# TP2

RAJA GANAPATHY Srinivas ENGUIX Précillia

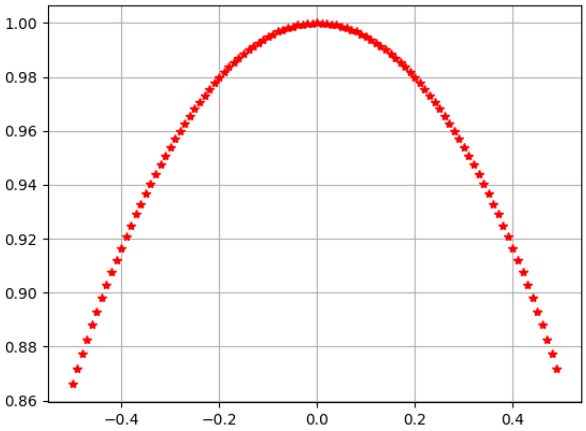
November 16, 2017

## 1 Création de la fonction f (x)

Nous avons crée la fonction suivante:

$$f(x) = \sqrt{1 - x^2}$$

Pour pouvoir créer la fonction f, on a du utiliser la commande 'from math import sqrt' pour ensuite utiliser la commande sqrt qui est la racine carré. Mais même après avoir utilisé cette commande, on a quand même eu une erreur. On s'était rendu compte que l'erreur venait de l'intervalle choisi, car au dela de 0.5 la fonction n'existe pas.



## 2 Méthode du point milieu

Nous avons repris la fonction de l'exercice 2 du TP1, qui consiste à calculer les premiers termes d'une suite pour ensuite les enregistrer dans une liste. On a crée une liste pour les subdivisions régulières xo,...,x4, puis une autre pour les milieux de ces sous-intervalles c1,...,c4. On a crée une nouvelle liste pour calculer la surface sk. On a crée une variable S pour calculer la somme de la surface.

Lors de la compilation de l'exercice 2, nous avons eu plusieurs erreurs, dont une que nous n'arrivions pas à résoudre. Le terminal affichait "out of range", nous avons fini par comprendre que nous devions augmenter les rangs des listes cc et xx, ou bien diminuer celui du ss, car on ne disposait pas d'assez de termes pour calculer le dernier.

Nous avons ensuite crée une variable A qui calcule l'erreur commise entre la somme des surface et l'intégrale de I. L'erreur commise est d'environ 0.033620716.

Nous avons ensuite utilisé la fonction clock() du module time, pour mesurer le temps de calcul de l'intégrale. Nous avons remarqué qu'à chaque fois que nous exécutons le programme, le temps de calcul de l'intégrale varie.

Pour écrire la fonction point milieu, nous n'avions pas besoin de reprendre les listes. De ce fait, la fonction était beaucoup plus simple et rapide à écrire.

Nous avons testé la fonction point milieu en variant le "n". Nous nous sommes aperçu que plus le "n" était grand, plus l'erreur était petite, en revanche, le temps de calcul de l'intégrale était plus long.

Nous avons crée automatiquement le tableau obtenu précédemment à l'aide d'une boucle for. Le voici ci dessous :

| n | erreur | temps (sec.) |
|---|---|---|
| 10 | 0.000480384 | 1.2e-05 |
| 100 | 4.81118e-06 | 4.6e-05 |
| 1000 | 4.81125e-08 | 0.000448 |
| 10000 | 4.8113e-10 | 0.005517 |
| 100000 | 4.81315e-12 | 0.07168 |
| 1000000 | 5.06262e-14 | 0.323827 |

# 3 Méthode du trapèze

Dans cette méthode, nous sommes parti de la fonction point milieu, ici la fonction trapéze est un polynôme de degré 1, c'est une fonction affine. Comme c'est un polynôme de degré 1 , nous utilisons 2 points de subdivisions pour calculer la surface. Dans cette méthode, nous n'utilisons pas les milieux. Nous avons fait de nouveaux tests avec la fonction trapéze, les résultats sont dans le tableau ci-dessous :

| n | erreur | temps (sec.) |
|---|---|---|
| 10 | 0.00678716 | 5.6e-05 |
| 100 | 6.73626e-05 | 0.000249 |
| 1000 | 6.73576e-07 | 0.002355 |
| 10000 | 6.73575e-09 | 0.034847 |
| 100000 | 6.73401e-11 | 0.064067 |
| 1000000 | 6.68465e-13 | 0.636837 |

# 4 Méthode de Simpson

La méthode Simpson est beaucoup plus complexe car c'est un polynôme de degré 2, c'est une parabole. Nous avons ici besoin de 2 points de subdivisions et le point de leur milieu pour calculer la surface. Dans le calcul, le point du milieu compte 4 fois plus que les bornes. Nous avons fait de nouveaux tests avec la fonction Simpson, les résultats sont dans le tableau ci-dessous:

| n | erreur | temps (sec.) |
|---|---|---|
| 10 | 2.11626e-07 | 1.9e-05 |
| 100 | 2.13813e-11 | 9e-05 |
| 1000 | 1.9984e-15 | 0.000858 |
| 10000 | 9.99201e-16 | 0.008552 |
| 100000 | 6.43929e-15 | 0.084432 |
| 1000000 | 3.90799e-14 | 0.842534 |

# 5 Comparaison des méthodes

1. Comparaison des méthodes :

La méthode du point milieu est un polynôme de degré 0, c'est une fonction constante, qui utilise un unique point; le point milieu des sous-intervalles, pour calculer la surface.
La méthode du trapèze est un polynôme de degré 1, c'est une fonction affine, qui utlise 2 points de subdivisions pour calculer la surface.
La méthode de Simpson est un polynôme de degré 2, c'est une parabole, qui utilise 2 points de subdivisions, et le point de leur milieu pour calculer la surface.
Nous pouvons donc voir que la méthode de Simpson est la méthode la plus complexe à écrire, suivi de celle du trapèze.

2. Comparaison des erreurs :

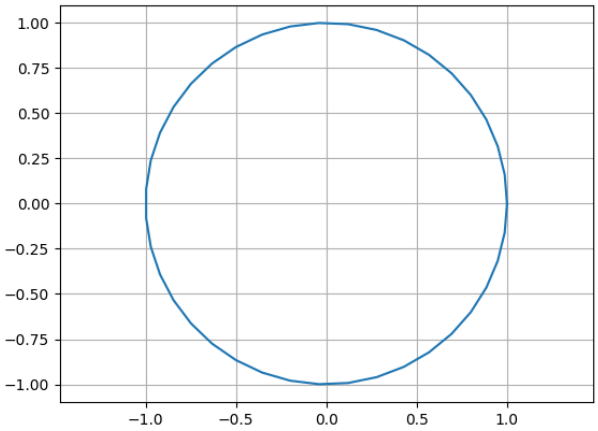
On remarque que la méthode du trapèze est celle qui produit le plus d'erreur lors des calculs comparé à celles du point milieu et de Simpson. La méthode de Simpson est la meilleure car elle produit moins d'erreur que les autres mais c'est la plus compliquée.

3. Comparaison des temps :

Comme vu précédemment, la méthode de Simpson est la plus efficace, mais ce n'est pas la plus rapide (fonction polynôme de degrés 2). Lorsque l'on compare les temps de calculs , la méthode la plus rapide est celle du point milieu puis du trapèze suivie de Simpson. Plus le degré du polynôme est élevé, plus la fonction est complexe donc plus le calcul est plus long.

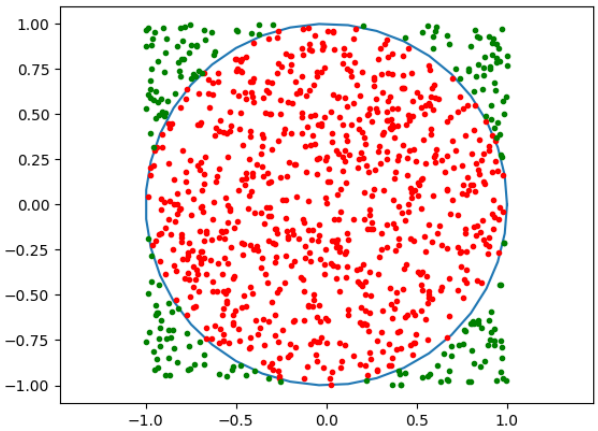
# 6 Méthode de Monte-Carlo

Voici le cercle unité :




La surface du disque unité est $\pi$.

A l'aide de la fonction numpy.random.rand(), nous avons généré des valeurs aléatoires suivant la distribution de loi uniforme sur [0,1]. Ensuite nous avons généré une variable N qui représente 1000 points suivant la distribution uniforme sur le carré $[-1, 1]^2$. Puis nous avons fait apparaître sur le graphique suivant, en rouge les points intérieurs au disque unité, et en vert les points extérieurs du disque.



Nous avons ensuite compté le nombre de points intérieur au cercle et le nombre de points extérieurs au cercle. La méthode de Monte-Carlo consiste à prendre le rapport 1/N comme approximation de la surface S du disque, nous avons calculé l'erreur commise : (1/N) - S, celle ci change à chaque fois que l'on compile. Ensuite, à l'aide de la fonction time.clock() nous avons mesuré le temps de calcul de l'erreur commise, celle ci change aussi à chaque fois que l'on compile. On a ensuite crée la fonction Monte-Carlo, qui renvoie une approximation de la surface du disque unité.

On a ensuite testé la fonction Monte-Carlo, cette fonction prend des valeurs aléatoires, le tableau affiché ci-dessous n'affichera pas les mêmes valeurs à chaque compilation.

METHODE DE MONTE CARLO

| n | erreur | temps |
|---|---|---|
| 10 | 0.858407 | 0.000148 |
| 100 | 0.258407 | 0.000206 |
| 1000 | 0.0415927 | 0.001888 |
| 10000 | 0.0191927 | 0.018849 |
| 100000 | 0.00504735 | 0.136166 |
| 1000000 | 0.00358865 | 0.743317 |

Ensuite nous avons repris le même travail avec le calcul du volume de la boule unité par la méthode de Monte-Carlo, voici ci-dessous le tableau correspondant aux résultats, celui ci n'affichera pas les mêmes valeurs à chaque compilation.

```
METHODE DE MONTE CARLO 2

n               |erreur         |temps
-------------------------------------------------
             10|        1.05841|        5.1e-05

            100|       0.278407|       0.000104

           1000|      0.0935927|        0.00102

          10000|      0.0347927|       0.010886

         100000|     0.00138735|       0.104719

        1000000|     0.00147735|           1.001
```

# 7 Analyse de la séance de TP

Par rapport au TP1, le codage en python est mieux maitrisé, de même pour le Latex. Nous avons eu du mal à finir ce TP, car il était long et la rédaction du latex a été conséquente. Nous avons quand même dû faire quelques recherches pour intégrer quelques commandes en plus comme par exemple le $\pi$.