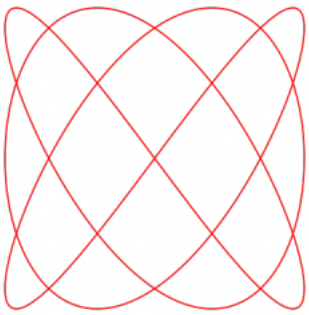
Le Moal Steven

Rapport :
Problème Langage Python

A3 – TD A

Comme à chaque semestre, nous devons réaliser un ou plusieurs projets en programmation. Ce rapport fait l'office du compte rendu quant au projet IA 'Star Wars' que j'ai choisi de réaliser en python.

L'objectif du projet était d'approximer les 6 valeurs réelles à partir d'une base de données (30 points) ainsi que la fonction des trajectoires (courbes de Lissajous) :



$$x(t) = p1 \times sin(p2 \times t + p3)$$
$$y(t) = p4 \times sin(p5 \times t + p6)$$

Avec x(t) et y(t) la position du satellite à un instant t donnée. Les p : i ∈ [1 ; 6] sont les paramètres qu'il va falloir découvrir afin de pouvoir anticiper au mieux les mouvements du satellite `a un instant t donné (t ∈ [0 ; 2π]).

Plan d'action

Afin de réussir la mission, vous allez devoir déployer un algorithme génétique capable de trouver une bonne combinaison de paramètres expliquant au mieux la trajectoire du satellite. Afin de restreindre la recherche, nous allons supposer que chaque paramètre est compris entre [−100 ; 100].

1. **Quelle est la taille de l'espace de recherche (utiliser une notation scientifique) ?**

   L'espace de recherche est sur R^2 pour x et y.
   Pour les *p : i ∈ [1 ; 6]* on a I : i : (] – 100 ; 100 [ ) avec I appartenant à *R*. (soit au final si on fait x et y séparément I x 3 appartenant à R^3 ou I x 6 appartenant à R^6)

2. **Quelle est votre fonction fitness ?**

   Ma fonction fitness peut être représenter par :

$$F_\alpha = \sqrt[2]{\frac{\sum_{i=1}^{N}[(\alpha_i) - (\beta_i)]^2}{N-1}}$$

$\boldsymbol{\alpha_i}$: valeurs de $x(t_i)$ ou $y(t_i)$ calculer avec les p des individus
$\boldsymbol{\beta_i}$: valeurs de $x(t_i)$ ou $y(t_i)$ de la base de données
$\boldsymbol{N}$: nombre de valeurs de la base de données

Ce qui ressemble très fortement à la racine carrée de la variance empirique de l'erreur. L'intérêt que plus un individu est proche en moyenne par rapport aux points plus son fitness sera petit (en particulier pour les écarts inférieurs à 1).
Je pense que cela ouvre une meilleure perspective que la moyenne car on nous a dit que les points contenaient des erreurs dans leurs coordonnées, le fitness est donc ici plus fiable pour sa stabilité.

On essai ici bien évidemment de minimiser le fonction fitness.

3. Décrivez les opérateurs mis en œuvre (mutation, croissement) ?

Les 2 opérateurs de la mutation et le croissement sont en mise en œuvre. L'opérateur de mutation est mis en œuvre pour une partie de la population (avec le fitness le moins : ici on prend la moitié de la population total) et l'opérateur de croissement est mise en place pour l'autre moitié qui a un meilleur fitness. Dans les 2 cas une première sélection des individus est faite : une variable aléatoire est comparée à une valeur pour savoir si un individu est choisi (le pourcentage 'loi normal' pour être pris en mutation est de 0.1 et le pourcentage 'loi normal' pris en croissement de 0.6). Cela permet de mieux simuler les processus dans la vrai vie (permettre de converger vers un meilleur individu en minimisant les chances de tomber dans un minimum local).

Pour la mutation, une deuxième condition aléatoire avec un pourcentage de 0.1 pour sélectionner les gènes choisis : dans ce cas une valeur aléatoire compris entre 0 et 1 (cela permet d'augmenter un tout petit peu les valeurs et est très efficace pour de grande population) et pour tous les gènes qui n'ont été pas pris, une troisième condition aléatoire avec un pourcentage de 0.4, est faite pour savoir si elle peuvent être sélectionner : les gènes choisit seront alors multiplier par une valeur aléatoire afin de pouvoir sur de grande et courte population parcourir l'ensemble de l'intervalle.
Il y a cependant un cas particulier pour p3 et p6 qui sont les valeurs qui sont rendent plus chaotique le système sur à l'intérieur du sinus. Afin de couvrir toutes les valeurs elles ont accès seulement à la 3^{e} condition et sinon elles ont une valeur attribuée aléatoirement sur I.

Pour le croissement, similairement en plus de la première condition et une deuxième condition de sélection est réaliser pour savoir si un gène est sélectionné pour le croissement (pourcentage de sélection de gène est de 0.7). Pour revenir au croissement en lui-même afin de sélectionner les individus qui vont être croisés entre eux, un système pour changer l'indice des individus sélectionnés (et permettre de ne pas tomber dans des minimums locaux : redondance de croissement). Exemple : cela va servir à changer les individus croiser ex : 1^{ere} génération 1 - 2 (indice croisées), 2^{e} génération 1 – 3, 3^{e} génération 1 - 4 ….
Pour le croissement des gènes, une valeur aléatoire (loi normal) est générée pour couper les 2 gènes (float avec 3 chiffres pour les entiers au maximum et une précision n pour les chiffres décimale) en 2 n'importe où sur la partie entière ou décimale et permutera de varier entre la partie entière est décimale (la plus probable d'arrivé).

On choisit un croissement en un point (on coupe une fois le gène) car c'est le plus optimale pour notre situation plutôt qu'un découpage en plusieurs points et en croissement uniforme.

L'idéale aurait été de représenter le float en binaire (1 pour la partie entière et 1 pour la partie décimale) qui aurait aidé à mieux représenter les chromosomes des gènes pour le croissement et la mutation.

4. **Décrivez votre processus de sélection ?**

Une grosse partie du processus de sélection est réaliser dans le processus mutation et le croissement que j'explique ci-dessus. Le tri des individus se fait par rapport à leur fitness, plus il est petit par rapport aux autres, plus il sera en avant dans la liste de sélection. De plus on sélectionne la moitié de la population meilleure pour le croissement et le quart le moins bon pour la mutation (la population final pour la prochaine génération sera composé de : la moitié meilleur, les croissements, les mutations sur le moins bon quart et un quart de la population initial généré aléatoirement. Cela permet de mieux converger grâce en plus aux conditions aléatoires que j'ai choisi de réaliser pour sélectionner les individus choisit dans les parties sélectionnées pour la mutation et le croissement ainsi que les gènes qui seront sélectionnés (dans l'intention de couvrir l'intervalle de manière efficace pour notre problème).

5. **Quel est la taille de votre population, combien de générations sont nécessaires avant de converger vers une solution stable ?**

Pour obtenir une solution stable, qui reviendra souvent (la meilleure convergence que je peux avoir avec un minimum de temps), il faut une population de 10 000 et ~ (80 – 100) générations. On peut choisir d'autres paramètres plus petits (qui seront aussi efficaces), mais l'algorithme avec les conditions aléatoires est optimisé pour une population grande afin de donner du réalisme au processus de sélection naturel.
Le fitness en x sera (en générale) entre 0.15 et 0.05 et en y le fitness sera (en général) entre 0.25 et 0.19.

6. **Combien de temps votre programme prend en moyenne (sur plusieurs runs) ?**

J'utilise le module time avec la fonction process_time() pour mesurer le temps d'utilisation d'une fonction. Pour les paramètres que j'ai cités avant (N : 10 000 et un nombre de génération : 80-100), il faut 30 secondes en moyenne (pour 100 génération).

7. **Si vous avez testé différentes solutions qui ont moins bien fonctionnées, décrivez-les et discutez-les.**

Pour commencer à répondre à la question, il serait intéressant de parler du choix que j'ai fait pour trouver plus rapidement des solutions convergence. Les fonctions x(t) et y(t) ont deux expressions similaires mais avec des paramètres, surtout pour la base de données qu'on a eue, différents. De plus les signaux sinusoïdaux sont

très chaotiques et nous n'avons aucune indication (avec confidence on peut mettre dire qu'avec l'énoncé du problème) que dans notre cas les fonctions x(t) et y(t) sont liées. Pour tester cela j'ai testé de trouver les solutions x(t) et y(t) ensemble (tableau de 6 valeurs) et séparément (2* tableau de 3 valeurs), j'ai pu remarquer que l'algorithme converger beaucoup plus rapidement en traitant les valeurs séparément.

Le choix des pourcentages et le choix de la loi utilisé (uniforme et normale) pour les conditions aléatoires ont aussi été une question importante. Sans celle-ci, l'algorithme est moins fiable dans sa convergence (j'explique pourquoi je fais cela dans la question 3 : rendre réaliste le processus de sélection). On doit essayer de minimiser les minimums locaux et couvrir aux maximums l'intervalle des p : i, on doit donc maximiser les chances qu'un croissement se produise tous en gardant une grande partie des informations des individus (la sélection des individus plus la sélection des gènes avec des pourcentages prédéfinis ainsi qu'un croissement en un point choisie aléatoirement sur le gène : le float) et pour les mutations on doit faire le compromis de couvrir au maximum l'intervalle et converger une base d'information de l'individu. On peut améliorer la combinaison en prenant en compte le nombre d'individu de la population (la sélection des individus en plus de la sélection des gènes ainsi que le type de mutation appliqué).

Le choix du fitness est aussi important. Je pense avec confiance que choisir une formule proche de la variance empirique de l'erreur est mieux pour notre problème que l'espérance empirique de l'erreur. On aurait put aussi utiliser la fonction d'inter-corrélation qui sert à comparer de signaux sinusoïdaux.