Université de Montréal

PHY 3075 – Modélisation Numérique en Physique

Algorithmes évolutifs : l'orbite de ρCrB

par :
Éric Pfleiderer
20048976

18 avril 2019

Résumé

L'objectif de ce laboratoire est l'exploration des algorithmes évolutifs et de leurs applications sous deux contextes : la diffraction en deux dimensions et l'optimisation d'un modèle d'orbite képlérienne à 6 paramètres à partir d'observations de vitesse radiales. Le premier contexte sert à faciliter la visualisation de l'espace paramétrique et à valider l'algorithme. Ensuite, on explore l'influence de 4 hyperparamètres, N, p_c, p_m et p_s, sur les solutions produites par l'algorithme. Additionnellement, on explore 3 types de mutations adaptives : basées sur l'objectif, basées sur la distance et non-uniformes. Ces stratégies sont mises en compétition pour déterminer le type de mutations employées lors de l'optimisation finale des données. On trouve que la performance des stratégies est comparable à l'exception de la stratégie basée sur la distance. La stratégie choisie pour l'optimisation du modèle est celle basée sur l'objectif, qui demeure plus stable lorsqu'on considère l'ensemble des solutions produites. On suggère le vecteur solution final $v^f_{sol} = (39.80 \pm 0.04, 85 \pm 50, 2.1 \pm 0.4, 0.110 \pm 0.005, 63.9 \pm 0.6, -47 \pm 1)$ pour l'ensemble de paramètres P, τ, ω, e, K, et V_0.

1 Introduction

Une quête perpétuelle de l'analyse numérique est sans doute le problème de l'optimisation. Plus spécifiquement, comment peut-on éviter de converger vers un minimum local lors d'un ajustement de données ? Existe-t-il des techniques pour augmenter la probabilité de converger vers un maximum global ? Peut-on s'inspirer de phénomènes biologiques pour développer des algorithmes dotés de comportements complexes ? La pertinence de ces questions prend de l'importance avec la popularité grandissante des réseaux neuronaux qui basent leur entraînement sur de telles techniques.

L'objectif de ce laboratoire est donc l'exploration des algorithmes évolutifs et des techniques de mutations adaptives. On applique d'abord un algorithme évolutif sur un problème de diffraction à deux dimensions afin de faciliter la visualisation de données et la validation de l'algorithme. On s'intéresse ensuite au contexte d'application principal, soit l'ajustement d'un modèle d'orbite képlérienne aux systèmes binaires $\rho CoronaBorealis$ et $\eta Bootis$. On emploie initialement des mutations de tailles constantes et on étudie l'influence des hyperparamètres N, p_m, p_c et p_c sur la convergence. On introduit ensuite 3 types de stratégies de mutations adaptives ; basée sur l'objectif, basée sur la distance paramétrique et non-uniforme. Les différentes stratégies sont mises en compétition pour déterminer laquelle est la plus prometteuse. Finalement, on procède à la tâche principale ; l'optimisation des paramètres du modèle à l'aide de données décrivant le système $\rho CoronaBorealis$.

2 Théorie

2.1 La diffraction

La diffraction décrit le comportement d'une onde lorsqu'elle rencontre une ouverture. Elle est expliquée par le principe de Huygens-Fresnel, qui suppose que chaque point d'un front d'onde se comporte comme une collection d'ondes. On introduit l'équation 1, qui définit la relation entre l'intensité normalisée d'une onde et sa position en deux dimensions. La figure 2, quant à elle, affiche un patron de diffraction en 1 et en 2 dimensions spatiales. L'intensité est normalisée de sorte que $I_{max} = 1$ à l'origine.

$$\frac{I(x,y)}{I_0} = \left(\frac{sin(x)}{x}\frac{sin(y)}{y}\right)^2 \tag{1}$$

On estime l'erreur d'une certaine solution $v_{sol} = (x, y)$ par l'équation 2, qui est simplement une différence entre le maximum global à l'origine et la valeur de notre solution.

$$e = I_{max} - I(v_{sol}) = 1 - I(v_{sol}) \tag{2}$$

La recherche du maximum global est compliquée par la présence de plusieurs maxima locaux dus aux termes sinusoïdaux dans l'équation 1. L'équation de diffraction exhibe une des vulnérabilités

de la descente du gradiant traditionnelle, qui converge vers l'extremum local le plus près du point de départ. Il s'agit de conditions idéales à l'application des algorithmes évolutifs.

2.2 L'orbite képlérienne

La méthode principale d'identification des système binaires est par étude du décalage Doppler causé par le mouvement orbital. Ce mouvement orbital est décrit par l'équation 3, où V est la vitesse radiale observée, K est l'amplitude de vitesse, e est l'excentricité de l'orbite et ω est le déphasage.

$$V(t) = V_0 + K[cos(w + v(t)) + e\ cos(\omega)] \tag{3}$$

L'anomalie de la vitesse projetée v est obtenue à l'aide de la relation 5, où E est l'anomalie excentrique.

$$tan\left(\frac{v}{2}\right) - \sqrt{\frac{1+e}{1-e}}\ tan\left(\frac{E}{2}\right) = 0 \tag{4}$$

Finalement, l'anomalie E est reliée au temps t par l'équation de Kepler (voir équation 5), où P est la période et τ est le temps de passage au périhélion. L'équation de Kepler définit un problème de recherche de racine non-linéaire, qu'on résoud par bissection.

$$E - e\ sin(E) - \frac{2\pi}{P}(t - \tau) = 0 \tag{5}$$

Ensemble, les paramètres P, τ, ω, e, K et V_0 définissent un espace paramétrique à 6 dimensions. C'est cet espace, où chaque point peut être décrit par un vecteur solution $v_{sol} = (P, \tau, \omega, e, K, V_0)$, que l'algorithme évolutif tente d'explorer.

3 Méthodologie

3.1 Analyse numérique

Les algorithmes évolutifs sont des procédés itératifs et stochastiques qui sont souvent inspirés de la génétique. Ils possèdent généralement les phases suivantes ; l'initialisation, l'évaluation, le classement, la sélection, la reproduction et le remplacement. On introduit ici la notion d'hyper-paramètre ; un paramètre qui concerne l'algorithme plutôt que le modèle d'orbite képlérienne. Ces hyperparamètres en questions, N, p_c, p_m et p_s, sont introduits dans les prochaines sections. Leur objectif est de contrôler le comportement de la population. L'implémentation particulière de l'algorithme employée durant ce laboratoire est disponible sur le repos GitHub de l'auteur[3].

Initialisation

La première étape consiste à définir une populations de plusieurs vecteurs solutions v_{sol} initiaux qui déterminent les points de départ de l'algorithme. Un choix judicieux de chaque v_{sol} demeure important et peut être informé par la définition des paramètres et par une étude des données. Les intervalles choisis et employés sont affichés au tableau 1.

TABLE 1 – Intervalle pour chaque paramètre à optmiser. Les paramètres sont initialisés et contraints à ces intervalles durant l'optimisation.

paramètre	**intervalle**	**unités**
P	$200 \leq P \leq 800$	$J.D.$
τ	$t_0 \leq \tau \leq t_0 + P$	$J.D.$
ω	$0 \leq \omega \leq 2\pi$	$radian$
e	$0 \leq e \leq 1$	[1]
K	$0 \leq K \leq max(V_j) - min(V_j)$	$km\ s^{-1}$
V_0	$min(V_j) \leq V_0 \leq max(V_j)$	$km\ s^{-1}$

Évaluation

La deuxième phase consiste à définir une fonction objectif pour guider notre recherche, c'est-à-dire qui décrit la qualité d'un vecteur solution. On utilise les données fournises[5] pour calculer un khi-deux, définit par l'équation 6, qu'on minimise en réduisant l'écart entre les observations et les prédictions. Puisque notre algorithme s'applique sur des problèmes de maximisation, on s'intéresse plutôt à l'inverse du khi-deux, qui définit notre fonction objectif (voir équation 7).

$$\chi^2(v_{sol}) = \frac{1}{N-6}\sum_{i=0}^{N}\left(\frac{V_i^{obs} - V(v_{sol}, t_i)}{\sigma_i})\right)^2 \tag{6}$$

$$f_{obj}(v_{sol}) = \frac{1}{\chi^2(v_{sol})} \tag{7}$$

Lorsque la différence entre les observations et les prédictions sont similaires en grandeur à l'erreur associé aux mesures, on termine l'algorithme afin d'éviter la suroptimisation ; le modèle ne doit pas incorporer les déviations dues aux erreurs, mais plutôt le comportement général des données.

Classement et sélection

Lorsque chaque individu est évalué, on procède ensuite au classement et à la sélection. Le classement consiste ici simplement à hiérarchisé les solutions à l'aide de f_{obj}. La sélection est effectuée par le biais d'un paramètre de pression sélective $p_s \in [0, 1]$ qui décrit la probabilité que chaque membre de la population soit choisit pour la reproduction. Une valeur nulle se traduit par un absence de biais, tandis qu'une valeur de 1 implique que seul la meilleur solution courante est reproduite.

Reproduction

Une fois les parents choisis, ils ont une probabilité p_m de se reproduire. S'ils échouent à la reproduction, on observe une reproduction asexuée de chaque parent ; les enfants sont des clones identiques. Par contre, s'ils se reproduisent avec succès, il y a *échange du matériel génétique*. Étant donnée deux parents p_1 et p_2, on déduit le paramètre i de deux enfants c_1 et c_2 par combinaison linéaire (voir équation 8).

$$\begin{aligned} i_{c_1} &= a i_{p_1} + (1-a) i_{p_2} \\ i_{c_2} &= (1-a) i_{p_1} + a i_{p_2} \\ a &\sim U(0,1) \end{aligned} \tag{8}$$

Lors de la reproduction, les enfants générés ont une probabilité p_c de subir une mutation aléatoire. Les types de mutations prennent plusieurs forment et sont explicités à la section 4.3. Puisque les paramètres à optimiser ne sont pas tous de la même ordre de grandeur, on définit un vecteur de mutation v_{mut}, qui définit la taille de pas associé à chaque composante du vecteur solution v_{sol}. Le vecteur v_{mut} est initialisé en moyennant la valeur de chaque paramètre à travers la population et en choisissant une certaine proportion de cette quantité.

$$v_{mut} = c\overline{v}_{sol},\ c \in \Re \tag{9}$$

La mutation non adaptive employée dans les premières sections du laboratoire est définie par l'équation 10, où v_{mut}^i est la i^e composante du vecteur de mutation.

$$c_i^{n+1} = c_i^n + G(0, |v_{mut}^i|) \tag{10}$$

Remplacement

Finalement, il ne reste plus qu'à remplacer la génération précédente par les enfants et continuer à itérer. Par contre, une technique essentielle doit être implémentée : l'élitisme. On préserve artificiellement la meilleur solution à chaque étape, ce qui nous garanti de ne jamais rétrograder durant l'exploration de l'espace paramétrique.

4 Résultats

4.1 Validation des résultats

On s'intéresse d'abord à explorer le comportement de notre algorithme évolutif dans un contexte où la visualisation et l'exploration de l'espace paramétrique est facilitée : la diffraction en deux dimensions. On définit un intervalle d'étude de $X \times Y,\ X, Y \in [-2\pi, 2\pi]$ et on initialise une population de $N = 10$ individus distribués uniformément dans ce domaine. On applique les processus itératifs décrits à la section 3 en évaluant la qualité des solutions à l'aide de l'équation 1. La figure 4 affiche les instantannées de la simulation pour les 9 premières étapes. Le point vert représente la solution optimale à chaque étape, tandis que la courbe verte affiche la trajectoire de la solution optimale de génération en génération. On remarque que l'initialisation aléatoire des positions place la solution optimale très près d'un maximum local, ce qui serait une situation facheuse si on utilisait la descente du gradient traditionnelle. Dans notre cas, une mutation ou un croisement déplace la solution optimale près d'un second maximum local qui s'avère supérieur au précédent. L'algorithme demeure sur cet extremum jusqu'à l'étape 4, où il débute enfin à grimper le maximum global. À ce point, l'élitisme nous garantit de ne pas régrésser ; la convergence continue indéfiniment jusqu'à ce que l'erreur 2 devienne plus petite que l'epsilon de la machine employée. La figure 3 affiche l'erreur correspondant à la simulation de la figure 4 ainsi que l'erreur d'une simulation de 100 générations.

Une fois assuré de la validité de notre algorithme, on cherche à implémenter une seconde version sur le sujet central du laboratoire, soit l'ajustement d'un modèle képlérien décrivant les orbites de $\eta Bootis$ et $\rho CoronaBorealis$. On s'intéresse d'abord à $\eta Bootis$, pour lequel un vecteur solution $v_{sol} = (494.20, 14299.0, 5.7397, 0.2626, 8.3836, 1.0026)$ est fourni [1]. Dans le cas présent, l'espace paramétrique est de 6 dimensions et est difficile à visualiser. On offre plutôt la figure 5, qui montre les observations ainsi que le modèle ajusté pour 3 étapes de la simulations, soit à 100, 250 et 500 générations. On observe bien la convergence désirée ; le modèle se juxtapose aux données et converge vers la solution fournise. On entame alors la tâche d'étudier l'impact des hyperparamètres sur le comportement de l'algorithme.

4.2 Influence des hyperparamètres

L'implémentation de l'algorithme évolutif permet la manipulation de 4 hyperparamètres, soit N, p_m, p_c et p_s. Ces paramètres contrôlent et influence l'exploration de l'espace paramétrique ainsi que le rythme de convergence. Par exemple, on sait qu'une augmentation de la population N entraîne une exploration additionnelle de l'espace paramétrique pour chaque nouvel individu. On s'attend donc à ce que la taille de la population impacte la capacité de l'algorithme à trouver le maximum global. Ensuite, par analogie à la génétique, l'information de population est partagée par reproduction sexuée entre ses membres. L'introduction d'une nouvelle information génétique provient des mutations. On s'attend donc à ce que le facteur p_m influence l'exploration en définissant la fréquence d'introduction de nouvelle information. Ensuite, p_c caractérise la popularité de la reproduction séxuée et asexuée. On s'attend à ce qu'un haut taux de reproduction augmente la convergence vers la solution optimale courante. Finalement, la pression de sélection p_s détermine quels individus sont considérés durant la phase de reproduction. On s'attend à ce qu'une pression trop élevée cause une convergence prématurée vers une solution sous optimale, tandis qu'une pression trop faible diminue la capacité de convergence dans les phases terminales de la simulation.

On compare nos prédictions avec la figure 6, qui montre l'influence de chaque paramètre sur la solution finale. On contrôle la génération stochastique des nombres afin de diminuer la dépendance sur la solution initiale. On remarque, tel que prédit, qu'une augmentation de la taille de la population mène habituellement à une meilleur solution. Ensuite, on constate que l'hyperparamètre p_m doit être restreint à des valeurs $<< 1$. Lorsque $p_m \approx 1$, le χ^2 tend à demeurer constant jusqu'à subir une diminution importante et subite ; l'algorithme est incapable de converger vers l'extremum le plus près de la solution optimale courante et tend à optimiser la fonction objective en trouvant un extremum de meilleure qualité. C'est un comportement recherché en début de simulation, mais lorsque celle-ci mature, on assume que le maximum global est repéré et on désire plutôt imposer un biais pour l'affinement de la solution courante. Quant à p_s, on remarque que nos prédictions sont supportées par cette simulation ; on cherche un compromis entre l'exploration par le hasard et la convergence par sélection sévère. Finalement, la probabilité d'accouplement p_c n'influence pas

significativement la capacité de convergence dans cet exemple. Tôt ou tard, on atteint le même extremum.

4.3 Les mutations adaptives

On s'intéresse maintenant à implémenter plusieurs stratégies de mutations adaptives ; on tente d'améliorer la performance de notre aglorithme et on cherche à comparer les différentes stratégies entres-elles. On introduit 3 types de mutations adaptives, soit la mutation basée sur la fonction objectif, la mutation basée sur la distance et la mutation non-uniforme.

Basé sur la fonction objectif

Le vecteur de mutation v_{mut} est potentiellement modifié à chaque génération par multiplication ou division par un scalaire c. La décision de modifier v_{mut} est effectuée à partir d'une différence entre la valeur de l'objectif de la solution optimale courante v^1_{sol} et celle de la solution médiane v^{med}_{sol}. Si la différence est plus petite qu'une fraction a de v^1_{sol}, on assume que la convergence est trop forte ; on augmente le pas. Par contre, si la différence est plus grande qu'une fraction b de v^1_{sol}, on assume que la population possède une bonne étendue paramétrique et on diminue la taille du pas afin de faciliter la convergence locale.

$$v^{n+1}_{mut} = \begin{cases} cv^n_{mut}, & si\ f(v^1_{sol}) - f(v^{med}_{sol}) \leq af(v^1_{sol}),\ a,c\ \in\ \Re \\ \frac{v^n_{mut}}{c}, & si\ f(v^1_{sol}) - f(v^{med}_{sol}) \geq bf(v^1_{sol}),\ b,c\ \in\ \Re \end{cases} \tag{11}$$

Basé sur la distance

Le vecteur de mutation v_{mut} est potentiellement modifié à chaque génération par multiplication ou division par un scalaire $g(c)$ provenant d'une distribution gaussienne G. La décision de modification est basée sur la distance paramétrique entre v^1_{sol} et la v^{med}_{sol}. Si la distance est plus petite que a, on assume que la population est trop concentrée dans l'espace paramétrique et on augmente la taille de v_{mut}. Au contraire, Si la distance est plus grande que b, on réduit v_{mut}.

$$v^{n+1}_{mut} = \begin{cases} g(c)\ v^n_{mut},\ g(c)\ \sim\ G(1+3c,c), & si\ \left\|v^1_{sol} - v^{med}_{sol}\right\| \leq a,c\ \in\ \Re \\ \frac{v^n_{mut}}{g(c)},\ g(c)\ \sim\ G(1+3c,c), & si\ \left\|v^1_{sol} - v^{med}_{sol}\right\| \geq b,c\ \in\ \Re \end{cases} \tag{12}$$

Non-uniforme

Dans le cas des mutations non-uniformes, le vecteur v_{sol} est modifié directement au lieu d'employer v_{mut}. Cette méthode[4] réduit graduellement l'amplitude des mutations avec chaque nouvelle génération ; on tente d'imposer un biais d'exploration au début de la simulation et un biais de convergence vers la fin. À chaque génération, on ajoute à chaque composante de v_{sol} une contribution provenant d'une distribution gaussienne centrée en 0. Un variable binomiale l détermine si on somme ou soustrait cette contribution. L'écart type est définit par l'équation 14, où le premier terme est l'écart entre une composante et une borne, r est une variable suivant une distribution uniforme et le troisième terme est responsable de supprimer l'amplitude des mutations lorsque la simulation mature. Le terme b contrôle à quel rythme les mutations sont supprimées et détermine donc fortement le biais imposé entre l'exploration et convergence.

$$v^{n+1}_{sol} = \begin{cases} v^n_{sol} + G(0,|\Delta_1|) & si\ l = 0,\ l \sim B(p = 0.5) \\ v^n_{sol} - G(0,|\Delta_2|) & si\ l = 1,\ l \sim B(p = 0.5) \end{cases} \tag{13}$$

$$\begin{aligned} \Delta_1 &= (v_{max} - v_{sol})\ r\ \left(1 - \frac{t}{t_{max}}\right)^c,\ r\ \sim\ U(a,b),\ a,b,c\ \in\ \Re \\ \Delta_2 &= (v_{sol} - v_{min})\ r\ \left(1 - \frac{t}{t_{max}}\right)^c,\ r\ \sim\ U(a,b),\ a,b,c\ \in\ \Re \end{aligned} \tag{14}$$

Comparaison de performance

Dans le but de sélectionner la meilleure stratégie, on organise une compétition entre les types de mutations, incluant la stratégie non adaptive qu'on nomme standard. Chaque scénario implique 25

simulations de 1000 générations. On liste un sommaire de la compétition au tableau 2. On remarque rapidement que les solutions offertes par les différentes stratégies sont toutes similaire, sauf pour celle basée sur la distance. L'implémentation de cette méthode converge avec difficulté ; on l'élimine des candidats pour l'optimisation. Ensuite, on constate que les pires solutions offertes sont aussi très similaires, sauf dans le cas de la fonction objectif, qui offre une solution inférieure d'une ordre de grandeur. Par contre, il est difficile d'évaluer si cette différence est due à la nature stochastique de l'algorithme ou à la méthode de mutation. Finalement, on s'intéresse à la synthèse des résultats. Parmis les trois stratégies restantes, les meilleurs solutions sont pratiquement homogènes. Par contre, la stratégie des mutations basées sur la fonction objectif se démarque des autres par sa faible moyenne sur l'ensemble des résultats. Sous ces considérations, on choisit de progresser avec cette méthode.

TABLE 2 – Résultats d'une compétition entre les types de mutations. Les simulations comptent $N = 10$ individus, une pression sélective $P_s = 0.25$, une probabilité de croisement $P_c = 0.8$ et une probabilité de mutation $P_m = 0.1$. On lance 25 simulations de 1000 générations pour chaque type de mutation et on affiche la valeur du khi-deux pour les 5 meilleurs solutions et pour la pire. On affiche additionnellement la moyenne et l'écart type pour les 5 meilleurs solutions, ainsi que pour l'ensemble des résultats.

χ^2	**Standard**	**Objectif**	**Distance**	**Non-Uniforme**
χ^2_1	0.36642	0.36559	0.53122	0.36615
χ^2_2	0.36677	0.36591	0.57684	0.36671
χ^2_3	0.36765	0.36594	0.59317	0.36700
χ^2_4	0.37007	0.36600	0.62379	0.36879
χ^2_5	0.37139	0.36622	0.69896	0.37238
χ^2_{max}	17.51611	1.87278	17.85782	17.91952
$\overline{\chi}^2_{(5)}$	0.36763	0.36593	0.58453	0.36821
$\sigma_{(5)}$	1.29×10^{-3}	2.04×10^{-4}	3.06×10^{-2}	2.26309×10^{-3}
$\overline{\chi}^2_{(25)}$	2.36603	0.92070	1.79563	1.75406
$\sigma_{(25)}$	4.66865	0.62107	3.30873	3.52739

4.4 Optimisation paramétrique de l'orbite de $\rho CoronaBorealis$

Armé d'une compréhension de l'effet des hyperparamètres et d'une stratégie de mutation adaptive gagnante, on s'intéresse finalement à notre tâche principale. Le procédé demeure le même que dans le cas de $\eta Bootis$, mise à part les conditions initiales et limites, qui dépendent de l'ensemble de données. On conserve le même ensemble de bornes offertes par le tableau 1, à l'exception de la période P, qu'on estime entre 10 et 100. On lance alors une dizaine de simulations avec $N = 10$, $p_m = 0.1$, $p_c = 0.8$, $p_s = 0.25$ et des mutations adaptives basées sur l'objectif. Les 5 meilleurs résultats sont offerts au tableau 3, ainsi qu'une moyenne et un écart-type pour chaque composante. Les solutions diffèrent peu ; une seule composante, τ, demeure incertaine avec un écart-type important. La figure 1 affiche les observations et la prédiction selon le vecteur solution v^2_{sol}.

TABLE 3 – Résultats de l'optimisation du modèle décrivant *ρCoronaBorealis*. La distance paramétrique entre les solutions demeurent très faible à l'exception du paramètre τ. Les simulations sont lancées avec $N = 10$, $p_m = 0.1$, $p_c = 0.8$, $p_s = 0.25$ et des mutations adaptives basées sur l'objectif. On spécifie un critère d'arrêt de $\chi^2 = 1$. Aucune des simulations rejoint ce seuil.

	χ^2	**P**	τ	ω	e	K	v_0
v^1_{sol}	1.18253	39.808	125.479	2.215	0.115	64.008	−46.237
v^2_{sol}	1.18357	39.784	85.501	2.16	0.108	63.923	−46.472
v^3_{sol}	1.18442	39.791	46.193	2.243	0.109	63.962	−46.592
v^4_{sol}	1.19045	39.771	86.431	2.295	0.110	63.220	−46.185
v^5_{sol}	1.20013	39.828	82.282	1.728	0.109	64.142	−47.739
μ	1.18822	39.796	85.177	2.128	0.110	63.851	−46.645
σ	6.56×10^{-3}	1.98×10^{-2}	2.51×10^{1}	2.05×10^{-1}	2.48×10^{-3}	3.24×10^{-1}	5.67×10^{-1}

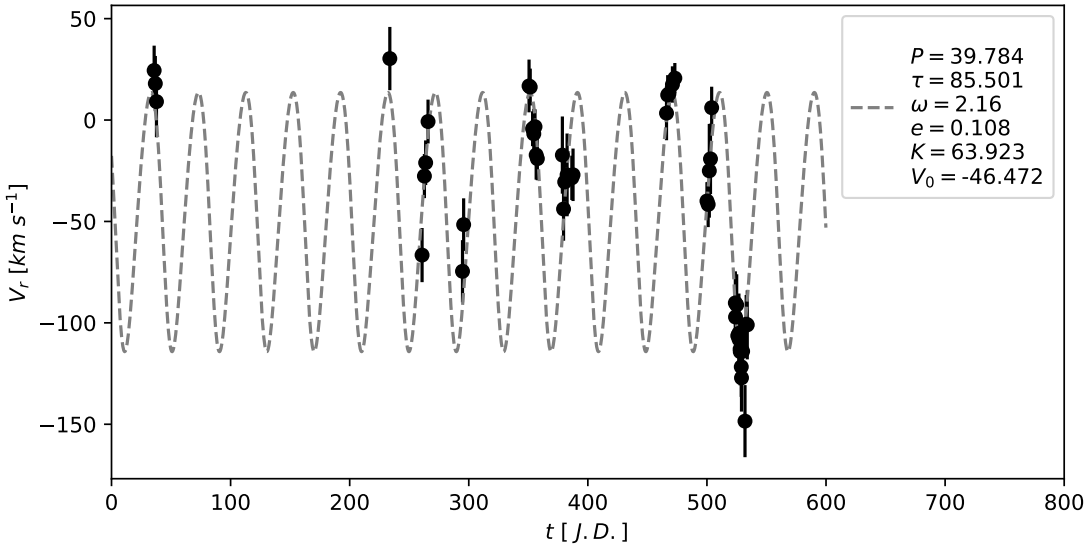


FIGURE 1 – Observations et prédiction pour le vecteur solution v^2_{sol} du tableau 3

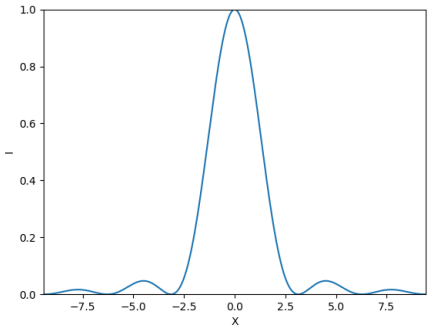
Au final, la distance paramétrique entres les solutions offertes à la table 3 demeure faible ; on suppose que l'ensemble des solutions offertes se retrouvent toutes près du même extrêmum, dans quel cas la moyenne des solutions est un meilleur estimateur de la solution optimale. On utilise donc la moyenne de chaque paramètre pour former un vecteur solution final. On caractérise l'erreur sur ce v^f_{sol} à l'aide des écart-types proposés au même tableau. On bâtit un intervalle de confiance à l'aide des propriétés des lois normales ; on choisit un erreur égale à 2σ pour construire un intervalle de confiance qui inclut $\approx$ 95% de nos données. Selon ces hypothèse, on obtient $v^f_{sol} = (39.80 \pm 0.04, 85 \pm 50, 2.1 \pm 0.4, 0.110 \pm 0.005, 63.9 \pm 0.6, -47 \pm 1)$.

Conclusion

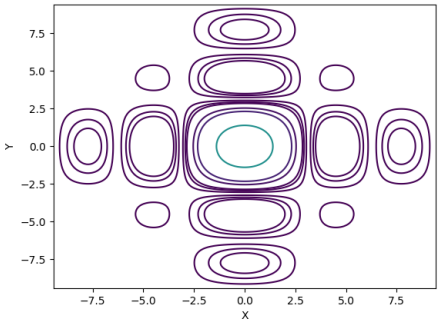
En implémentant une variété d'algorithmes basés sur différents types de mutations, il a été possible d'explorer et de comparer les algorithme évolutifs. On conclut que, parmi les stratégies explorées, celle basée sur l'objectif possède la meilleure performance générale. Par contre, 2 autres types de mutations génèrent régulièrement des vecteurs solutions v_{sol} de haute qualité, soient les mutations standards et les mutations non-uniformes. Une seule stratégie demeure systématiquement moins performante que les autres, soit la stratégie basée sur la distance. Au final, la stratégie employée pour l'optimisation des paramètres P, τ, ω, e, K, et V_0 produit des v_{sol} qui sont près les uns les autres en termes de distance paramétrique. On estime alors que ces solutions décrivent toutes le même maximum, qui se situe au point $v^f_{sol} = (39.80 \pm 0.04, 85 \pm 50, 2.1 \pm 0.4, 0.110 \pm 0.005, 63.9 \pm 0.6, -47 \pm 1)$ de l'espace paramétrique.

Au final, il demeure difficile de déterminer si v^f_{sol} est réellement un maximum global. Trouverait-on une meilleur solution si on continuait l'exploration ? Notre algorithme est-il biaisé dans ses choix de solutions ? Bref, il ne s'agit pas de nouvelles questions dans le cadre de l'analyse numérique. Par contre, on peut tout de même se contenter d'une performance accrue et incomparable à celle de procédés itératifs plus simples qui convergent vers l'extremum local le plus près. Avec une telle capacité d'exploration et d'optimisation, il serait intéressant de comparer l'optimisation d'un réseau neuronal par rétropropagation conventionnelle et par évolution adaptive. Les algorithmes évolutifs sont-ils une solution pertinente et réalisable à un problème qui demande déjà beaucoup de ressource de calculs ?

5 Annexe

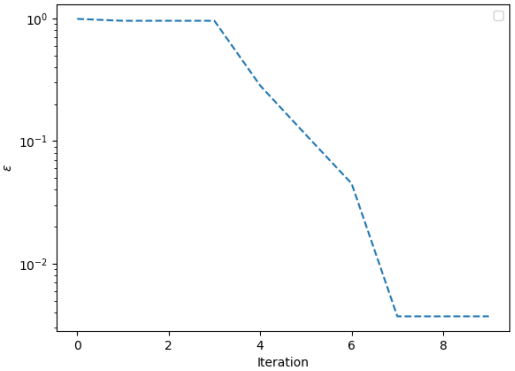


(a) Problème de diffraction 1D. Intensité en fonction de la position.

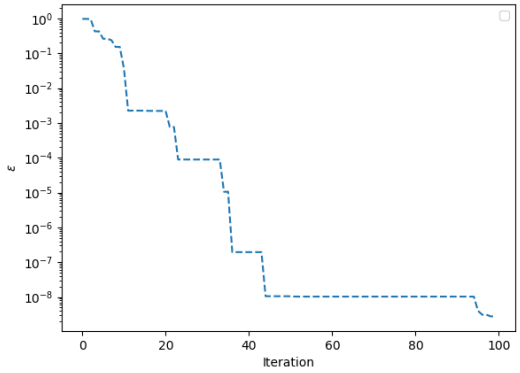


(b) Problème de diffraction 2D. Les isocontours prennent une valeur de $[0.001, 0.005, 0.01, 0.05, 0.1, 0.5]$.

FIGURE 2

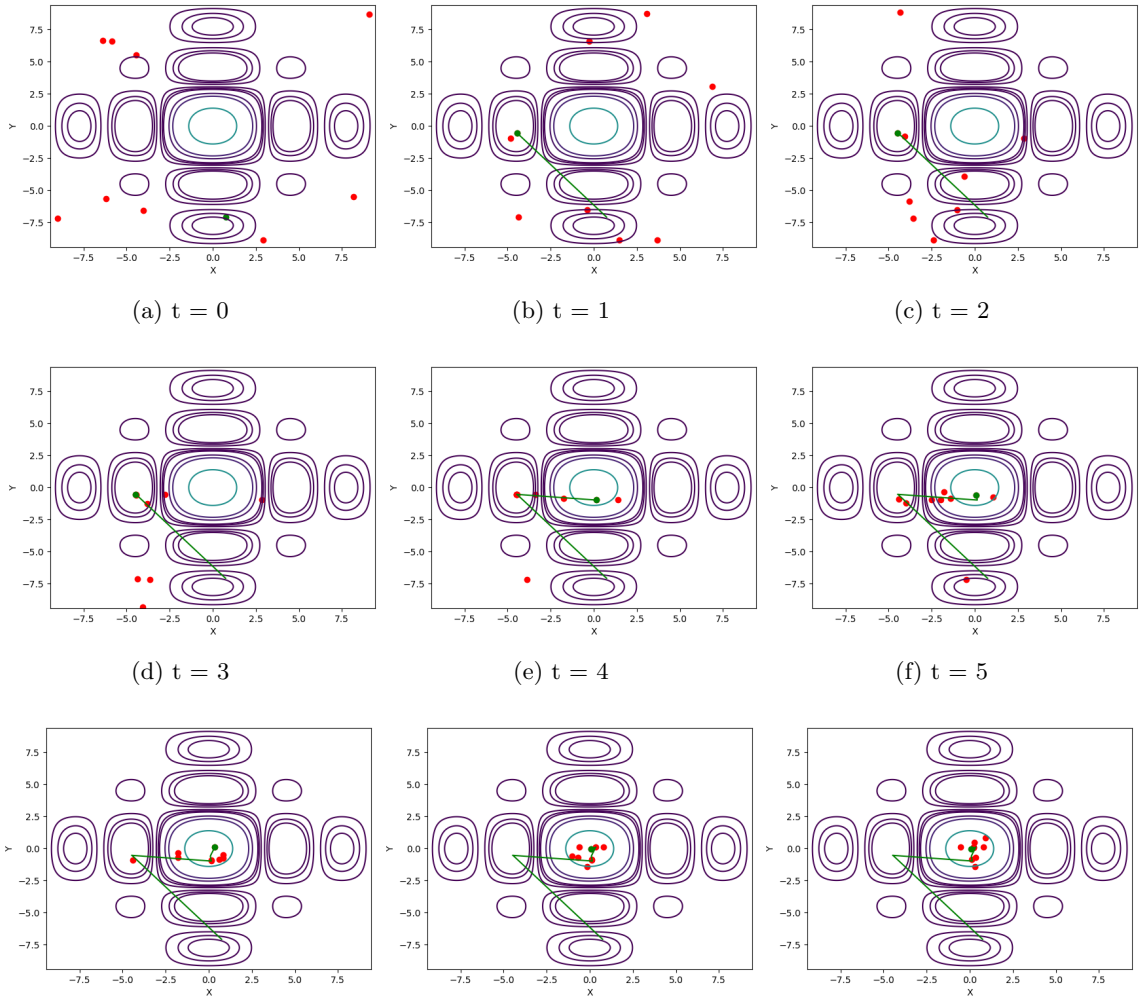


(a) Erreur de la simulation décrite à la figure 4.



(b) Erreur d'une simulation avec les même hyper-paramètres que la simulation décrite à la figure 4, mais qui se poursuit jusqu'à 100 générations.

FIGURE 3

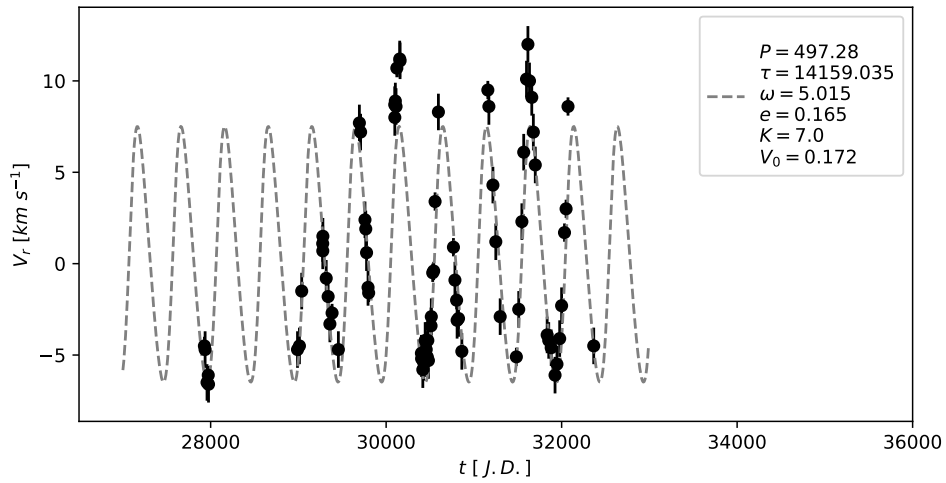


(a) t = 0 (b) t = 1 (c) t = 2

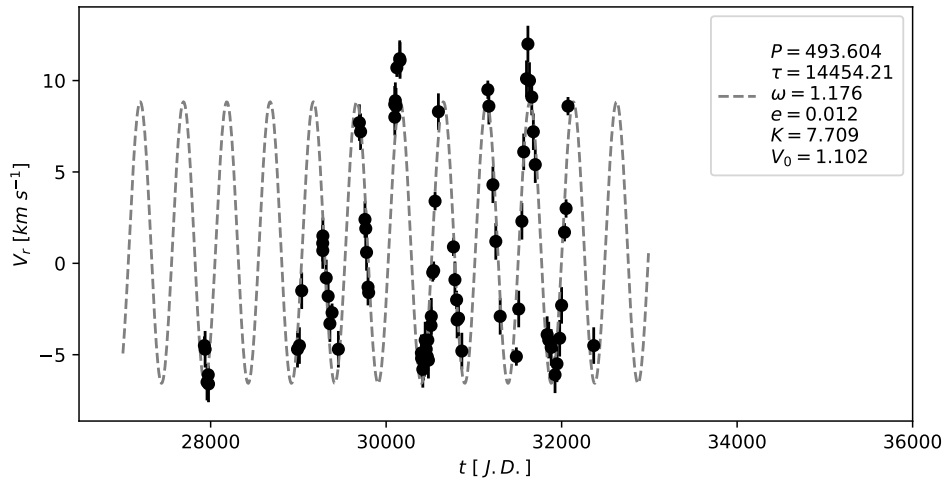
(d) t = 3 (e) t = 4 (f) t = 5

(g) t = 6 (h) t = 7 (i) t = 8

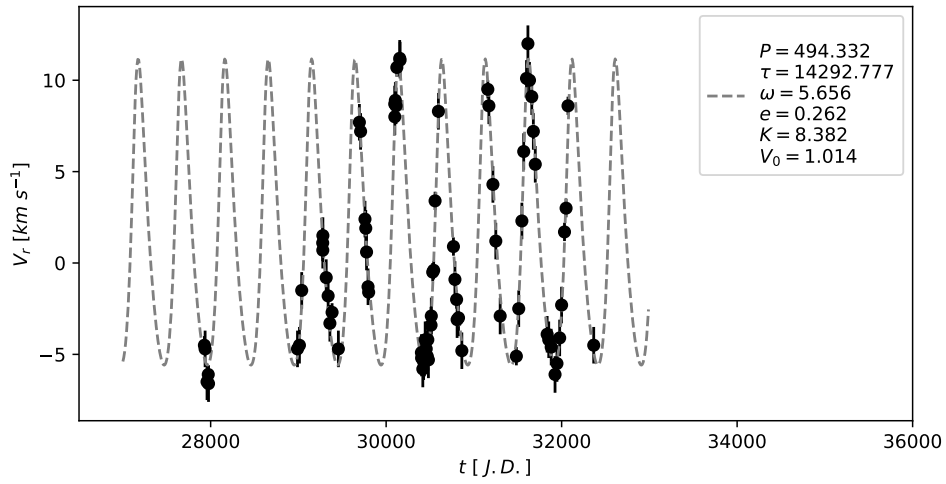
FIGURE 4 – Instantannées des 9 premières étapes d'une simulation à $N = 10$, $p_m = 0.1$, $p_c = 0.8$ et $p_s = 0.2$. On remarque que la meilleur solution se situe initialement près d'un maximum local. L'algorithme converge vers un meilleur maximum local dès la première étape. À l'étape 4, on converge vers le maximum global. L'élitisme nous garantie de ne pas s'échapper une fois l'obejctif atteint.



(a)



(b)



(c)

FIGURE 5 – Montre trois étapes de l'optimisation du modèle décrivant l'orbite de ρCrB, soit à 100, 250 et 500 générations. On remarque la convergence vers la solution connue.

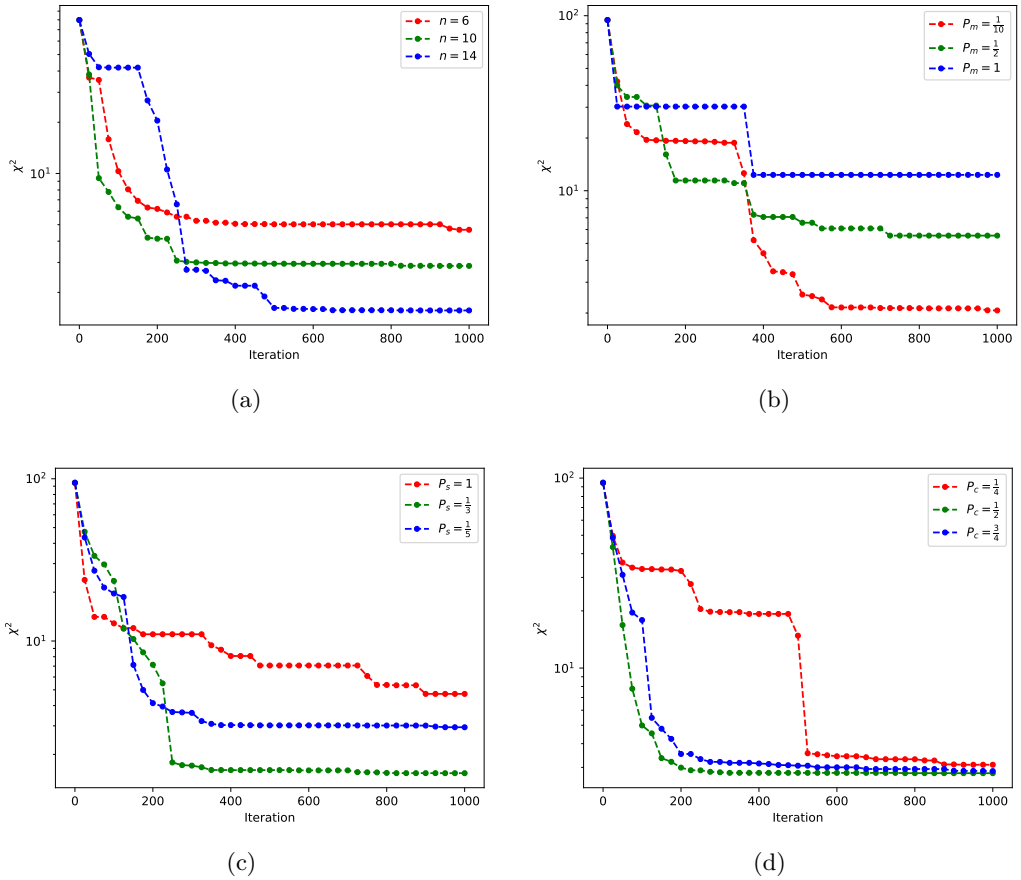


FIGURE 6 – Trois simulations sont lancées pendant 1000 générations afin de comparer l'effet des hyperparamètres. La figure a montre l'effet de la taille de la population, la figure b montre l'effet de la probabilité de mutation, tandis que la figure c montre l'effet de la pression sélective et la figure d, la probabilité de croisement.

6 Bibliographie

Références

[1] CHARBONNEAU, P., Recueil de notes, Modélisation numérique en physique, Département de Physique, Université de Montréal, Janvier 2019

[2] CHARBONNEAU, P., LAFRENIÈRE, D., Recueil de notes, Introduction à la physique numérique, Département de Physique, Université de Montréal, Automne 2016

[3] PFLEIDERER, E., Dépôt GitHub, https ://github.com/EricPfleiderer/Portfolio/tree/master/PHY3075/PROJET5

[4] WWW.NEURODIMENSION.COM, http ://www.neurodimension.com/genetic/documentation/OptiGenLibraryDotNet/GeneticServer/Non-Uniform_Mutation.htm

[5] CHARBONNEAU, P., http ://www.astro.umontreal.ca/ paulchar/phy3075/phy3075.html