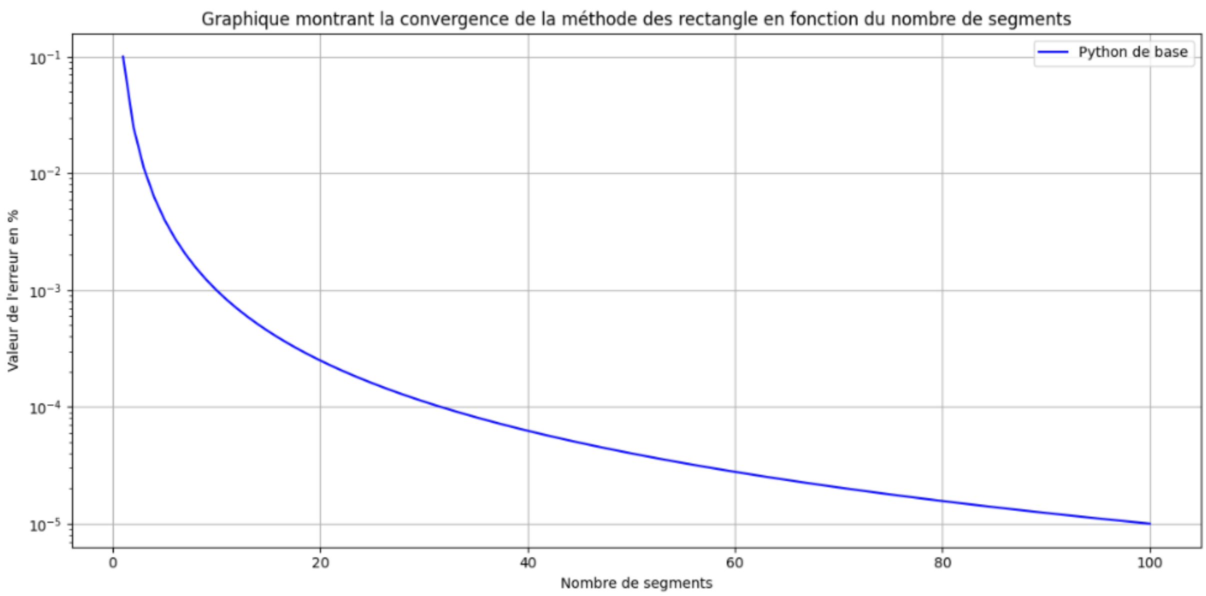
Rapport du mini projet B :

L'objectif du projet est de comparer 3 méthodes d'intégration numérique (rectangle, trapèze et Simpson) pour pouvoir évaluer leur efficacité en termes de précision et de rapidité d'exécution sur le calcul d'une l'intégrale d'un polynôme de degrés inférieur ou égale à 3.

Dans ce rapport nous étudierons dans un premier temps la convergence et la complexité de chacune des 3 méthodes, implémentée en python de base, vectorisée ou avec une implémentation pré-programmée dans un module python scientifique. Ensuite, nous comparerons les 3 méthodes entre-elles et évaluant leurs précisions ainsi que leurs temps d'exécution.

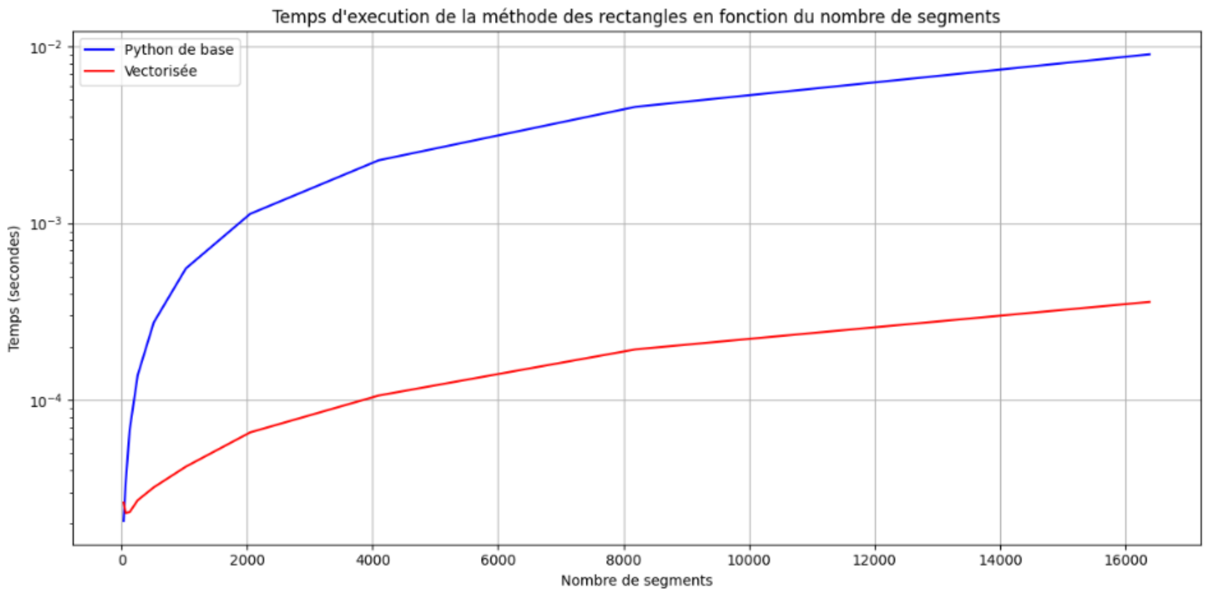
I.1 Méthode des rectangles

L'objectif de cette méthode est d'approximer l'intégrale d'une fonction en découpant l'aire sous sa courbe en plusieurs rectangle. L'utilisateur peut alors choisir la largeur de ces rectangles en indiquant le nombre de segment de découpe de l'intervalle d'intégration. Ainsi plus le nombre de segment est important, plus les rectangles seront nombreux et "minces", plus la méthode des rectangles donnera une approximation exacte. Nous pouvons retrouver ce résultat sur le graphique ci-dessous obtenu à partir de notre code, on peut voir que l'erreur entre la méthode et l'intégration exact décroit avec le nombre de segment.



Ce test a été effectué pour le polynôme 1 + X + X^2 + X^3, sur l'intervalle [0,1].

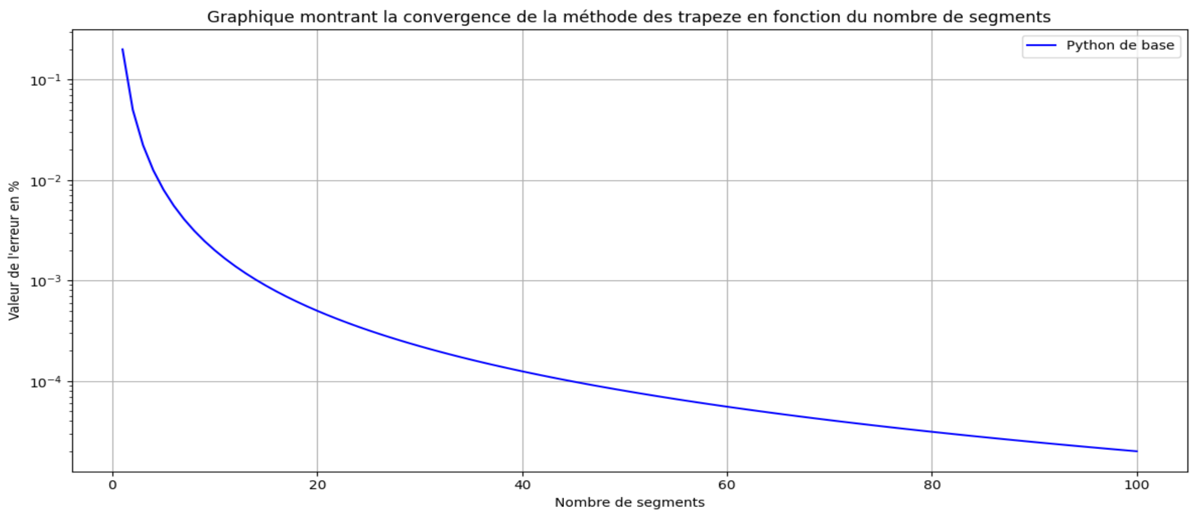
Par la suite nous avons comparé la rapidité d'exécution entre cette méthode implémentée sur python de base et sa version vectorisée. Nous obtenons alors le graphique suivant :



On peut donc conclure que la méthode vectorisée permet un gain de temps significatif : en prenant en compte l'échelle logarithmique du temps d'exécution de ce graphique on remarque que la version vectorisée est plus de 30 fois plus rapide pour un grand nombre de segment.

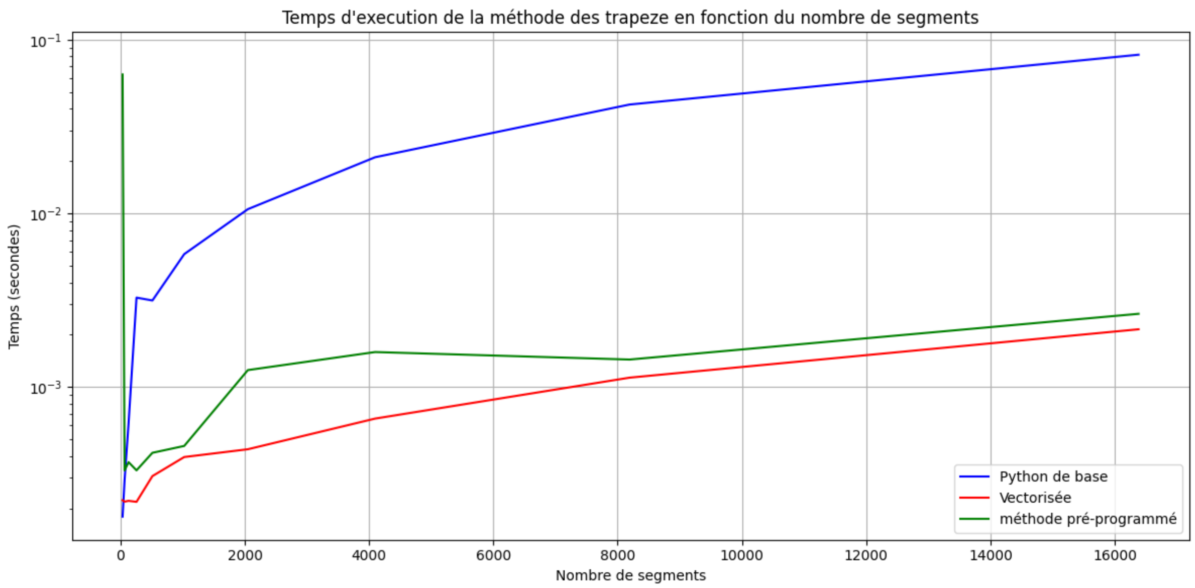
I.2 Méthode des trapèzes

L'objectif de cette méthode est de découper l'aire sous la courbe en plusieurs trapèze après avoir découper la fonction en plusieurs segments. Ainsi la formule pour l'aire de chaque trapèze est simple à calculer et donne une bonne approximation. De la même manière que précédemment on s'attend à ce que plus le nombre de trapèze est grand, plus ils sont mince et approximent la courbe correctement. Cela est vérifiable en testant la convergence de cette méthode dans la figure suivante :



Ce test a été effectué pour le polynôme 1 + X + X^2 + X^3, sur l'intervalle [0,1] et montre clairement que l'erreur diminue avec le nombre de segment.

Par la suite nous avons comparé la rapidité d'exécution entre cette méthode implémentée en python de base, sa version vectorisée et la fonction préprogrammé de sciPy. Nous obtenons alors le graphique suivant :

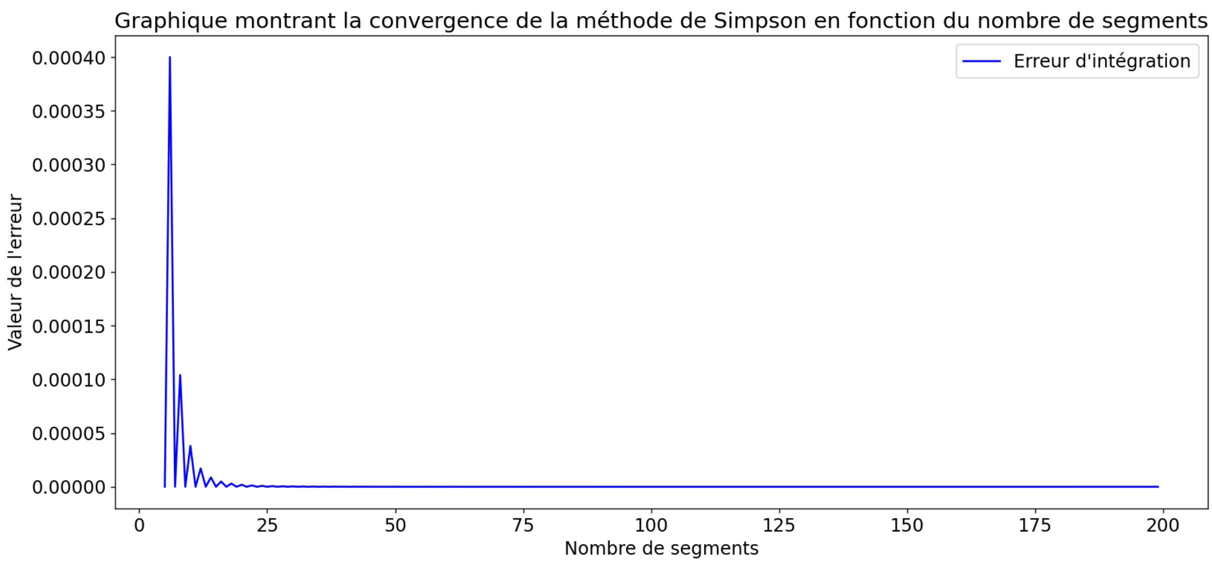


Ce graphique montre clairement que la vectorisation de la fonction offre un gain de temps majeur comparé à la programmation en python de base. La méthode pré-programmée ne semble pas ici plus rapide que notre propre version vectorisée mais on peut supposer qu'elle l'est pour des nombre de segment très élevé bien que ce ne soit pas le prouvé.

I.3. Méthode de Simpson

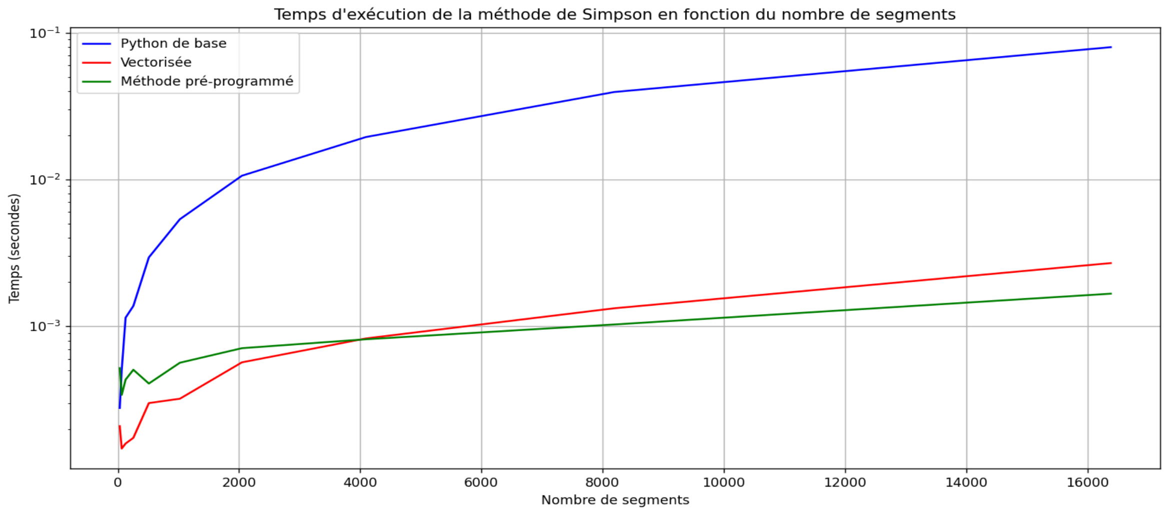
L'objectif de cette méthode est d'approximer l'intégrale d'une fonction en utilisant une parabole pour estimer l'aire sous la courbe. Plus précisément, la méthode de Simpson utilise une combinaison de la méthode des trapèzes et de la méthode des paraboles pour obtenir une approximation plus précise de l'intégrale en approximant un segment de la fonction par une parabole de degré 2.

Pour évaluer sa précision nous avons déterminé l'erreur entre cette méthode d'intégration numérique et la solution analytique (exacte).



Ces tests ont été effectué sur le polynôme x^3 + x^2 + x + 1 dans l'intervalle [0 ;1]. On constate alors que le résultat obtenu devient rapidement très précis. Dès que le nombre de segments dépasse 15, l'erreur devient presque nulle.

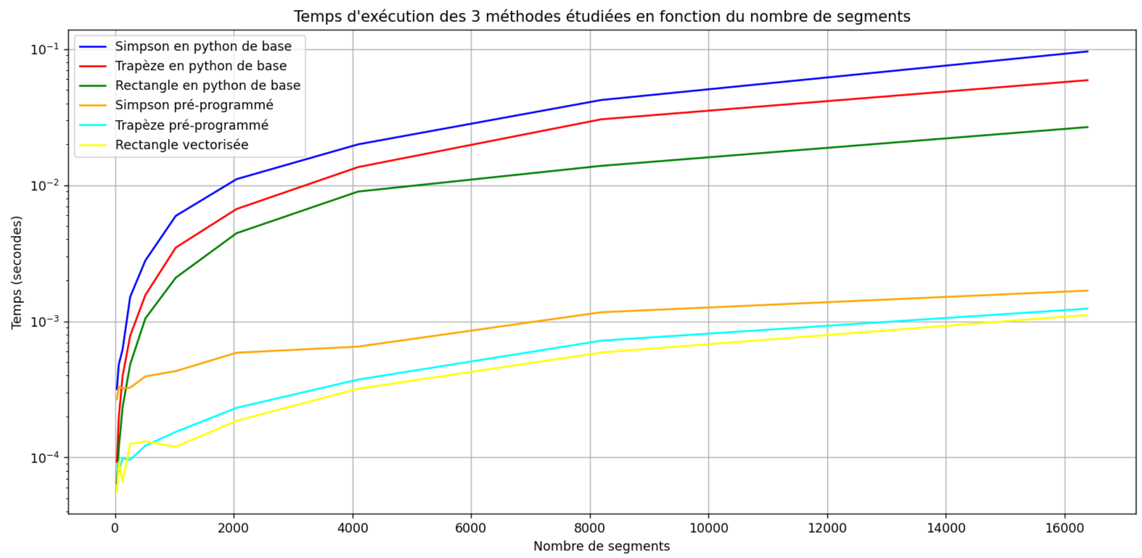
Nous avons ensuite décidé d'évaluer la rapidité d'exécution de la méthode de Simpson écrite en python de base, celle préprogrammé dans le paquet Scipy et une méthode que nous avons vectorisée. Pour cela nous avons tracer les courbes du temps d'exécution en fonction du nombre de segments entrés en paramètre de chaque fonction



On constate alors que la complexité de la fonction écrite en python de base est bien plus importante que la fonction vectorisée (pour 8000 segments, la méthode en python de base s'exécute en 0.02s tandis que la méthode vectorisée s'exécute presque 20 fois plus rapidement en 0.001s) même si la dynamique des courbes entre ces deux méthodes est semblable. La complexité de la méthode pré-programmée à partir du module 'Scipy' est la plus faible et possède une dynamique plus linéaire que les deux autres. L'efficacité de la méthode Scipy s'explique par la vectorisation, leurs permettant de traiter des ensembles de valeurs plutôt qu'une seule valeur à la fois.

II. Comparaison des 3 méthodes entre-elles

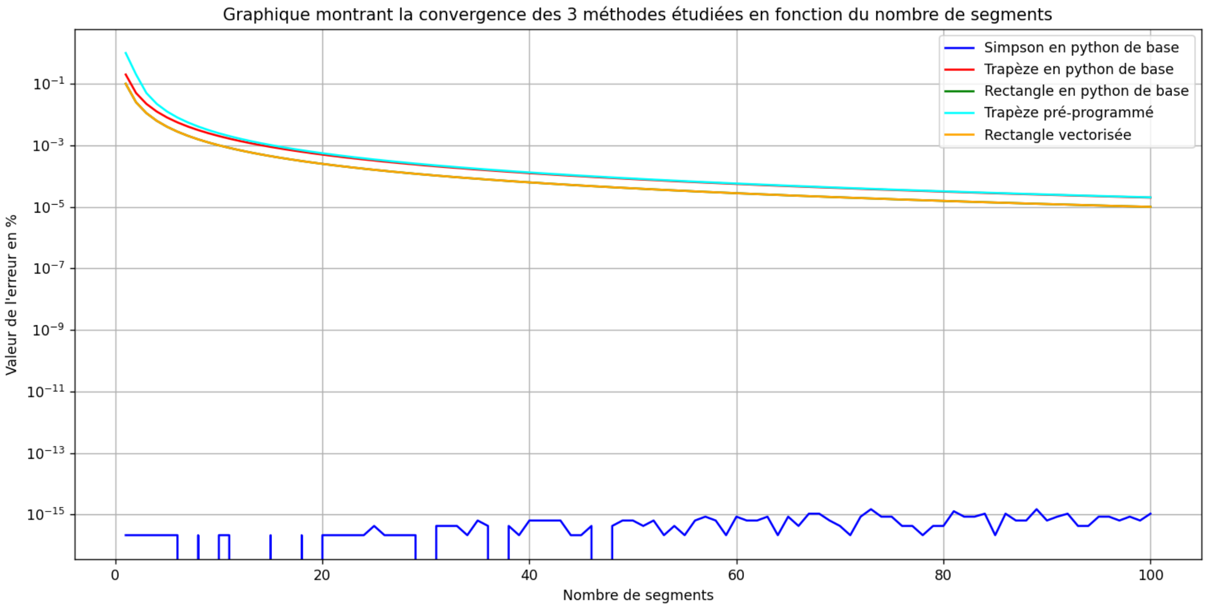
II.1 Comparaison de la complexité



Ce graphique présente le temps d'exécution des 3 méthodes vues précédemment pour une implémentation en python de base et vectorisée.
On constate alors que le mécanisme de vectorisation est l'élément le plus important à considérer pour augmenter la rapidité d'exécution d'une fonction, en effet, pour une intégration d'environ 10000 segments, les méthodes vectorisées sont environ 30 fois plus rapide que les méthodes implémentées en python de base (en moyenne 0.001s de temps d'exécution avec la vectorisation contre 0.03s en python de base).

On constate aussi que quel que soit le type d'implémentation, la méthode la plus rapide est celle des rectangles, elle demande en effet moins de calculs que la méthode des trapèzes et de Simpson.

II.2 Comparaison de la précision



Enfin, ce graphique nous montre la convergence des 3 méthodes étudiées, on constate dans un premier temps que le mécanisme de vectorisation, bien que très efficace pour améliorer le temps d'exécution, n'impact pas la précision des calculs. De plus, on remarque que la méthode des trapèzes et la méthode des rectangles ont une précision très similaire. La dynamique des courbes de convergence est la même pour les 2 méthodes, à partir de 15 segments, la valeur de l'erreur d'intégration passe sous les 10^(-3) % jusqu'à atteindre 10^(-5) % pour 100 segments.

La méthode de Simpson est quant à elle beaucoup plus efficace, la valeur de l'erreur est en moyenne d'environ 10^(-15) %, quel que soit le nombre de segments. Cette méthode est donc la plus précise sur le calcul d'une l'intégrale d'un polynôme de degrés 3 mais son temps d'exécution est le plus élevé.