point sur ce que c'est. Ouais, entre l'exploitation et l'exploration, il faut toujours explorer pour avoir des informations, pour les exploiter, pour essayer d'avoir la meilleure position. Exactement. Et donc là dans le cas dans le cas de montmin, l'exploitation ça veut dire OK j'ai trouvé une solution qui marche à peu près et Ben je vais tout le temps faire cette solution là. Je vais pas chercher à explorer de nouvelles choses donc là ce serait Alpha égal un je trouve un truc c'est bon je vais faire que ça tout le temps et je vais pas chercher à faire la solution à côté et l'exploration du coup c'est on on garde une une variance forte sur nos actions et on et par cet axe par comment dire par ce choix là ? On va faire des actions débiles, mais on va probablement découvrir des nouvelles, des nouveaux endroits ou ou des nouvelles nouvelles techniques potentiellement. En général, ce qu'on fait dans les algorithmes de la R, et on va le faire aujourd'hui en CP, on va avoir un facteur d'exploration qu'on va être très très fort au début pour bien explorer tout et petit à petit, au moment où au fur et à mesure qu'on va avoir de la connaissance sur notre milieu, on va diminuer notre facteur d'exploration pour bah pouvoir exploiter un peu plus et aller plus loin dans le dans l'environnement. Ok, et donc dans le cours ? Dans le cours. Sur les modèles Free, on a parlé de Monte-Carlo, on a parlé de bootstrapping parce que bootstrapping, ça vous parle ? Non, parlera un peu aujourd'hui de stratégie. On a vu les 2, les 2 premiers algorithmes de. Qui sont samsa et que leur King, ça vous parle ? Donc ça pour le coup c'est des très grands classiques, ça recycle learning, faut vraiment les connaître. C'est pas voilà je sais il y a beaucoup de choses dans ce cours mais c'est pour ça que j'essaie vraiment de vous pointer à chaque fois sur les les les choses qui sont très importantes. Sortie curling c'est vraiment les les algos de base en OK bien savoir qu'est-ce que c'est ? Pourquoi ? Sur la ligne, c'est pas le fait que le modèle il est après la table avec toutes les récompenses de ces actions. Utiliser ce que tu peux regarder, ouais. Est-ce que je l'affiche ? Oh ça. Alors ça vous, ça me rappelle des choses. Ouais. S'il vous plaît ? La mise à jour de la valeur, ouais. Pendant faire ça, il faut connaître l'action suivante. Enfin selon la la politique on a pas besoin de faire le Max exactement donc que learning notre police ce sera toujours un Epson ingrédient. Donc dans tous les cas on fait toujours un peu d'exploration, mais en même temps on se rappelle quand même parce que à la fin, on va quand même faire la police optimale alors que dans samsa on est, on apprend la policier en même temps. Et donc au fur à mesure de l'apprentissage on va apprendre des choses. De plus en plus important donc c'est encore une fois une histoire d'exploitation exploration, donc t'as tout à fait raison et une petite subtilité que je voudrais. Ce que je voulais rappeler ? Entre les 2, enfin, sur le gouvernement en particulier, il y a encore une fois c'est encore le problème du RL. On parle de Q Learning dans le sens où on modélise Q plutôt que V par exemple. Donc on on va être basé sur la action value function. Mais en même temps le the learning en tant que tel c'est un algorithme, donc là le learning c'est un algorithme de l ? Et pour ça qu'on évite de dire on va faire du cul learning pour dire on va juste modéliser une table, une table d'action value. Mais en soit enfin le la nomenclature de ce rythme là est pas est pas incroyable OK ? Et donc. Est-ce que c'est ce que tu disais ici, on apprend bien notre table cul. Et pour faire notre update, OK on est encore sur quelque chose qui ressemble à bellmann hein. D'accord on a toujours du RT plus gamma fois quelque chose. Sauf que là au lieu de prendre. Au lieu de prendre la valeur de Q à l'état d'après, on prend le Max possible, c'est-à-dire qu'on on ne dépend pas de la police là où avec. Avec farsa. Et Bah vous voyez que on prend bien le la valeur de QA été à s prime à prime OK. Et donc là il y a bien une notion de One policier versus of police. Ok, ça, ça doit être plutôt bien connu quand même. Vous êtes obligé d'apprendre par cœur l'algorithme, mais en tout en tout cas savoir que dans les dans les pardon, dans les algorithmes de RL en tabulaire. Donc quand on parle, on travaille sur des tables en en modèle Free, il y a 2 grands algorithmes, ça, ça tu le mets OK ? Et donc la notion de. La notion de modèle Free model bail on policy versus of policy on vient de refaire le point on policy, on utilise la policy actuelle pour. Pour apprendre notre value function action value function OK et of policy. Donc là comme dans peeling là dans sa RSA on est on policy dans cul learning on utilise pas notre policier pour prendre la décision, on utilise un hippie ingrédient ou un juste un quelque chose qui explore quoi ? Et en même temps on apprend la value function associée à la policier optimale. Okay ? Ouais. De quoi la donc pour le of policy, on va apprendre. Pour le officie, on va apprendre notre. Grande valeur ou son action valeur. En utilisant une politique qui n'est pas la police optimale. Ok. C'est souvent intuitivement. R, on peut se dire Bah vu qu'on apprend la policie au fur et à mesure, autant utiliser cette policier là pour avoir les meilleurs. Pour avoir les meilleurs résultats, il faut apprendre encore plus d'accord dans beaucoup de cas c'est vrai parce qu'en fait, si vous prenez un petit ingrédient dans un mode dans un jeu très compliqué, en fait votre option grédy votre il va être tellement nul. Il va jamais explorer suffisamment le jeu pour que vous puissiez apprendre des des choses. Sauf que dans certains cas vous voulez pas converger trop vite vers le la police optimale et parce que la police optimale elle va pas suffisamment explorer et donc vous allez vouloir faire du of policy donc où on on explore donc. Euh, on explore différemment et on prend ensuite le maximum des valeurs que. On a exploré à partir dans les psylone greedy, typiquement. Réponse à poursuivre vers des actions qui ont maximisé la récompense. On policier, ouais. Un policier, on était directement la valeur de l'action qu'on vient de prendre. Alors que of policier, on s'autorise à suivre une une politique qui est différente que celle qu'on apprend quoi ? Parce que là, typiquement, on apprend plus. Donc on sait qu'à partir de Q, on peut avoir la police optimale parce qu'il suffit de prendre la remax à chaque fois. Pourtant là il dit bien, on choisit l'action. À partir de l'État utilisant une une policier dérivée à partir de pub et donc c'est pas c'est pas la la politique optimale quoi. Par exemple, on va prendre épisode crédit parce que on se dit on se garde la possibilité d'aller explorer un peu à côté. 1 h de révision je pense, c'est bien, on va faire le cours, on aura pas le temps de faire des petits. Ok. Donc aujourd'hui on va commencer à faire de l'approximation du value function ou de l'approximation direction value function. Et donc le constat, je l'ai déjà dit, c'est que. En pratique, pour l'instant on a fait uniquement du R tableau ou à chaque fois on me disait tous les États possibles. Donc là ce que vous pouvez imaginer, le genre de le nombre d'États qu'on a dans ce genre de jeu là. Ça fait beaucoup hein ? Si on fait, si on fait rien que rien comme l'image hein, ce serait pas la manière la plus la plus intelligente de faire de résoudre ce jeu ? Rien que l'image vous avez quoi ? C'est 1080P. Ceux qui jouent en 4K c'est encore pire. Le nombre de pixels, fois 300 256 au cas au cube. Ça fait beaucoup beaucoup d'États donc en pratique c'est impossible de retenir tous les États possibles. Et même si vous pouviez le faire en fait si jamais le pixel qu'est là échange de couleur parce que je sais pas il y a un avion qui passe ou ou j'en sais rien en fait. Vous en fichez vous voulez pas avoir un État différent pour pour pour ce jeu là OK. Donc on va vouloir en fait avoir des États qui sont enfin circuler dans un espace qui est beaucoup plus petit et approximé beaucoup de choses. Donc par exemple si le bonhomme là il fait pas, si ça a l'air d'être un pote si tu mal là. Il est pas là ? Bah peut être qu'on voudra faire la même action parce qu'en fait on on veut pas baser sa décision sur la présence de ce mec là alors qu'on s'en fiche quoi. OKYA un peu toutes ces idées que. On n'a pas envie de faire du du R tabulaire quand on a des des situations réelles et donc compliquées. Donc on va arrêter de tout mettre dans une table et on va faire des approximations, OK ? Donc. Le le paradigme s'appelle value function approximation, toujours une histoire de mots-clés et maintenant on va paramétriser. Nos fonctions c'est à dire que notre notre value function ici ou l'action value function. On va les paramétrer, c'est-à-dire on va, on va leur dire OK on va, on va, on va vouloir trouver des valeurs. Il va falloir trouver un espace dans lequel stocker cette fonction donc. Et la manière de stocker une fonction d'une certaine manière c'est c'est ces paramètres. Puisqu'il définit à fond fonction. Donc si définit une droite par exemple, c'est le lecteur que votre droite est pas suivre, c'est des paramètres. OK pour ça qu'on dit qu'on paramétrée la fonction. Donc on l'utilise dans des États dans des États continus. En grande à grande échelle. Ouais, donc on a beaucoup, beaucoup de de possibilités. À partir de là, à partir du moment, on a décidé de faire de l'approximation. La plus connue qu'on va bien entendu traiter, c'est les réseaux de neurones. Ok, mais il y a aussi beaucoup de choses. Y a aussi des fonctions qui sont, on peut utiliser des approximations linéaires. On va voir, ça peut être intéressant dans certains cas et et des arbres de décision. Je sais pas si vous allez faire un peu de ml en pratique quand vous avez des données. Le premier truc à tester c'est pas un réseau de neurones. Le premier truc à tester c'est que j'ai boosté. Ça vous parle ? J'ai de zébus. C'est quoi ? Bah c'est un c'est des arbres de décision hein. C'est pas un réseau de neurones. C'est super. Enfin, c'est beaucoup plus simple de principe que qu'un réseau de neurones. Donc c'est des arbres de décision. Et en fait ça marche super bien. Des arbres de décision OK c'est pas, c'est pas. C'est peut-être un peu bourrin et simpliste, mais en fait ça marche beaucoup. Si vous Regardez sur sigle donc taggle c'est ça rassemble tous les tous les défis de data science. Souvent vous pouvez même faire beaucoup d'argent en gagnant un concours qui gueule parce que les entreprises elles publient des problèmes. C'est OK, on a telle donnée, on veut, on veut prédire tel paramètre. Faites nous un modèle qui marche et nous on vous paye quoi ? Et bah sur Kyle, la plupart des des des méthodes utilisées par les par les gens qui gagnent les challenges en fait c'est des c'est des boosts en. Donc pas forcément crier tout de suite au réseau de neurones quand on a un problème comme ça avec. Donc voilà, à pas oublier. Même si en l en pratique on fait beaucoup de neurones quand meme. Je ici. Si. On parle de notion de de fonction différentiable, OK. Pourquoi est-ce qu'on utilise des fonctions qui se différentielles ? Parce que quand c'est différentiable en plus calculer des gradients et quand on a des gradients on peut faire des descentes de gradients et donc on peut optimiser notre fonction OK ? Ça devrait être assez clair pour tout le monde. Donc, qu'est-ce qui différentiable ? Les fonctions linéaires, les réseaux de neurones, malheureusement pas encore les exhausts. Et c'est pour ça qu'on préfère qu'on préfère ce genre de de méthodes différentiable en ok là y a pas forcément besoin de de de méthodes différentiables dans le problème classique des de. Nation. Et. Donc. Le. Dans le cadre d'études aujourd'hui c'est on a notre value function l elle existe, elle définit. OK enfin on la connaît pas, on aimerait la trouver. On sait qu'on peut pas la trouver pour toutes les valeurs possibles parce que c'est impossible s'il y a trop d'États possible, donc on l'approxime par une fonction qui prend en entrée notre État. Avec des qu'on va paramétriser. Ok, donc les paramètres ici je le mets dans dans le Var. W. Donc typiquement ici la la fonction coût de notre de notre modélisation, ça va être l'erreur qu'on va faire, c'est à dire la différence entre la valeur réelle et le. Et notre notre approximation. OK. Euh. Donc là on écrit simplement quelle forme peut avoir notre notre régisseur ? OK donc là valy fonction chapeau ça veut dire ça veut dire je veux dire régressa un mot plus précis pour dire ça dans stata estimateur. Merci donc donc on sait on utilise cette notation là. Donc notre estimateur de V donc ça bien prend bien R on est à S et on on met en paramètre le W parfois on écrit aussi V Underscore en V indice W parce que en soi ça reste une fonction qui prend en entrée uniquement si c'est pas vraiment WW ils sont à l'intérieur du modèle quoi. Oui OK donc les les 2 existent donc là faut pas. Faut pas, faut pas être paniqué, c'est toujours une fonction qui prend en entrée de ta s simplement la des paramètres AMPLI OK et donc typiquement pour. Pour une fonction linéaire, bah on va pouvoir faire une combinaison linéaire de certains vecteurs. Je prendrai en. Donc là donc là c'est des fonctions de s donc on va faire une combinaison linéaire de fonction de OK et donc on va apprendre les poids de chaque fonction I c'est le poids de la fonction OK, donc on va apprendre les poids optimaux pour. Pour vous, pour résoudre un problème, OK. Donc ça c'est linéaire et c'est pratique parce que quand c'est linéaire c'est facile. Et malheureusement quand c'est trop compliqué Ben on fait plus, on a des choses sont plus linéaires et on peut modéliser par un gros n donc pas de neurone. Et là vous pouvez faire ce que vous voulez hein ? OK vous allez encoder un tout à l'heure. On va voir ça. Donc ouais donc voilà notre futur Victor. Ok. Donc l'imitation n'est rente des. Les modélisations linéaires c'est que vous pouvez pas modéliser autre chose que de droite OK donc quand quand vous avez les données de ce genre là bah en fait une fonction lumière vous allez avoir du mal OK ? Après y a des ce qu'on appelle des tricks quoi, qui permettent de faire du linéaire même si le la forme de la fonction est plus compliquée. Donc je sais pas si en en ml vous avez fait des du des trucs du style SVT et tout ça. Le non, ça vous parle pas ? Non ? Bah en gros l'idée c'est que vous gardez quelque chose de linéaire et vous faites une transformation qui fait que votre espace sous-jacent, il devient plus linéaire. Donc vous faites quelque chose de linéaire sur les futurs lecteurs qui sont compliqués. Et en fait c'est les les features vecteurs compliqués qui vous permettent de transformer ça en. En modèle non linéaire à la fin, mais c'est pas en pratique, les réseaux de neurones le font aussi donc. Donc on est ça OK ? Rapide lien avec ce qu'on disait précédemment, OK. Si jamais donc précédemment quand on fait de la value function quand on faisait de la value pardon quand on faisait value éthérisation par exemple sur notre labyrinthe, OK à la fin ce qu'on obtenait c'était une liste avec pour tous les États possibles la la valeur exacte, enfin la valeur associée. D'accord, donc en fait ça c'était ce qu'on pouvait obtenir en faisant valu interaction ou en faisant du cul learning ou du samsa par exemple. Ok, en récupérant la fonction valeur par la fonction. Donc en fait ça, ça revient. D'une certaine manière, à l'écrire comme étant l'est. Amateur avec un produit scalaire avec le notre poids ici et le X ainsi, qui est simplement le la base canonique. On est d'accord, c'est juste une manière de l'écrire pour vous dire que finalement. On pourrait écrire nos notre R fabulaire comme. Comme des. Comme un problème modélisé par des fonctions linéaires. OK. Et l'intérêt justement c'est que maintenant qu'on l'écrit comme ça, Ben on se rend compte qu'on est pas obligé de mettre que des zéros et des uns dans notre dans notre site vector. Et en fait ça se trouve on va pouvoir mettre des points un peu plus compliqués pour modéliser les choses plus compliquées. Ok donc ça le le le le passage est un peu plus facile. Et enfin un réseau de neurones c'est ce que c'est un réseau de neurones ça va et Ben on va pouvoir modéliser pareil notre vecteur comme raison neurones. Et donc là on va avoir plusieurs possibilités. En sortie, on aura bien la la valeur OK d'autres estimateurs, il est là et en entrée on aura donc l'État. Donc soit on va choisir de mettre tous les États possibles et d'activer un neurone par État possible, soit on va modéliser notre État comme étant enfin avec des valeurs, des valeurs réelles. Le mettre en entrée du réseau de neurones et et récupérer la valeur. Là on verra aussi pareil, y a beaucoup de manières de faire, donc dans l'idée on modélise simplement par zone grand. Ensuite, 2. 2 petits mots clés, généralisation versus discrimination. Ok et pourquoi est-ce que on a pas besoin de se rappeler de de tous les États possibles ? Ok donc là dans ce cas-là notre robot une fois qu'il a appris à faire le chemin du haut ? Bon, on se dit qu'il n'a pas à prendre de 0 pour faire le chemin du bas hein. Ok donc on dit qu'il va généraliser, c'est-à-dire que on a pas besoin de de remplir toute une autre nouvelle partie de la table. On va utiliser ce qu'on connaît déjà et on va simplement généraliser. C'est à dire il va voir quelque chose d'un peu différent, mais il va quand même bien se comporter. OK ? C'est ce que je disais tout à l'heure avec. Avec le jeu là-haut, là si le si l'écran il change un petit peu à des endroits pas importants. Et Ben on s'attend à ce que le modèle il généralise bien à cette nouvelle situation parce que c'est quand même une nouvelle situation. Il a jamais vu, mais il va quand même se baser sur les les cultures importantes pour avancer. Et enfin, discrimination, c'est justement parfois. Même si l'État, il est très proche. Là il y a juste entre guillemets un mur entre les 2, Ben on va quand même vouloir bien séparer les étapes parce que là on va pas vouloir qu'ils, on va vouloir qu'il fasse un chemin très différent entre les 2. Ok, donc maintenant qu'on fait le plus du tabulaire ? Chaque État. Et et et un peu toujours nouveau et donc il va falloir être. Être un peu malin pour que. Euh bah à la fois on puisse rassembler des États qui sont très proches pour se comporter de la même façon et à la fois séparer des États qui peuvent paraître très proches mais dans lesquels en fait on va vouloir avoir des comportements très différents. Ok. Donc c'est toujours une idée de on part du principe que chaque État il est-il est-il est unique maintenant. Mais. Et on va vouloir le modéliser de la meilleure manette. Vous avez fait du ml donc vous savez vous savez ce que c'est l'apprentissage supervisé ? Ouais et Ben maintenant en fait. Une fois si on prend un principe qu'on vient de faire un épisode, bah en fait on a un ensemble de État gain État gain État gain d'accord. En fait une fois qu'on a ça c'est facile. Parce que une fois que vous connaissez les gains réels, vous avez votre et votre y et vous pouvez faire escaladeur point ce que vous voulez. Fit XY d'accord. Donc en fait maintenant vous pouvez faire du superbe learning avec les données que vous avez récupérées. On est d'accord. Donc là toutes les méthodes que vous connaissez sont applicables. Vous avez un estimateur V de S. Et Bah vous flattez votre estimateur sur vos données de la manière dont vous voulez. Ok ? Et donc toute la subtilité en RL, c'est que. Et Ben vos données, vous pourrez jamais avoir des données, toutes les données possibles parce que justement on n'est pas en tabulaire et vous allez pas attendre de faire 5 milliards d'épisodes pour explorer tout fortnite ? Parce qu'en fait vous pouvez même pas. Donc la subtilité c'est que certes on peut avoir des données, mais les données sont ultra biaisées. Il va falloir à la fois savoir en récupérer suffisamment pour apprendre des régisseurs pour ensuite peut-être aller un peu plus loin, et cetera, de la même manière que ce qu'on a un peu fait précédemment, mais avec maintenant des des estimateurs et plus avec une table. Ok, mais dans l'idée franchement, vous pouvez là tout de suite commencer à coder votre truc. Faites un épisode, vous récupérez des gains et et des. Et des États ? Et vous pouvez commencer à apprendre à n'importe quel agresseur. Hop hop. Et donc de la même manière pour le pour la polythéismes là simplement pour réussir avec Batman. On va avoir un État, notre récompense, plus la destruction plus tard et donc cette fois. Cette fois, on n'est plus en on attend plus la fin de l'épisode. OK parce que avec belman on écrit on on est, on réécrit avec l'état d'après, donc ça veut dire que vous pouvez pendant que vous êtes en train de jouer d'une certaine manière, vous pouvez déjà apprendre en fait. Parce que vous vous êtes dit Bah là je suis à l'étape 100 en fait, j'ai déjà 99 étapes derrière moi, peut-être que je peux déjà commencer à prendre des choses de ces étapes là OK ? Donc on écrit comme ça et donc là pareil, on a notre temps d'entrée et on peut essayer d'avoir des estimateurs qui est qui nous estiment que ça va se passer. Donc ça c'est l'idée fondamentale. Vous allez voir que ça va donner à des ça va donner lieu à des algorithmes un peu plus précis. On va voir. Euh. Et moi, petit, petit, petit ? Reprendre attention c'est que quand on fait ça, c'est comme la différence entre le temporel différence et Monte Carlo. On est très biaisé parce que si vous avez pas encore atteint le bout du labyrinthe, en fait on va mieux. Nos estimateurs de ville, ils veulent rien dire. Donc en fait vous avez rien à prendre. Ok donc y a toujours cette histoire de oui on peut le faire, mais en fait il faut maîtriser ce que vous avez. Et si vous avez pas suffisamment d'informations dans votre environnement, bah vous avez pas forcément vouloir faire du temple différence. Donc enfin vous allez peut-être vouloir l'écrire comme ça. On attend de finir l'épisode en entier pour avoir l'information. Ok c'est clair le Bootstrapping, enfin ça pourra différence versus Monte Carlo. Ouais, donc là on le fait, on le fait à la Monte-Carlo d'une certaine manière. Et là on la ferait à à la temporale. Différence temporal différence. Donc tu dis pourquoi ? Parce que on on a juste s 2 et s un qu'on a une différence temporelle. OK, c'est c'est juste ça que ça veut dire. Et donc j'en profite pour caser tous les mots clés pour que pour que ça rentre dans ce genre de cas, on dit qu'on fait du bootstrapping. Ça vous rappelle des choses ou pas ? Bootstrapping ça veut dire on va recoller des morceaux les uns par-dessus ça OK on m'attend pas à la fin c'est pareil, on recolle. Je fais un nouvel État OK Bah je j'ai un nouveau recollement, bah je l'utilise déjà pour apprendre des choses avec tout ce que ça implique de monter Carlo qui est non biaisé mais très fréquente variance. Et le mot strapping qui est loin de variance mais qui est très plaisir. Ouais, ça me rappelle les choses ça. Stop, OK, je. Et maintenant policier évaluation, c'est la même chose. Madame Thêta et notre Estimateur Ben on on peut essayer de fixer à chaque fois. Tu as comment est-ce que c'est pour que ? À la fin, notre fonction, elle. Elle correspond à la vraie quoi ? Et on garde messe. Ça vous parle j'imagine ? C'est ce qu'on a dit tout à l'heure. Notre loss c'est la valeur réelle moins la valeur de nos consommateur et on optimise mon paramètre W pour sa visite. Ok. Ouais on connaît la valeur réelle du coup bah c'est c'est pour ça que ça tu peux pas le faire en pratique. Mais toute l'idée c'est comme ce qu'on a fait avant. Tu vas, tu vas, tu vas voir un estimateur. Tu vas avoir des samples parce que du coup t'as la variance c'est là pardon le la vallée function c'est une espérance. Donc en pratique des gains tu peux en avoir en faisant un épisode. Donc tes ça ça va permettre de faire un estimateur du gain, une estimation du gain que tu vas vouloir faire correspondre à ton estimation à toi et t'as une estimation extérieure. Une estimation à ça, et c'est toute la difficulté. C'est ce que je vous disais. Certes on on a des des, des valeurs qu'on peut citer, mais en vrai on on veut. Enfin quand on a ça là, nous, ce qu'on aimerait en réalité, ce serait directement. La vraie valeur quoi pour la police optimale. Mais ça on l'aura jamais. On peut c'est faire un estimateur de de de cette valeur là qui est une une espérance sur un espace qui est très compliqué qu'on pourra jamais explorer en entier. Et donc c'est tout, c'est tout. L'art du renforcement learning, c'est comment ce que je fais à partir des estimateurs, en général pauvres, pour apprendre des choses quand même, pour arriver à aller plus loin, pour avoir des des estimations meilleures, pour fitter encore mieux mon modèle, et cetera. Donc là on peut-on voit qu'on peut pondérer par exemple nos États avec un estimateur d'importance. Pour par exemple se dire que on veut, on veut vraiment. Enfin on sait par exemple que la value function à la fin du jeu elle est exacte. Donc en fait on va vouloir que notre agresseur y soit absolument bon à prédire cette valeur là et pas forcément aussi bon. Pour prédire un État au milieu qu'on connaît pas trop. Ok. Donc ici on voit que vous faites ce que vous voulez quoi, votre fonction de course, c'est vous qui la définissez, c'est vous qui la codez. Bah si on veut ajouter un poids à certains à certains États, Ben en fait. Et je pense que il est temps de faire une pause, de savoir. Vous avez dit quoi ? Vous préféreriez faire 2\*15 Min ? C'est ça plutôt que une fois. Ouais bah très bien, on fait une pause de 15 Min, on se retrouve dans à 09h45. Vous avez pu me mettre présent ? Merci. Trop de cours en visio dessus, un peu les courants. Ah oui ? C'est un peu bougie 5 année, 4e 90, je suis très très bien. Des recommandations ? Donc oui en fait moi j'ai monté les mains, qui le croise dedans ? Ça, ça, ça va. C'est pas fini. Pour l'instant. Mais voilà, il faut des. Honnêtement, j'ai jamais trop entendu parler de telle application. Alors en gros, c'est encore pire quand. Alors je vais aller je ? L'utilisateur plein d'emplois ? Le match d'initiateur avec les meilleurs emplois pour une sorte d'accord, le problème fondamental c'est comment est ce que tu fais pour noter le match entre l'initiateur et ? Pour ça, on a plusieurs manières de le faire, tu peux noter. Alors, le truc ? Ça, c'est. Une fois que t'as 3 like ? À partir de ces 3 lik genre si sur tes 3 Lik t'as 3 fois Deep learning dans l'intitulé et tu vas dire Ah peut-être que la future diplômée l'intéresse donc ça c'est le. De. L'idée, c'est que on peut prendre la piste de base. Probabilité. La télé ? Like je sais pas trop. Ben en fait il y a. C'est proche du réel dans le sens quand même. Un agent qui propose de. C'est fini que si nous avions fait qu'il est un petit tournant pour retomber sur. Garde un peu de temps un paramètre. On va faire des tests et voir de là, il me propose pour ceux qui vont. Il dit, c'est l'ordonne, mais en même temps. Un peu le. Mais c'est à toi, tu va implémenter quelque chose qui prend-on part le la figure de l'utilisateur de base. Que tu vas modifier petit à petit en utilisant les les préférences que t'as. Mais en même temps de laisser quelques profils en général ce que tu vas faire c'est que t'auras un le meilleur estimateur. Tu vas avoir collaboratif ? Je pense que. Et ensuite ? Et ensuite je vais ok, là j'ai mis mon top 10. Des emplois juste en fonction de préférence. Bah je vais m'autoriser avec une certaine probabilité de prendre je sais pas 3 jobs dans le top 50. Peut-être intéresse un peu, mais c'est pas dans la conférence que tu sais pas si t'as dit. 3 fois. Mais peut-être que peut-être que il était quand même toujours intéressé par un peu de web dans ce qui est quand même pas trop loin parce qu'il a vu mais il sera pas dans le top 10 du coup tu vas dire bah avec probabilité vous savez bien tu vas, tu vas chercher. Le cœur de ton projet, de ton problème, c'est comment est ce que t'obtiens cette vraie liste, cette image ? En plus de cette image, tu vas ajouter des idées. En Colombie, elle est. Comprend quelques trucs sur haut en bas, on les, on est en. Oui, alors ? Si un utilisateur est. Il y a une pas avec les fonds publics. 2 trucs de Software mais un truc de faut les garder comme ça et ensuite sur un pourcentage de. Et du coup Tu Mates comme ça OK ? Par mots-clés ça ? Parce que t'as des erreurs ? Résidence du coup c'était des vrais vecteurs comprendra pas du coup on a plus de sens à priori que juste l'agro un sur : le 0 un c'est dans quel ? Et pourquoi juste 0 un ? Tu veux même ? Si si t'es si ton lecteur c'est Big data informatique en 2 et en 3 dimensions. Ton job t'auras un si c'est si y a des plantes dedans. 0 dans la lecture de Clermont un dans la future Big data 0 0 pour chaque objet ça une attends donc ça fait un vecteur de job et un vecteur utilisateur. Ici c'est la colonne ou quoi ? 0 un et tu vas pouvoir ajouter au retirer en fonction de ce qu'il a été par exemple, tu vois, ce serait une manière de faire. Vu. Plus vecteur. En plus sa future d'économie sa grande dans son lecteur et donc quand tu vas faire ton produit scalaire avec le job Deep learning plus le score est important. Ton job d'epernon va monter, on s'en fout si on a une manière de faire. Danse plus que quoi ? Mais les échecs on a genre 40 calculé lambda de ton job lambda de ton perso et tu vas à chaque fois. Retirer l'appli. Et après on fait là. De ma fonction V par rapport à l. Et c'est là que c'est important d'avoir un estimateur qui est différentiable. Ok parce que si vous avez pas un estimateur différentiable ce gradient là n'existe pas. Donc vous pouvez jamais vous pouvez jamais faire update de vos paramètres. Ouais. Et donc là intuitivement c'est c'est plutôt simple je trouve. Ça veut dire que si votre erreur est positive. Donc là en gros si vs il est plus grand enfin la valeur réelle est plus grande que la valeur que vous avez estimée donc ça veut dire que votre estimateur il devrait augmenter sa valeur à ce moment-là. Ok donc là l'estimateur est trop petit parce que ma ma mon erreur est positive. Donc le delta il va prendre la direction vers laquelle ? Donc ça le delta c'est ça, c'est la direction vers laquelle la direction selon W vers laquelle la prédiction va augmenter. Ok parce que c'est ça un gradient, c'est la direction vers laquelle elle fonctionne augmente hein, c'est y a de plus compliqué et donc je prends ce truc là et je le multiplie par l'erreur, c'est à dire je fais une très grande erreur positive. Bah on va aller très fort dans cette direction et on normalise, on normalise, on multiplié par le même régal. Donc si l'erreur devient négative, Eh Ben là le sein il va changer à l'intérieur de ça. Et donc on va faire le contraire, c'est-à-dire que notre notre estimation est trop grande et donc on va. On va prendre la direction vers laquelle on augmente et on va aller dans la direction opposée, chercher à faire diminuer notre fonction, voilà. C'est clair pour tout le monde. Là c'est très simple, c'est un estimateur, un estimateur classique en machine learning pour. Et ça vous donne un peu une impression de pourquoi est-ce que la MS ça marche bien en fait quand même super intuitif quoi. L'estimateur est trop grand, bah on prend la différence et on multiplie par le gradient et c'est très, ça marche très bien et c'est pour ça qu'on fait le carré. Parce que si on mettait pas le carré en fait on on enlèverait la dépendance en en la valeur directement, c'est-à-dire si on enlève, si on enlève le carré. On aura directement le gradient de ce truc là. Et on va pas le normaliser entre guillemets par la la vraie différence. C'est à dire que si votre modèle est un tout petit peu au-dessus de la valeur réelle et Ben on va lui donner la même. On va faire la même update que si le modèle était avait une erreur énorme vous voyez ? Alors que là avec la messe quand l'erreur est très fine en fait bah update est va être très fine aussi parce que le modèle il a pas besoin de beaucoup jouer. Donc voilà, si si un jour vous avez à choisir entre ma et messe pour un problème de MYA aussi rappelez-vous qu'il y a. Voilà et donc comme dans le cas Stochastique Bah c'est la même chose sauf que au lieu d'avoir une espérance et bah on va on va faire un exemple donc on va prendre directement un État s donné ou un un patch d'État S 2. C'est bon ? En fait, je je me permets d'aller vite là-dessus, normalement, c'est à peu près clair pour tout le monde. Et donc pareil quand on quand on prend. On remarque quand on prend l'espérance, en fait on se retrouve avec la même update ici. Et donc ça se recoupe quoi ? Quand on fait du Stochastic, ça revient à faire un sampling sur la valeur réelle de l update. Si on le faisait sur le bas de centre ? OK. Avec toujours cette idée de comme on sait que dans tous les cas ça reste faux. Ben c'est pas c'est pas grave, voire c'est mieux de faire ce camping là pour avoir des meilleures valeurs. Ok dans le cas d'une d'une valeur qui est réelle donc c'est notre estimateur, là on le remplace par X transposée W donc ça vous parle x transposer W on fait le produit scalaire entre. Enfin on paie. Pardon en fait le produit entre nos features et nos poids. Et bah notre. Notre-Dame est encore plus simple, on a même pas à calculer gradients, on le connaît déjà un des intérêts du un un des intérêts des des valeurs linéaires. Et c'est pour ça que la régression linéaire la régression linéaire en soit c'est juste un estimateur qui est fixé en MSE, d'accord. Simplement comme c'est linéaire le Gradient vous le connaissez et encore plus que ça, l'optimum vous le connaissez déjà. Donc vous avez pas besoin de faire une régression linéaire normalement vous avez déjà dû le faire en M je pense que vous allez faire une descente de gradient sur un truc linéaire et vous a montré que vous trouviez exactement la même solution qu'un qu'une régression linéaire et logistique. Parce que en fait, c'est y a l'optimum, il existe, il est unique quoi. Et donc dans le cas de de la l'approximation linéaire, en fait, la solution, vous vous la connaissez déjà nécessaire. L'intérêt, quoi. Vous pouvez avoir du coup, pas besoin d'entraînement, vous avez directement. Vous allez directement la solution qu'on par contre comme on a vu ça risque limité parce que vous avez pas un modèle linéaire il faut pas, il faut pas profiter d'un. OK clair pour tout le monde ça je pense que normalement, enfin vous devriez déjà connaître. Ok, donc. Maintenant si on applique ça à notre les algos qu'on avait déjà utilisé avant, je sais pas à good Monte-Carlo Monte-Carlo. Pour rappel on fait un épisode en entier et ensuite on update notre notre modèle. Ok donc pour tous les épisodes on récupère tous les toute la chambre avec toi donc. SARSARSAR jusqu'à la fin, OK. Et ensuite ? Ensuite, une fois qu'on a tout ça, on connaît notre gain, on est d'accord ? Donc le lien on a un estimateur de la moyenne qui est juste un sample et dans dans notre cas on va pouvoir faire peut être plusieurs épisodes pour avoir une valeur moyenne du lien par exemple. Euh. Et ensuite on fait notre update exactement avec la la, avec la formule que j'ai montrée tout à l'heure. Tout ça en fait là c'est le gradient de ma fonction coupe, c'est le c'est ma blonde J Ah d'accord, on fait la différence entre le bien et la prédiction réelle. Et multiplier par le gradient de le gradient de notre estimateur par rapport à W. Ok donc ça c'est déjà un algorithme qui peut fonctionner. Par contre avec le problème que là on a besoin de faire des épisodes en entier en utilisant PI. Donc si PI on l'a pas au début parce que c'est le cas en RL, Bah si vous avez pas moyen d'avoir une politique qui est à peu près OK pour explorer l'environnement, alors vous allez avoir du mal. Ok, donc dans un dans le compliqué on va préférer faire du un policier, parce que si on est of policy. Enfin c'est pour juste évaluer une policier en l'occurrence. Euh et Bah vous allez voir faire du online Learning quoi. Au bout de certaines étapes dans le dans le l'épisode déjà commencé à apprendre des choses. Hop, semi Gradient Descent. Donc toujours la même chose. Voilà, j'ai passé encore plus vite, c'est vraiment le. Ce c'est vraiment la même chose, sauf que cette fois on fait en temporal différence, on utilise un delta avec la valeur HT +1 la valeur STOK. Au lieu d'avoir une, c'est la, c'est la. Même pas dû mettre la tête d'avant. Donc avant étant monté-carlo maintenant on passe en Tom Ford de France OK donc cette fois notre update elle est là, c'est plus le gain. C'est plus le lien final que j'obtiens à la fin, mais je le fais pour chaque étape de mon épisode. C'est exactement ce que je vous disais. Et je remplace le gain par un estimateur. Biaiser, mais moins variant, vous rappeler. Qui est plus gamma V de l'État d'après. C'est bon pour tout le monde. C'est vraiment beaucoup de redites mais je pense que c'est bien comme ça, ça rentre plus facilement et donc la même update. Sauf que cette fois, la dernière fois dans Monte Carlo on faisait. On faisait bien un update mais après l'épisode en entier ? Mais là on fait un update à chaque fois qu'on fait un step dans les dans, dans l'environnement. Ok. Et donc voilà pareil, vous pouvez faire du gradient temporal différent si vous voulez maintenant. Hop. Tac, TAC TAC. Et donc là pareil pour pour le Gagnaire, bah le c'est le même délire. Votre gradient, vous remplacez par X de St. Et donc vous pouvez faire votre bête comme ça. Je sais pas si ? Ouais, dans le cas. Non, on va passer, c'est que tu vas regarder en gros là, ça peut donner une intuition de. Spécifiquement j'ai cette feature qui est arrivée à ce moment-là. Bah dans dans notre update on va updater uniquement le le WI et pas tout le reste, donc on va pas interférer avec le reste avec le reste de ce qu'on a appris. Ok c'est un peu comme dans le dans le cas du Tabular RL ou. Quand on apprend à partir d'un État I en particulier, on modifie la valeur de de l'État à l'endroit où on dans la table où on a appris cet État là et on va pas modifier ce qu'on a appris sur l'État 0 par exemple. Donc ça c'est intéressant parce que dans le cas d'un réseau de neurones en fait, quand vous faites votre update, potentiellement quand vous apprenez l'état 1000, Ben vous modifiez la valeur que vous allez avoir sur l'État un parce que tous les paramètres sont partagés. Ok donc c'est ça c'est ça l'intérêt, c'est ça l'intérêt du du modèle linéaire dans ce cas-là. Sauf que bon, encore une fois on est dans, on est dans un cas de où on peut avoir une feature par État possible et en pratique on sait que bah c'est pas possi. Ok. Ensuite là on fait un rapide point sur le tile Coding, donc le tile coding je sais pas si vous vous rappelez c'est le dernier TP qu'on a fait on va dire. On a fait, on a fait le Mountain car le montagne car là c'est un environnement qui est continu et on avait vu comment est-ce qu'on pouvait le discrétiser pour pouvoir appliquer nos méthodes de tabular RL dessus. Ouais. Euh et donc pour ça on avait utilisé du Talk Coding. Et donc en fait, ce qu'on a fait à ce moment-là, c'est que. On a transformé notre État continu, donc notre variable V là qui prenne en entrée à un État continu. Et on l'a transformé en en estimateur. Continue puisque ils prenaient en valeur en entrée une valeur continue. Simplement ils passaient par une table pour générer sa valeur. Mais bon à la fin ça reste un estimateur et c'est c'est un peu comme faire un décision très d'une certaine manière on sépare notre État en pleine case et en fonction de la case on se trouve on donne une réponse OK et donc en fait donc quand on fait du R de manière générale. Ça peut être utile de faire ce genre de choses parce que y a des problèmes qui se séparent très bien en en sous-espace, en fait en case. Donc au lieu de faire un gros réseau de neurones qui globalement quand même très continue quoi, donc avec des relais, et cetera, on va pouvoir quand même faire des des brisures. Mais au lieu de mettre un raisonnement qu'on contrôle pas trop, bah on peut se dire Ah bah en fait là mon problème il serait super bien si je sais pas mon espace en case donc je vais faire du tel coding donc mon estimateur c'est pas un réseau de neurones. Ça va être on va voir, c'est en fait avec une table. Avec un nombre de, un, un nombre donné de. Un nombre donné de de cases. Et dans chaque case, je vais retenir une valeur. OK, ça revient un peu à faire 1KN certaine manière, on sépare notre espace en un endroit clair et pour chaque endroit, on prend une décision. Ok. Donc en pratique, souvent ce qu'on fait c'est qu'on fait, on sépare notre espace, mais on va le faire avec plusieurs. Plusieurs tailings différents, plusieurs discrétisation différentes. Ça, ça permet d'éviter d'avoir des problèmes de prédiction quand on est au bord de chaque case. Parce que typiquement, si on décale un peu nos. Nos bords, on est certain, on n'aura jamais un point qui sera sur le bord de d'une case. Si je les décale un peu, vous voyez si je me mets sur le le bord de cette case là ? Ben en fait, je suis sûr que je serai pas en même temps sur le bord de la case bleue. Ouais je veux dire je serai tout le temps à l'intérieur d'une case, c'est juste ça que tu veux. On se on se débrouille pour que la chaque fois n'importe quel point que je prenne dans mon espace il soit à l'intérieur du casque et comme ça on peut prendre la valeur qu'il a dans cette case là et s'en sortir comme ça. Ouais. Dans le cas par exemple dans le virement par un labyrinthe, est-ce que c'est possible que la tuile sorte un peu de. Notre environnement ou enfin de faire ça directement, parce que si tu connais déjà le le Tiling de ton environnement justement, tu vas faire en sorte de de résoudre ton problème de manière tabulaire exactement ce qu'on a fait précédemment. Après si tu connais pas la forme exacte de ton labyrinthe, oui tu tu fous ton timing. Il y a en effet des cases qui vont mal tomber, mais c'est pas très grave en fait. Tout ce qui nous intéresse c'est que quel que soit l'état que je prends dans mon espace, donc c'est un espace 2D, c'est très facile, on a des casques quoi, quel que soit mon espace. Eh bien, je sais que j'aurais une valeur retenue à un moment donné parce que je suis passé par là ou pas. Ok donc si elle est un peu décalée, bah c'est pas très grave. Si elle a une case dans un mur bah tant pis, ta case mémoire elle sera jamais utilisée parce qu'on va jamais aller dans cette case là parce que c'est impossible mais c'est pas grave. Voilà. Et parfois ça peut mettre quand même plus intéressant de faire. En tiling un peu grand sur un labyrinthe, je sais faire un réseau de neurones qui certes servira tout le temps, mais en fait t'es pas du tout sûr qu'il arrivera à résoudre le problème OK ? Donc là c'est l'idée du timing time connu hein, ça c'est le mot clé à retenir. Et donc pour chaque case dans mon dans mon timing, là vous voyez j'ai 2. Enfin j'ai 2 séparations en 4 donc ma table de valeur elle aura une valeur. Ouais donc on retient donc nos paramètres. On devait parce que ici on on écrit comme ça ce sera 4 valeurs. Et donc si et donc notre notre État, hein ? Notre État X de S et Ben ici on est à la fois dans la taille une et la 4, donc notre État on va activer la case une à la question. OK. Du coup Monsieur, c'est pour représenter cet effet de de profondeur en 3D là le fait qu'on fasse 2 cases de de quadrillage différents, ouais non. C'est parce que si jamais j'avais fait un seul quadrillage et que le poisson il était au milieu parce que compliqué, OK ? Plus le fait que on va pouvoir avec du Time coding, tu vois tu vas pouvoir différencier. Bah peut-être que la case une elle va retenir un truc global sur le sur ce qui se passe en haut en haut à droite. Et peut-être que la 4 et là 5 elles vont avoir un comportement différent. Ça pourrait d'avoir un comportement, un comportement pardon différent entre cette case là et cette case là. Alors qu'en soit ils sont toutes les 2. Une tu vois, mais du coup c'est pas dit. S'il était totalement Dieu, il serait placé par exemple le poisson sur le trait vertical. Ouais, par exemple, tu pourras le mettre là ou parce que du coup t'as t'as un changement brutal, un changement brusque de. Un changement brusque de comportement vu que ta valeur échange entre les 2. Sauf que du coup là comme j'ai j'ai le discontinuité mais comme j'ai une case en plus ici qui recouvre, et bah il y aura quand même une continuité sur la feature 4 qui fait que ton modèle va se comporter un peu mieux. Pour pouvoir échantillonner exactement, c'est pour lisser un peu le fait que ce que discrétiser un espace comme ça, c'est quand même très brutal quoi. Il y aurait-il y aurait une limite très fine entre 2 États et il y aurait. On lit un peu parce que entre les 2 à la transition et Ben on partage toujours dans la feature. Là il y a dans le 4 il reste activé donc quand je vais activer quand je vais utiliser ce truc là bah mon paramètre 0 5 là il va rester activé quand à la transition. Donc ce que je suis en train d'apprendre entre 0 j'ai moins un et 4. Moi j'aurais un énorme saut entre les 2 et ça va mal se passer. Okay sinon je te rembourserai, ça permet de lisser un peu ces. Donc là on estimateur de ma value function c'est. Le produit scalaire entre WX. Et donc on a un 5 ici +0 5 ici, ça me donne 2 Vous voyez, ça me permet de faire des monsieur de notre matrice de points. W, c'est la matrice de points, ce qui correspond en fait dans le cas du tabulaire R qu'on vous a précédemment à ma table de valeur de OK, c'est juste ici comme on fait de la de l'approximation de value function. Eh Ben on on l'écrit sous forme de points. Donc le là je vous ai résumé un peu les les idées clés pour quand est-ce qu'on utilise du talc ? Cody OK. Ce premier point, c'est quand même très efficace. On sépare notre environnement, notre, notre espace en les cases précises, quoi, et l'inférence. Elle est immédiate. Vous savez, vous êtes-vous savez quelle valeur vous avez dit OK ? Donc c'est quand même efficace souvent. Ça, ça généralise plutôt bien, juste grâce au fait qu'on a des des cases qui se qui s overlap. Toc toc, Toc Toc. Ouais, parfois on peut même le le combiner avec des modèles linéaires et des réseaux de neurones et ça peut, ça peut marcher très bien. Je sais pas si vous vous rappelez du coup là dans dans le TP précédent on avait vu qu'en fait l'espace ici paraît très bien entre entre les 2 ici. Donc si on a un talk coding en séparant bon endroit, pas un modèle qui apprend ça pas fonctionné très bien. Euh bah ça se passe donc ça ça ? C'est très efficace en terme de mémoire et ça converge très vite parce que c'est c'est très simple quoi comme modèle. Mais bon, gardez à l'esprit que. Il y a quand même un truc difficile dans lequel coding c'est bah comment est ce que vous faites votre séparation ? Et donc en l'occurrence quand vous connaissez le problème comme ça c'est facile de faire la séparation ici. Mais en pratique quand vous arrivez arrivez sur un nouveau problème, vous savez pas quel quel séparation il peut faire. Donc c'est pour ça vous allez faire on part. En pratique vous allez faire des tiles qui seront différents et et aléatoires et donc là il faut être un peu chanceux pour tomber sur le bon. Et une fois vous l'avez fixé c'est pas évident de le changer. Et d'une certaine manière, on va voir aussi que les réseaux neurones peuvent approximer ce genre de ce genre de choses. Donc ça va coûter plus cher mais dans l'idée ils peuvent aussi le faire donc parfois il vaut mieux rester avec des modèles style raisonnement ok donc c'est c'est comme d'habitude il faut toujours connaître un peu les les peu les les donc là vous allez voir tout à l'heure. Du coup le TP dépêche le TP ça va être. On va reprendre cette cet environnement là et on va utiliser un réseau de neurones cette fois on va coder un réseau de neurones, on va l'optimiser et on va voir si on arrive à trouver la bonne police ici sachant que la dernière fois on l'avait fait avec du tel coding et ça marchait très très rapidement. Ouais je remettrai le lien du du du TP de la dernière fois si vous voulez le le signer avant. Donc cette fois donc juste avant on avait des casques comme ça hein ? Maintenant quand on utilise un réseau de neurone, donc là en entrée c'est mon x, c'est mon y d'accord ? À quoi va ressembler l'activation de mauvaise neurones ? Bah ça va ressembler à des Trucs comme ça quoi. Ok, en pratique ça c'est le genre de truc à quoi peuvent ressembler des réseaux de merde ? Vous voyez en fait quand quand on utilise des relues, notamment des relais, vous savez si les fonctions. Il coupe en dessous 0 là en fait les relues justement, elles permettent de faire des des brisures comme ça. Donc en fait avec des réseaux de neurones on peut arriver à faire des genres de case. Et ça en fait, vous pouvez l'apprendre quoi ? La case elle peut bouger, elle peut-elle peut changer, et cetera. Donc c'est pour ça que ça peut être. On peut arriver un peu, d'une certaine manière, à faire la même chose avec. Ok. La suite ? On se rappelle update pour le la temporelle différence learning c'est bon pour tout le monde. On a notre gradient mais on remplace par la différence temporelle. Donc juste avec la différence St +1 OK. Hop, on a peut-être nos paramètres de cette manière-là. Versus. Versus le paramétrer date pour le pour la version Monte Carlo ou pull learning cette fois. On est of policy, donc on prend plus la décision qu'on a prise, mais on prend le Max. Ouais, ça on l'a revu tout à l'heure. En plus ça devrait être, ça devrait être clair. Ouais. C'est la même chose tout à l'heure. Donc là. On est sur sarenza. Vous vous rappelez samsa faire ça c'est of of policier, un policier ? Ouais, on police un policier, on le voit, on le voit ici. Parce que on utilise la valeur réelle. De la décision qu'on a prise, on n'utilise pas le armax comme on a vu juste. Toc toc toc et là on on update notre estimateur exactement comme ça. De la même façon, sauf que cette fois, on a. L'utilise les gradients d'un réseau de neurones, OK. Hop. À chaque fois je vous mets les les algos mais parce que c'est vraiment ça qui est important en fait en réel. Vous avez des dizaines de dizaines d'albums particuliers, vous avez des librairies qui implémentent plein d'infos particuliers. Et en fait enfin, quand vous faites du RP, votre vie c'est ça c'est. J'ai un problème, je connais, je connais mes espaces, et cetera. Quel algo je vais utiliser ? Et en fait du coup faut que vous puissiez voir un algo comme ça pour vous dire Ah bah en fait là je comprends ce que fait cet algorithme, ce que ça a marché, ce que ça va pas marcher et ce que je vais utiliser, ce que je veux pas utiliser. Ok ça vous maîtrisez ce que vous faites et vous pouvez développer une infection sur ce qui va fonctionner ou pas. Voilà donc là, ça, ça toujours pareil. Ouais donc là cette fois on appelle ça expliquer ça, mais c'est juste que on fait la somme sur les actions possibles donc on a une meilleure estimation. Réel OK, avec la fameuse qualité. Et pareil pour le pour le cul NU, sauf que c'est 3, on est un policier, on prend le Max. Avec moins de. Ok. On passe sur la dernière partie du cours avant. Ah non, il reste 2 de. Le. Ouais, on va rapidement passer là-dessus, c'est. Pourquoi est-ce que, enfin, on va revenir un peu sur la manière de stocker les données ? On a vu que quand on quand on faisait un épisode, on avait déjà plein de transitions en stock. Et on allait pouvoir apprendre à partir de ces solutions là. Et donc là il y a une notion qu'on va reprendre tout à l'heure qui s'appelle le replay buffer je sais pas si ça vous parle. Donc le replay buffer c'est ça c'est je retiens. C'est je trouve, c'est assez explicite, c'est on retient tout ce qui s'est passé avant et on va pouvoir se dire après coup Ah mais quand j'étais dans tel étalage, il fait telle action et j'ai eu tel truc. Donc peut être que ça va m'aider pour optimiser mes paramètres plus tard. Et donc dans l'idée, euh, dans l'idée en R, il y a un peu de ça. Et donc quand on fait du Batch RL et Ben on va. Explorer un maximum un environnement. On récupère plein plein de données et ensuite on apprend, mais sans réutiliser l'environnement. C'est-à-dire ? On considère que les données qu'on a récupérées sont suffisantes pour. Pour pour apprendre de. Ok donc là ce qui est marqué ici tu vas y a pas de y a pas d'interaction réelle avec l'environnement pendant le l'apprentissage. Et donc ça, c'est très utile pour beaucoup de scénarios. Pensez à des véhicules autonomes, notamment les véhicules autonomes. Ils récupèrent plein de données tout le temps quand il roule. Et en fait, ils sont traînent, mais de manière offline. S'entraîne à la fin quoi, vous voulez pas que votre autonome il s'entraîne et qui reste de l'exploration pour quand vous êtes sur la route ? Donc c'est ça. Donc ça s'appelle offline RL, offline learning, c'est ça ? C'est je retire tout ce qui s'est passé, je retiens toutes les transitions et j'essaie d'apprendre à partir de. Ok. Donc en effet en fait ce qu'on a fait avec le Nutella et le politisation, c'est ça. Parce que simplement dans notre cas, on connaissait déjà toute tradition à l'avance. Et on fait notre apprentissage à partir de ça. Donc ça revient à retenir tous les États qu'on a traversés, même si ces États continue, qu'on n'a pas tous les États possibles, mais on a quand même un ensemble assez conséquent d'État qu'on a traversés et on peut faire descendre, tirer parti de la. Donc là, c'est les motivations. C'est ce que je vous ai dit, hein ? Les vérificateurs, on est content de de pas les faire apprendre sur le terrain. On peut récupérer des données qui ont été collectées plus tard, ça veut dire que si vous êtes un chercheur et que vous voulez faire apprendre un robot à prendre un objet. Bah en fait c'est super dur comme problème qu'il faut avoir le robot, il faut prendre les 2, il faut faire du scan, et cetera. Et donc c'est pour ça qu'il y a d'autres équipes de recherche qui elles sont spécialisées dans la production de données et donc qui vont vous donner un dataset tout fait de d'interaction avec un robot. Donc ça vous permet de faire un apprentissage même si vous avez pas le robot qui coûte super cher avec vous. Et donc vous pouvez faire pareil. Le cadre autonome est assez probant je trouve. Je vais réutiliser toutes les données de vos utilisateurs pour réapprendre de nouveaux modèles encore meilleur. C'est exactement ce que fait Tesla. C'est la raison pour laquelle ils ont des ils ont déployé partout, c'est on récupère un axe de données pour être un maximum meilleur et c'est la raison pour laquelle j'ai pété le meilleur aussi, c'était les premiers de te récupérer plein de données des utilisateurs. Donc ils ont encore plus de données pour être encore meilleurs pour répondre aux OK. Et c'est pour ça d'ailleurs que j'ai cité est gratuit, c'est Claude et Mistral, et cetera. La raison, c'est la raison pour laquelle vous leur donnez de la valeur, c'est parce que vous leur donnez de la donnée permettait de d'entraîner leur modèle. Et en finance c'est pareil, si vous voulez faire apprendre un modèle à placer des des fonds sur un marché, vous voulez pas qu'il apprenne online. Vous allez apprendre sur des données que vous connaissez déjà et ensuite, une fois qui marche bien, le déployer. Bon, l'histoire, prédiction, je pense qu'on en a déjà assez parlé. Euh. Donc là c'est la même chose, sauf que on se capte dans en sline. Et donc on a notre notre dataset avec toutes les transitions, OK. Et où on on optimise notre W pour que ils puissent faire le converge dans la bonne valeur et on optimise simplement le le la MSE quoi sur. Hop. Et donc encore une fois, 2 manières de faire, soit je le fais en temporaire de France. Soit je le fais en mode Monte Carlo. Toujours les 2, les 2 qui sont. Qui sont, comment dire, mis en contraste ? D'un côté, j'utilise le le dire en entier parce que je connais l'épisode, de l'autre côté, je j'apprends à chaque étape avec toutes les traditions politiques. Hop donc là c'est la manière de l'écrire, je pense que cette slide est résumée à peu près bien. Qu'est ce qu'on fait ? Qu'est-ce qu'on fait pour dans le cas du le Square Monté-Carlo et du liste Square Tom différence ? Où là on va avoir vraiment le lien versus ? L'équation de l'Allemagne avec la différence ? Peu les les OK ? Encore une fois, c'est l'application au au cas linéaire. Où les choses s'écrivent de manière plus simple, c'est pas forcément très important. Dans le cas du contrôle, cette fois on est online, on peut quand même faire du Batch RL parce que on va. Dans le cas du contrôle, on va retenir tout ce qui s'est passé avant et même si on est en train de enfin, on est en train de continuer à interagir avec son environnement. C'est pas pour ça qu'on peut pas retenir dans notre dataset tout ce qui s'est passé juste avant et et l'utiliser pour améliorer notre politique. Donc on utilise typiquement si on est sur du on Ball ici. Si on est sur un policier, on peut utiliser même le la policier précédente pour apprendre de nouvelles choses parce que ça reste des transitions qui ont du sens dans le dans, dans le top on dit, je suis passé un peu vite là-dessus. Et c'est pas son nom. Au moins 01h15 pour le TP je pense. Donc on en va. Ok. Ouais, je propose de terminer là-dessus, je vous lance sur le TP et on fera la pause pendant le TP, ça vous va ? Comme ça, je travaille un peu plus pour eux, normalement, c'était de se passer. Ok, donc maintenant je vais vous parler de l'architecture un un peu importante, et c'est celle qu'on va, celle qu'on va implémenter tout à l'heure, OK ? Donc ça s'appelle Deep the Network DUM OK et c'est et ça a été publié par bulletin dans je sais plus quelle limite on verra. C'est à dire une des plus, une des premières, un peu. Brève évolution en ROA vraiment réussi à faire à faire jouer des Jeux Atari à à des réseaux de neurones ce que ce dont je vous parlais tout à l'heure. Donc le DN il prend-il utilise des réseaux de neurones. Qui a dit quelque part ça veut dire qu'on utilise des réseaux de neurones ? Encore une fois ça donne ce genre de chose, OK ? Et l'architecture, c'est la suivante. On a notre agent avec notre Deep Nord Network. Euh, on a nos États en entrée, hop. Et en sortie en fait, au lieu de donner directement la valeur parce que vous rappelez que normalement la, c'est quoi prend d'entrée l'État et l'action. Eh Ben en fait dans le dans le DQN. On l'écrit différemment, l'entrée de mon réseau de neurones, c'est l'état seulement. Et en sortie il nous donne la value mais pour chaque action. Ok, vous voyez la différence ? Si on avait fait un un juste Cue Network, donc pour approximer la fonction de Q hein ? En entrée on aurait dû avoir State action comme ça hop et en sortie uniquement un neurone qui nous donne la value function. Enfin la la la l'espérance du gain on est d'accord. Dans des quand on l'écrit différemment, on l'écrit, comme ça on donne l'État en entrée. Et en sortie nous donne la probabilité pour chaque, la pardon la, la valeur pour chaque. Ce qui revient au même, hein ? Ça veut dire que si vous cherchez la valeur pour l'état S et l'action a Ben vous faites-vous faites passer votre à l'aise dans votre réseau et vous récupérez la valeur pour l'État a que pour l'action a que vous voulez. D'accord, ça revient au même. Simplement, l'architecture est très différente, OK, et intuitivement pourquoi c'est intéressant ? Parce qu'en fait ça vous permet de. Enfin, en général pour, vous avez quand même beaucoup d'actions possibles, et la, la prédicteur de votre action de la valeur de votre action. Il va utiliser le même genre d'information pour toutes les pour, pour toutes les actions. C'est à dire peut-être que ici j'aurais. Ok en fait le l'optimum pour quand je suis dans l'État dans l'état s c'est d'aller tout droit. Et bah c'est quand même plus facile de dire Bah je sais qu'il faut aller tout droit et donc j'ai prédit une bonne value pour la la fonction aller tout droit et une mauvaise value pour la version reculer. D'accord, c'est plus facile que de partir de 0 et de dire à partir de je suis ici et allez tout droit, tu me prédis la valeur. OK. Bon, je vous, je vous laisserai méditer ça plus tard. Et donc dans l'idée, il y a une petite subtilité ici, le le Deep Network, il est implémenté de manière un peu, enfin un peu surprenante par rapport à ce à quoi on pourrait s'attendre, OK ? Ouais. Voilà et donc là ma Ford pass justement là il s'est trompé, c'est uniquement uniquement l'état ici et il me semble justement plus de sa OK je. 38 là en gros j'ai des slides sur la bac propre. Ouais c'est chaque récompense associée en fonction de l'État et la fin de l'action qu'on prend en fonction de l'État dans tu peux refaire donc là dans les dans le réseau en l'État ouais et du coup le. QSA, c'est chaque récompense associée à une action en fonction de l'État dans lequel on la la. L'espérance du gain, c'est ça l'espérance du gain, ça c'est ici, ce truc là. Donc là, on l'écrit de manière un peu différente, on l'écrit dans des cubaines, on dit. Euh, on va dire. Écrire tu donc que je paramètre par lever de s ? Égal à l'ensemble des. L'ensemble de la vraie valeur de plus de SA 0. Tu le laisses AM ? Ouais, c'est clair comme ça. C'est la dernière coupe qu'on en fait exactement. La s c'est ma première couche et ça c'est la dernière couche. Donc c'est on est d'accord que c'est ça revient au même. On peut avoir le même information, simplement on l'écrit, différemmentdans.de droit. Ok donc du coup là il est 40 ? Il nous reste 01h20, j'ai des slides sur la bac pro. Je pense que vous avez déjà vu le lac propre et il y en a qui veulent rapidement en réfléchir. Un petit rappel donc je vais passer rapidement dessus. Donc notation, OK. Est-ce notre État ? Ok. X c'est l'État caché. Ah c'est mes poids. Ok, c'est le poids de I vers J et B Paris. Donc là vous pouvez regarder sur les slides directement. Et donc j'ai une noce où on a l'estimateur YK, et j'ai ma valeur réelle, OK ? Donc. Je fais ma dérivation, OK ? Ouais c'est ce genre de trucs vraiment faut regarder, faut regarder sur les slides directement mais. Vous allez dériver votre fonction et avec la avec la. Le comment dire ? Le. La propriété fondamentale que vous voulez connaître, c'est la chaine roll. Quand on fait la dérivée, donc la dérivée de ma loss ici OK. Dérivée de Edge sur DB. Je peux utiliser la chaîne roule pour dire Ah mais en fait c'est la, c'est la dérivée sur YK. Et hop, je remets le y a ici. C'est comme si on faisait une simplification là hop, et ça ça revient même OK. Et en faisant ça en fait, vous pouvez séparer la lose sur chaque étape de calcul de votre de votre zone neurone. Exactement. Et donc grâce à ça, vous pouvez écrire. Que la dérivée de la Los, donc la dérivée de la lose complète. Par rapport à mes paramètres, parce qu à la fin c'est ça qu'on veut, on va vouloir dériver par DB là. Et et et ensuite par rapport à a mais là ici on fait juste les gradients pour B. Eh Ben on écrit on. On les craint que je roule quoi ? Ouais, C'est pourquoi je sais pas, mais il y a plein de celle de la manière de l'écrire, comme ça donc peut-être. Mais ouais. Ouais, je sais pas si j'ai passé vite là-dessus. Si ça vous intéresse pour regarder et il y a, il y a d'autres sources aussi. Sur la date je pense pour mieux fait que ça. Ouais c'est le même algorithme, c'est c'est l'algorithme initial de la bague propagation c'est pas spécifique. Non ça c'est juste OK. J'ai un réseau de neurones, j'ai une entrée, j'ai une prédiction et j'ai une grande trousse. Comment je fais pour calculer un gradient de mon modèle par rapport à ma sortie ? En pratique, vous faites juste Lost en backward avec Victor, c'est pas facile. Et c'est bien de savoir pourquoi ça marchait quoi et pourquoi est-ce que Ouais, pourquoi ce qu'on a besoin ? Euh ouais donc là il faut regarder sur les têtes, ça vous amuse. Et sinon sur Internet, y a plein de ressources, c'est assez bien. Pour parler de ça, moi j'ai, j'ai passé. Et ensuite ? Donc une fois qu'on a fait notre bac propre on trouve les gradients par rapport à chaque paramètre du modèle. On a vu, on a on a rétro propagé donc on trouve les gradients de de la sortie par rapport à mes baies par rapport à mes a et ensuite pendant la pendant le update bah. On prend les gradients de baie, on les retire, on les retire pour, on la voiture au paramètre. Pour que, au coût d'après, la sortie soit un peu plus faible que l'os diminue petit à petit. Ok, alors ? Tels que rapidement sur des stratégies d'optimisation pour les réseaux de neurones. Ça, c'est pas mal envers à connaître. Quand vous commencez à rencontrer des problèmes d'entraînement sur zone neurone, bah vous pouvez penser un peu à chacun de ces trucs, donc le premier c'est normaliser vos inputs. Ça, non. Vous avez appris à le faire. On fait beaucoup en mail de manière générale, avec des réseaux de neurones. Ça, ça marche aussi. Ok. Changez votre architecture. Ok, parfois il faut plus de couches cachées. Ou des coûts cachés du plus grande dimension ? Changez vos fonctions d'activation. Ça super important par défaut, on prend des relais partout, mais si jamais vous avez un problème où en fait vous avez envie d'avoir une fonction qui est lisse à la fin ? Bah les relis c'est pas du tout le meilleur choix, parce que le relu c'est pas c'est pas. D'accord, mais c'est pas. Dérivable. C'est y a une brisure quoi, si vous voulez, c'est pas, c'est pas lisse. Bah c'est un exactement. Et donc vous allez peut-être préférer avoir une sigmoïde ou avoir une U ou de y a des différentes. Y a plein de fonctions de situation qu'on peut utiliser, mais si vous voulez un un modèle qui est à proxim, quelque chose de lisse, bah on pourrait éventuellement penser à. Mettre des fonctions qui sont ici. Ensuite la régularisation, OK Régularisation, vous pouvez faire du Drop out, vous pouvez mettre du drop out dans vos layer. Vous pouvez aussi faire régénération sur. Le poids enfin sur vos points, c'est-à-dire éviter d'avoir des poids trop forts que souvent un poids trop fort ça veut dire Vous allez avoir des gros pics de prédiction sur certains assets et souvent c'est les risques. Vous êtes en train de faire lever City OK donc mettre une régulation l 2OL un lasso est riche je sais pas si ça vous parle et l'ascenseur tout ça. Donc ça, ça peut servir. Euh optimizer par défaut, mettez Adam pareil pareil, parfois sgd ça marche mieux. Voilà franchement remettez tout en question, c'est pas parce que vous avez un template de optimizer égal Adam, ensuite je mets 1E, j'en sais rien tu vois, voilà posez vous la question, est-ce que ça vaut pas le coup de de d'essayer quoi changer et la fonction coup ? Bah nous par défaut on est sur la messe, mais ensuite faut voir si vous avez un problème de classique, et cetera, on est pas en train de faire un cours de ml donc juste posez vous la question, c'est quoi ma fonction coût ? Pourquoi est ce qu ? On pourrait m'en mieux marcher ou moins bien marcher. Ok bon désolé pour le le je retiens un peu rapide donc je reste 01h15 pour finir. Pour finir, le TP donc. On se base en gros, on va réimplémenter le papier original de DQN, OK ? Hop, j'essaie de vous le montrer. De la recherche du coup, ça vous fait un peu un. Un. Une première confrontation, un papier de recherche qui lui est super précis et à peu près bien écrit. Mais vous inquiétez, vous êtes pas obligé de lire cela de occupé. Mais donc, comme je vous l'ai dit, vous allez réimplémenter. Dès que tu vienne dans le cadre du Montagna, OK. Et donc je vais vous mettre le lien du TP ce que je mette si je me sur team ça vous va ? Écoute, ouais. Alors ? Normalement vous avez encore le lien vers vers le TP de la semaine dernière sur avec le le type codé. Donc pour ceux qui ont pas fini le type Coding, vous pouvez essayer de de regarder un peu aussi vous allez pouvoir faire la comparaison. L'idée à la fin, c'est que vous puissiez comparer les 2 méthodes, voir bah qu'est ce qui marche mieux que ce qui marche bien, et cetera. Peu les les normalement vous devez avoir reçu le lien. Ok alors rapide présentation du TP alors donc on se place dans ce cas là ? Là vous vous faites un camp fiche sur ajouter quelques trucs, on va créer un replay buffer comme ça je le crée pour vous donc vous avez pas vous en vous en occuper. Mais donc dans l'idée, le le replay de fort c'est quoi ? Simplement on va, c'est une pile dans laquelle on va stocker un ensemble de États. Action reward Next the et le fait que ce soit un épisode terminé ou pas, OK, c'est exactement ce qu'on vient de voir dans dans le cours. Donc notre notre replay buffer il va servir à ça OK ? Donc ça va nous permettre de retenir toutes les étapes qui sont passées précédemment. Ensuite ? Que donc là, c'est rapidement un point pour. Justement on va, on va utiliser un algo qui ressemble à du co Learning. Je dis bien qu'il ressemble, mais dans ces gens d'ARGO, vous savez, on fait diminuer petit à petit le le l'epsilon de d'exploration. OK, on épilé ingrédient et au fur et à mesure de l'entraînement on diminue notre epsilon pour explorer de moins en moins et donc exploiter de plus en plus qu'on est en train d'apprendre. Ok et donc là je vous, on essaie de. Visualiser un peu ce psylone et comment est-ce que ça va marcher ? Ensuite, on va implémenter. Euh, ça, vous allez commencer à avoir des choses à faire, on va implémenter notre notre DQN. Ok, vous pouvez, vous pouvez aller voir le, vous pouvez aller voir le papier en question, OK, et du coup, ce qu'ils le mettent exactement. Tac, TAC TAC. J'ai remis, j'ai remis la citation, j'aurais dû le mettre en en entrée, mais en gros, dans le papier, ils mettent ça. C'est ce que je vous ai expliqué tout à l'heure. La sortie, c'est les les cube alue. Fred dit sur sur sur toutes les actions possibles et il prend bien en entrée. Euh. Ils utilisent une une, une sortie séparée pour chaque action possible. Donc faudra faire attention quand vous allez coder votre votre réseau de neurones ici. À bien faire le bon nombre de sorties, bon nombre d'entrée quoi. Ensuite vous allez coder le l'Epsilon crédit ici. On a instancié le modèle, on calcule la Los OK. Et ensuite, on va pouvoir l'entraîner et visualiser un peu. Donc c'est pas c'est pas très long mais j'ai essayé de de vous laisser plus de choses. Donc je vous laisse, je vous laisse vous lancer et n'hésitez pas. Et jetez un coup d'œil aussi sur. Sur le TP précédent. Avec avec le Titanic. Peu les les oui. C'est quoi ? C'est même State. On va, on va le faire complètement à côté, c'est très bien. Donc là imaginons que mon réseau de neurones il prend en entrée un vecteur de dimension 5. Ok ? Donc là juste un petit print pour voir le lecteur qu'on a d'accord et. Toc. Voilà dans la différence que ça fait, on s'en fiche, OK ? Et. Même Portugal à. Donc là on va se coder un petit, un petit réseau neurone rapidement. Avec. Ok là on a un réseau de neurones avec 5 features en entrée, il sort une feature en sorte OK et donc là on va voir ça print. Nette. De. Je pense. Là, on va voir. Ça marche, OK ? Donc là on a raison neurones on a 5 futurs en entrée. Et on a. Une feature en sortie donc quand on fait modèle de notre feature en entrée, on a bien le OK ? Sauf qu'en pratique quand on est en torch ça ça arrivera jamais parce qu'on voit toujours faire de l'influence par batch. Ok, donc en pratique notre temps sort, il va avoir une dimension de plus donc ça veut dire par exemple il va être comme ça. Ok