Documentation de la Bibliothèque Standard - Extension math

GL 03

M. Migné[*] E. Tellier[†] A. El Ahmar[‡] A. Benmoussa[§]
A. Fuentes[¶] Y. Amoura[‖] J. Vandeputte[**]
F. Constans[††]

21 juin 2023

[*] Matthieu.Migne@grenoble-inp.org>
[†] Emilien.Tellier@grenoble-inp.org
[‡] Anas.Elahmar@grenoble-inp.org
[§] Alae.Benmoussa@grenoble-inp.org
[¶] Aubin.Fuentes@grenoble-inp.org
[‖] Yanis.Amoura@grenoble-inp.org
[**] Jules.Vandeputte@grenoble-inp.org
[††] Florence.Constans@grenoble-inp.org

Table des matières

1 Algorithmes utilisés et choix de conception

1.1 sin

Cette fonction a été implémenté en utilisant une série de Taylor pour approximer sa valeur. L'algorithme utilisé est une version basique de la méthode de réduction d'angle et de la série de Taylor pour le calcul du sinus.

Voici une explication détaillée de chaque partie de l'algorithme :

1. Fonction reduceAngle(float x) Cette fonction est utilisée pour réduire l'angle d'entrée x à la plage [-pi, pi]. Cela se fait en utilisant l'opérateur modulo (

Ensuite, la fonction vérifie si l'angle réduit x est supérieur à pi. Si c'est le cas, cela signifie que l'angle initial était dans la plage [pi, 2*pi], et nous devons le ramener dans la plage [-pi, 0]. Pour cela, on soustrait 2*pi à x.

Enfin, si l'angle réduit x est inférieur à -pi, cela signifie que l'angle initial était dans la plage [-2*pi, -pi], et nous devons le ramener dans la plage [0, pi]. Pour cela, on ajoute 2*pi à x.

Cette fonction de réduction d'angle est importante car elle permet de normaliser l'angle d'entrée pour garantir une approximation correcte de la fonction sinus.

2. Fonction sin(float f) Cette fonction calcule le sinus de l'angle f en utilisant la série de Taylor. Tout d'abord, elle appelle la fonction reduceAngle(f) pour réduire l'angle à la plage [-pi, pi].

Ensuite, elle initialise les variables nécessaires : - result : la variable qui stocke le résultat du calcul du sinus - term : la variable qui stocke le terme courant de la série de Taylor - power : la variable qui stocke la puissance courante de l'angle f - factorial : la variable qui stocke le factoriel courant utilisé dans le calcul de chaque terme de la série de Taylor - sign : la variable qui stocke le signe courant du terme de la série de Taylor (-1 ou 1) - i : la variable qui stocke l'indice courant du terme de la série de Taylor

Ensuite, une boucle while est utilisée pour calculer les termes de la série de Taylor jusqu'à un certain nombre d'itérations (dans cet exemple, jusqu'à i $<=$ 10).

À chaque itération de la boucle, le terme courant est ajouté au résultat en utilisant le signe, la puissance, et le factoriel. Ensuite, les variables power, factorial, sign, i, et term sont mises à jour pour la prochaine itération.

Finalement, lorsque la boucle se termine, le résultat approximatif du sinus est retourné.

Cependant, il est important de noter que cette implémentation de la fonction sinus est une simplification et n'est pas aussi précise que les algorithmes plus sophistiqués utilisés dans les bibliothèques mathématiques standard. Cet exemple a été donné à titre illustratif pour montrer le concept de la série de Taylor et de la réduction d'angle utilisés dans le calcul du sinus.

1.2 atan

Cette fonction a été implémenté en utilisant une approximation de la série de Taylor. L'algorithme utilise une méthode d'approximation pour calculer la valeur de l'arctangente.

Voici une explication détaillée de chaque partie de l'algorithme :

1. Fonction _atan_approximation(float f) Cette fonction calcule une approximation de l'arctangente en utilisant la série de Taylor. Elle prend en paramètre un nombre f pour lequel on souhaite calculer l'arctangente.

La variable "iterations" définit le nombre d'itérations utilisées pour l'approximation de la série de Taylor.

Les variables nécessaires sont initialisées : - result : la variable qui stocke le résultat de l'approximation de l'arctangente - power : la variable qui stocke la puissance courante de f - divisor : la variable qui stocke le diviseur courant utilisé dans le calcul de chaque terme de la série de Taylor - is_addition : un indicateur booléen pour alterner entre l'addition et la soustraction des termes de la série de Taylor - i : la variable qui stocke l'indice courant du terme de la série de Taylor

Ensuite, une boucle while est utilisée pour calculer les termes de la série de Taylor jusqu'à atteindre le nombre d'itérations spécifié.

À chaque itération de la boucle, le terme courant est calculé en utilisant la puissance et le diviseur, puis ajouté ou soustrait au résultat en fonction de la valeur de "is_addition". Les variables power, divisor et is_addition sont mises à jour pour la prochaine itération.

Finalement, lorsque la boucle se termine, le résultat de l'approximation de l'arctangente est retourné.

2. Fonction atan(float f) Cette fonction calcule l'arctangente de f en utilisant l'approximation de la série de Taylor.

Tout d'abord, la fonction vérifie si f est égal à zéro. Si c'est le cas, elle retourne simplement zéro, car l'arctangente de zéro est zéro.

Ensuite, la fonction détermine le signe de f en utilisant une condition. Si f est supérieur à zéro, le signe est défini sur 1. Sinon, le signe est défini sur -1.

La valeur absolue de f est calculée en utilisant une fonction _abs(f) (qui n'est pas donnée dans cet extrait de code), afin de travailler uniquement avec des valeurs positives pour l'approximation de l'arctangente.

La fonction vérifie si la valeur absolue de f est supérieure à 1. Si c'est le cas, f est inversé (1/f) pour se situer dans la plage [-1, 1], puis la valeur de l'arctangente est calculée en utilisant (_pi / 2) - _atan_approximation(f). Le résultat est multiplié par le signe approprié.

Si la valeur absolue de f est inférieure ou égale à 1, l'arctangente est calculée directement en utilisant _atan_approximation(f), et le résultat est multiplié par le signe approprié.

Enfin, la valeur de l'arctangente calculée est retournée.

Il est important de noter que cette implémentation de la fonction arctangente utilise une approximation de la série de Taylor, ce qui peut entraîner des erreurs d'approximation.

1.3 arcsin

1.4 cosinus

1.5 ulp

2 Précision théorique et pratique des algorithmes

3 Méthode de validation

4 Pistes d'amélioration des algorithmes

4.1 sin

Voici quelques pistes d'amélioration possibles pour cet algorithme :

1. Augmenter le nombre d'itérations de la boucle : Dans l'exemple donné, la boucle effectue 10 itérations pour calculer la série de Taylor. En augmentant ce nombre, on pourrait obtenir une meilleure précision de l'approximation du sinus. Cependant, il est important de noter que l'augmentation du nombre d'itérations entraînera également une augmentation du temps de calcul.

2. Utiliser une précision numérique supérieure : La représentation en virgule flottante (float) utilisée dans cet exemple peut entraîner des erreurs d'arrondi et une perte de précision. En utilisant une précision numérique supérieure, telle que double, on pourrait obtenir une meilleure précision dans le calcul du sinus.

4. Utiliser des améliorations spécifiques à la série de Taylor : La série de Taylor utilisée dans cet exemple est une approximation basique. Cependant, il existe des variantes plus optimisées de la série de Taylor, telles que la série de Taylor de Maclaurin, qui peuvent converger plus rapidement et offrir une meilleure précision.

5. Utiliser des méthodes d'interpolation : Au lieu de calculer le sinus directement à partir d'une série de Taylor, on pourrait utiliser des méthodes d'interpolation, telles que l'interpolation polynomiale ou l'interpolation par splines, pour obtenir une meilleure approximation du sinus. Ces méthodes permettent de mieux ajuster les valeurs entre les points de la série de Taylor et d'obtenir une courbe plus lisse.

4.2 atan

1. Augmenter le nombre d'itérations de la boucle : Dans l'exemple donné, la boucle effectue 5 itérations pour calculer l'approximation de la série de Taylor. En augmentant ce nombre, on pourrait obtenir une meilleure précision dans le calcul de l'arctangente. Cependant, cela entraînera également une augmentation du temps de calcul.

2. Utiliser une méthode d'approximation plus précise : La série de Taylor est une approximation basique qui converge plus lentement pour des valeurs éloignées de zéro. on pourrait utiliser des méthodes d'approximation plus précises, telles que l'algorithme de CORDIC (Coordinate Rotation Digital Computer), qui offre une meilleure convergence et précision pour le calcul de l'arctangente.

3. Utiliser des formules trigonométriques ou des identités trigonométriques : Au lieu d'utiliser la série de Taylor pour approximer l'arctangente, on pourrait utiliser des formules trigonométriques ou des identités trigonométriques pour exprimer l'arctangente en termes d'autres fonctions trigonométriques. Par exemple, on pourrait utiliser l'identité $atan(x) = atan(y) - atan(z)$ avec $y = x + sqrt(1 + x^2)$ et $z = sqrt(1 + x^2)$ pour calculer l'arctangente de manière plus précise.

4. Considérer les propriétés de symétrie et de périodicité de l'arctangente : L'arctangente a des propriétés de symétrie et de périodicité qui peuvent être exploitées pour réduire l'intervalle de calcul ou améliorer l'efficacité du calcul. Par exemple, on

pourrait utiliser l'identité atan(x) = -atan(-x) pour réduire l'intervalle de calcul à [-1, 1].

5. Utiliser des méthodes d'interpolation : Au lieu d'utiliser une approximation par série de Taylor, on pourrait également envisager d'utiliser des méthodes d'interpolation, telles que l'interpolation polynomiale ou l'interpolation par splines, pour obtenir une meilleure approximation de l'arctangente sur un intervalle donné.

4.3 arcsin

4.4 cosinus

4.5 ulp