# Introduction aux méthodes numériques et projet (PROJ0001): Modélisation de la trajectoire d'une balle de tennis

Premier bachelier en sciences de l'ingénieur
Année académique 2021-2022.Travail réalisé par:

Loic Delbarre (S215072) Rafik Lourmathi (S212097)
Salman Guseynov (S216862)

Le 8 avril 2022

# Contents

# 1 Question 1

## 1.1 Méthode sécante

Nous avons donc implémenté la méthode de la sécante et avons étudié les 3 cas qui peuvent se présenter à nous comme annoncé dans l'énoncé.

Tout d'abord, envisageons le première possibilité : lorsque la sécante converge, elle renvoie un tuple contenant les coordonnées du zéro approximé.

Ensuite, dans le deuxième cas, afin de ne pas diviser par zéro dans l'équation de la sécante(ref), on retourne un statut d'erreur.

La troisème plausibilité est que la méthode de la sécante ne nous permet pas d'être assuré de la convergence de la fonction. C'est pour cela que nous implémentons un nombre d'itérations maximum comme gardien de boucle. Si jamais la fonction n'a pas convergé après le nombre maximum d'itérations (`iteration_max`), la fonction en déduit qu'il n y a pas convergence. On a défini le nombre maximal d'itérations à 100.
On sait que la trajectoire d'une balle de tennis est parabolique. Donc, en utilisant la méthode de la sécante, on converge rapidement vers un zéro. Dans le cadre de ce cours, nous étudions des trajectoires paraboliques. Une vingtaine d'itérations devrait être suffisante. Mais, au vu de la faible complexité, nous avons fixé le nombre d'itérations à100.

Voici un exemple(ref) d'un graphique d'une fonction que l'on pourrait rencontrer lors de nos calculs de trajectoire. On remarque qu'avec 10 itérations, on atteint assez rapidement une précision satisfaisante.

## 1.2 méthode de la bissection

La technique de la bissection est plus efficace que la sécante. En effet, cette méthode assure la convergence lorsque les bornes sont de signes contraires et elle est definie partout entre ses bornes. Si la fonction n'est pas convergente, on aura une sortie d'erreur.

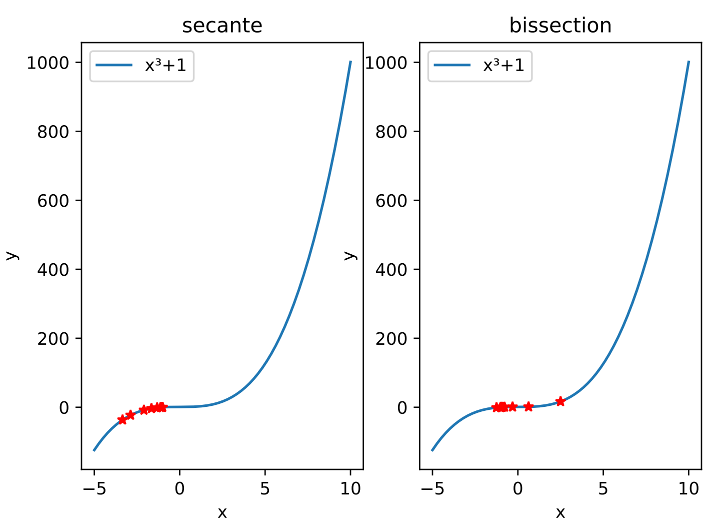



Figure 1: difference bissection et secante

## 2 Question 2

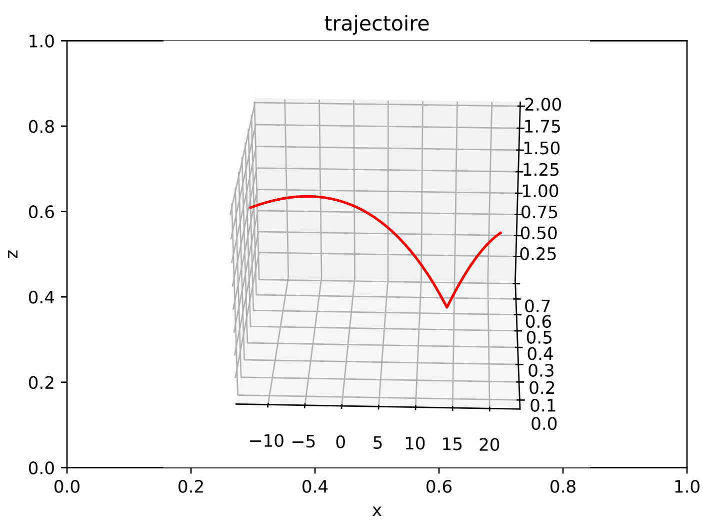
Afin de regrouper les différentes constantes utilisées dans le projet, nous les avons centralisées sous un fichier nommé const.py . On a remarqué egalement la possibilite d'une mise en évidence des différentes constantes. Ainsi, on évite recalculer le produit des constantes à chaque itération (ref). Ici la fig 1 On voulait profiter de la caractéristique dtype des `numpy.array` pour choisir le nombre de décimales dans le calcul si besoin. Mais finalement, on s'est rendu compte que la valeur attribuée par défaut était amplement suffisante.

Pour stocker les valeurs dans Euler, il a été plus judicieux d'allouer statiquement un tableau de la taille du nombre d'étapes maximal et de le réduire lorsque l'événement (si la balle touche le sol) se produit. Dans la première option envisagée, on réallouait dynamiquement un tableau à chaque itteration avec `np.append`. Mais on a constaté des pertes impressionantes de performances pour la réallocation de grands tableaux. Pour un tableau de 800 000 cases , le fait de changer de technique a réduit le temps de moitié.

D'une autre part, nous avons implémenté l'utilisation `solve\_ivp` de `scipy`. Au vu des valeurs par défaut et de leur précision, nous avons decidé de les conserver. Et donc les valeurs par défaut sont respectivement pour `rtol` de 0.001 et pour `atol` de 0.000001.

$$\ddot{x} = \frac{\rho \cdot \pi}{m} \cdot \frac{d^2}{8} \cdot \|\dot{x}\|^2 \left(\frac{1}{2 + 1.96 \frac{\dot{x}}{\|w\| \cdot d}} + C_d\right) - g$$

## 3 Question 3




### 3.1 Comparaison des deux méthodes

En comparant solve_ivp et Euler, on se rend compte assez rapidement de la différence de temps d'excution. solve_ivp est instantané pour une précision de 10e-5 alors qu'Euler prend déja 6 secondes. Et pour une précision 10 fois plus petite, Euler augmente drastiquement vers 60 alors que Solve_ivp reste instantané.

### 3.2 conception de `trajectoirefilethorizontal`

On a décidé de prendre un argument non obligatoire à la fonction `trajectoirefilethorizontal` pour gérer le cas où on a un rebond de façon recursive. En effet un rebond peut être considéré conmme le point de départ d'une nouvelle trajectoire. L'avantage de cette technique, c'est qu'elle permet d'être modulaire au niveau du rebond. Ainsi si on décide de ne plus prendre en compte le rebond, on ne devra pas changer de fonction.

Les événements choisis pour l'utilisation de solve_ivp ont été déterminés de manière analogique et non discrète car `solve_ivp` est capable d'utiliser ces résultats pour converger plus rapidement et plus précisement. Pour le choix des événements, pour la prise en compte du filet, nous avons opté pour `CubicSpline` avec `bc_type` en type natural pour un resultat plus naturel.

## 4 Question 4

### 4.1 Choix de la méthode

Le choix s'est porté sur la méthode de la sécante car celle-ci ne nécessite pas de bornes opposées pour pouvoir fonctionner. Les points d'existence choisis pour la sécante sont à determiner au cas par cas, en fonction de la fonction donnée. En effet, les valeurs intiales determineront l'ensemble des valeurs pouvant être obtenues par la méthode.

### 4.2 Implémentation

Pour réaliser les diverses fonctionnalités, au vu de la tâche répétitive, nous avons décidé de séparer le travail en plusieurs fonctions ( `get cible hauteur`et `getcible rebond`). Pour faire également varier les différents éléments, nous avons implémenté `multinorm``multiomega`ainsi que `rotangle`. L'idée est de faire varier un paramètre d'entrée x à travers une sécante ou une bissection. Pour les différentes fonctions, nous avons dû choisir des bornes différentes. Pour la rechercheHauteur, il etait spécifié dans les consignes de prendre une hauteur de 2 à 3 m. Pour la rechercheOmega et recherchevitesse, nous avons dû poser une valeur maximale a la norme, que nous avons fixée à 100m/s et 100rad/sec.

## 5 Question 5

Nous avons pris en compte que la direction de la vitesse angulaire était donnée en entrée de la fonction. Nous avons décidé de calculer la hauteur pour chaque répartition avec une étape de 10e-2 . A chaque étape, on recalculera la hauteur. A la fin de toutes les répartitions, on essayera de retourner la répartition qui permet de trouver la valeur h la plus élevée.

Avec une étape de 10e-2, on fait 100 tours de boucles, ce qui est la valeur minimale pour avoir une valeur en pourcentage.

Pour connaitre la répartition de l'énergie, on se base sur l'équation suivante, premettant de calculer les diferentes fluctuation.

Voici ici 2 un graph de la hauteur finale en fonction de la vitesse. Ce graph démontre clairement que la dérivée s'annule bien uniquement lorsque la balle est à l'apogée de la hauteur maximale.

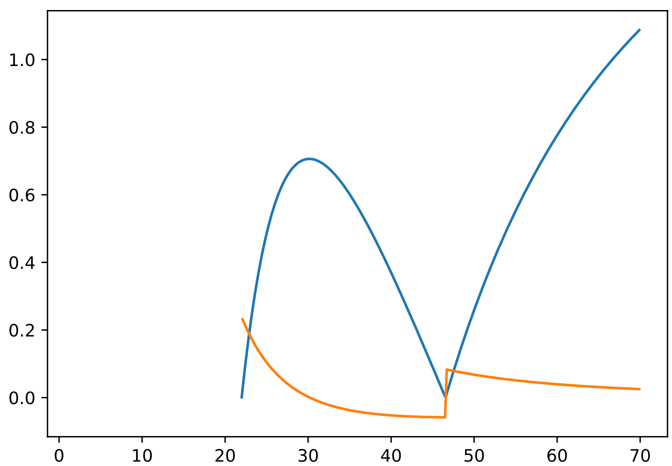



Figure 2: test