Université de Lyon - Université Lyon 1 | Master Informatique - M1
Programmation avancée C++ - M1if23
http ://liris.cnrs.fr/~ rchaine/COURS/M1if23_Fall24/ | 2024 - 2025

INTERPOL (à déposer dans Tomuss en fin de séance)

1 Conditions de l'épreuve

Le travail est à faire tout seul, sans aucune interaction avec vos voisins, et sans explications de votre chargé de TP, pendant 2h30. Vous êtes placé dans la salle qui vous a été affectée dans Tomuss pour cette épreuve. Si vous n'avez pas le temps de finir le sujet, ce n'est pas grave, il est volontairement long pour se prêter à l'interrogation de TP du 12 décembre.

Vous avez uniquement droit :
— à une page de votre navigateur ouverte sur Tomuss (AUCUNE AUTRE PAGE AUTORISEE, Y COMPRIS LA PAGE WEB DU COURS),
— à un lecteur de pdf ouvert sur les SLIDES DU COURS PRECEDEMMENTS RAPATRIES ainsi que les sujets pdf de TP de l'UE (ces supports peuvent également être imprimés),
— à votre éditeur de code préféré ouvert sur un projet vide, non relié à un gestionnaire de version style git.
— à votre compilateur C++.
Cela signifie que vous n'avez pas droit à internet, ni à vos codes écrits précédemment (ni aux corrections fournies), ni à copilot dans visual studio code (ou autre assistant de programmation). Votre application de mail ou de tchat doit également être fermée, ainsi que votre téléphone portable.

Attention : Le chargé de TP vérifiera tout au long en de la séance que vous respectez bien ces règles (liste des processus en train de tourner sur votre machine, fichiers précédemment ouverts dans votre éditeur, penser bien à tout purger avant le début de la séance). Votre écran doit rester visible tout le temps de la séance afin que le chargé de TP puisse constater l'évolution progressive de votre code, avec une taille suffisamment grande de la fonte.

A la fin de la séance, vous déposerez une archive de votre travail sur Tomuss :
— Attention de ne mettre que les `.h` et `.cpp`. Le `Makefile` n'est pas obligatoire, mais ce n'est que quelques lignes de plus à écrire pour une compilation plus rapide et plus écologique. Vous ajouterez également un README où vous expliquerez ce que vous avez fait.
— L'archive (`.zip` ou `.gz` ou `.tgz`) que vous déposez doit porter votre nom.

Le sujet est dans la lignée des travaux faits pendant les séances de TP. Vous ne pourrez recourir aux `vector`, `deque`, `list` et `slist` de la STL que quand c'est spécifié. Vous ne pourrez pas non plus utiliser les pointeurs intelligents proposés dans la bibliothèque standard depuis `C++11`. En revanche, vous pourrez choisir de coder en utilisant ou non des éléments du langage correspondant aux normes postérieures à `C++11`.

2 Interpolation

2.1 Introduction

En économie, géographie, météorologie et toutes les sciences où on cherche à connaître le profil du comportement d'une fonction à partir de quelques valeurs ou observations éparpillées, on utilise un mécanisme d'interpolation. Ces observations peuvent dépendre de plusieurs paramètres. Par exemple en météorologie, on ne connaît la température qu'aux endroits (caractérisés par leurs coordonnées sur le globe terrestre) où sont placés les capteurs et où ont été enregistrées des mesures. Si la température est enregistrée à des périodes différentes de l'année suivant les points de captages, on ajoute une coordonnée temporelle aux coordonnées spatiales. On désire ensuite extrapoler cette température en n'importe quel autre point du territoire, et à n'importe quel moment de l'année. Autrement dit, on cherche à reconstruire une fonction f de N variables réelles x_1, x_2, .., x_N (ici la longitude, la latitude et le moment de l'année). Il s'agit donc d'une fonction définie de $\mathbb{R}^N$ dans $\mathbb{R}$.

Exemple :
- si $N = 2$, il s'agit d'une fonction définie dans le plan,
- si $N = 3$, il s'agit d'une fonction définie dans l'espace,
- mais on peut également être confrontés à des valeurs de N plus grandes, lorsqu'on travaille dans des espaces de plus grande dimension.

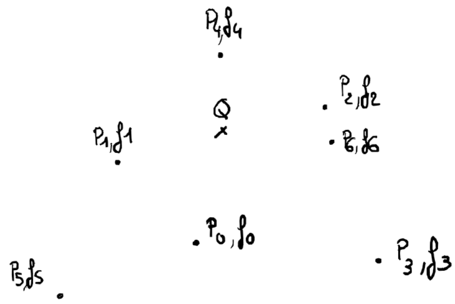

FIGURE 1 – Etant donné un ensemble de points P_i pour lesquels on connaît le relevé f_i d'une fonction f, on souhaite estimer la valeur $f(Q)$ en un point Q donné par l'utilisateur (exemple en dimension $N = 2$).

On connait la valeur de f en K échantillons P_1, P_2, .., P_K et on note $f_i = f(P_i)$ les observations correspondantes. Le point P_i est caractérisé par les coordonnées $x_{i,1}, x_{i,2}, .., x_{i,N}$. Etant donné un point $Q(x_1, x_2, .., x_N)$, on souhaite connaître une approximation de la valeur de f en Q. On procède pour celà à une interpolation des valeurs connues de f sur les échantillons en accordant plus d'importance aux mesures sur les voisins les plus proches.

Travail à faire Ecrivez une classe vecteur de dimension N qui contiendra en donnée membre un tableau statique (ou un `std::array`) de ses N coordonnées (en `double`). Cette classe sera `template` quand à la dimension N. Vous munirez la classe des constructeurs et autres opérations nécessaires, sans faire de zèle si le langage fait très bien les choses à votre place. Votre classe vecteur pourra aussi bien être utilisée pour stocker des points que des vecteurs, du coup vous pourrez prévoir un constructeur à partir de deux vecteurs V1 et V2 qui calculera la différence entre les deux vecteurs V2 et V1 pour initialiser son argument implicite. Le constructeur qui prend les coordonnées en paramètre est plus difficile à écrire, car ces paramètres sont au nombre de N. Si vous n'avez pas le temps aujourd'hui d'utiliser les templates variadiques transmettez plus simplement un tableau de valeurs à votre constructeur.

Testez l'utilisation de votre classe dans un petit programme de test (mainTestVecteur), où vous construirez par exemple des `Vecteur<2>` ou des `Vecteur<5>`.

2.2 Interpolation dans un simplexe

En dimension N, il existe un moyen très simple pour interpoler les valeurs de f en $N+1$ points, au sein d'un espace délimité par ces $N+1$ points, on utilise pour cela les coordonnées barycentriques.

Simplexe de dimension N En dimension $n \geq N$, un N-simplexe délimité par $N+1$ points est le plus petit convexe de dimension N contenant les $N+1$ points.

Exemple :
— un 1-simplexe est un segment s'appuyant sur 2 points,
— un 2-simplexe est un triangle s'appuyant sur 3 points,
— un 3-simplexe est un tétraèdre s'appuyant sur 4 points,
— et ainsi de suite.... un N-simplexe s'appuyant sur N+1 points.

Travail à faire Ecrire une classe simplexe de dimension N qui contiendra comme donnée membre un tableau de ses $N+1$ sommets. On se restreindra au cas où les sommets sont des points en dimension N, en utilisant la classe vecteur<N>. Vous munirez la classe des constructeurs et autres opérations nécessaires. En particulier, n'oubliez pas le constructeur qui prend les $N+1$ points en paramètre (l'usage de template variadique peut-être remplacé par le simple passage d'un tableau).

Volume signé d'un simplexe de dimension N Le volume du simplexe $D = \{P_{S0}, P_{S1}, ..., P_{SN}\}$ est $Vol(D) = \frac{1}{N!} det(P_{S1} - P_{S0}, P_{S2} - P_{S0}, .., P_{SN} - P_{S0})$

Par exemple, en dimension 2, le volume signé d'un triangle délimité par 3 points P_{S0}, P_{S1} et P_{S2} est la moitié de la valeur du déterminant des 2 vecteurs $P_{S1} - P_{S0}$ et $P_{S2} - P_{S0}$.

Le signe du volume d'un simplexe dépend de son orientation. Si vous permutez 2 points, vous modifiez l'orientation de votre simplexe et son volume change de signe, en revanche si vous permutez un nombre pair de paires de points, l'orientation de votre simplexe reste inchangée ainsi que son volume.

Calcul du déterminant de N vecteurs de dimension N Pour calculer le déterminant de N vecteurs de dimension N, on peut exploiter la nature récursive du calcul d'un déterminant.

Calcul du déterminant de 2 vecteurs $W_1(w_{1,1}, w_{1,2})$ et $W_2(w_{2,1}, w_{2,2})$ en dimension 2 :

$$det(W_1, W_2) = w_{1,1} * w_{2,2} - w_{1,2} * w_{2,1}$$

Calcul du déterminant de N vecteurs $W_1(w_{1,1}, w_{1,2}, .., w_{1,N}), W_2(w_{2,1}, w_{2,2}, .., w_{2,N}), .., W_N(w_{N,1}, w_{N,2}, .., w_{N,N})$ en dimension $N \geq 3$:

$$det(W_1, W_2, .., W_N) = \sum_{k=1}^{k=N} (-1)^{(k-1)} * w_{1,k} * det(tronq(W_2, k), tronq(W_3, k), .., tronq(W_N, k)))$$

où $tronq(W, k)$ est une fonction qui fabrique un vecteur de dimension $N-1$ en retirant la k-ième coordonnée d'un vecteur W de dimension N. Le calcul d'un déterminant en dimension N se ramène donc au calcul de N déterminants en dimension $N-1$.

Travail à faire Ecrire une fonction déterminant, templatée par le nombre N de `Vecteur<N>` qu'elle prend en paramètre (l'usage de template variadique peut-être remplacé par le simple passage d'un tableau). Pour implémenter la fonction déterminant, vous aurez également besoin d'implémenter la fonction `tronq`.

Testez l'utilisation de votre implémentation de la fonction déterminant dans un petit programme de test (mainTestDet).

Travail à faire Munissez votre classe Simplexe d'une fonction de calcul de volume signé.

Testez l'utilisation de votre classe Simplexe et de son calcul de volume dans un petit programme de test (mainTestSimplexe).

Coordonnées barycentriques d'un point Q inclus dans un simplexe de dimension N Un point Q est inclus dans un simplexe $D = \{P_{S0}, P_{S1}, ..., P_{SN}\}$ si et seulement si le volume signé $Vol(Q, i, D)$ du simplexe D modifiié en remplaçant le ième point P_{Si} par Q est du même signe que le volume de D, ceci pour toute valeur de i comprise entre 0 et N.

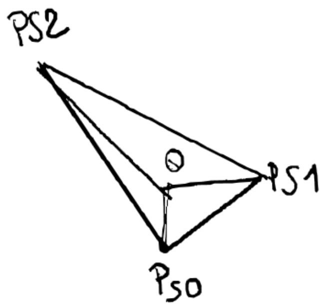

FIGURE 2 – Les coordonnées barycentriques du point Q par rapport aux sommets du Simplexe $D = \{P_{S0}, P_{S1}, P_{S2}\}$ sont calculées en effectuant des rapports de volumes. Ici $\lambda_0 = \frac{Vol(Simplexe(Q,P_{S1},P_{S2}))}{Vol(Simplexe(P_{S0},P_{S1},P_{S2}))}$, $\lambda_1 = \frac{Vol(Simplexe(P_{S0},Q,P_{S2}))}{Vol(Simplexe(P_{S0},P_{S1},P_{S2}))}$, $\lambda_2 = \frac{Vol(Simplexe(P_{S0},P_{S1},Q))}{Vol(Simplexe(P_{S0},P_{S1},P_{S2}))}$.

Les coordonnées barycentriques du point Q par rapport au sommet P_{Si} s'obtiennent alors en calculant le ratio de volume $\lambda_i = \frac{Vol(Q,i,D)}{Vol(D)}$, pour toute valeur de i entre 0 et N.

Interpolation dans un simplexe en dimension N Il est possible d'interpoler les valeurs $f_i = f(P_i)$ des sommets d'un simplexe D en un point Q (inclus dans le simplexe), en faisant la moyenne des $N + 1$ valeurs f_i pondérées par les coordonnées barycentriques.

$$f(Q, D) = \sum_{i=0}^{i=N} \lambda_i f_i$$

La qualité de cette interpolation sera meilleure si tous les sommets du simplexe D sont proches de Q. On peut donc mesurer la qualité en considérant l'inverse de la distance d_Q de Q au sommet de D le plus éloigné. On préfère également les simplexes les plus équilatéraux possibles, ce que l'on mesure en évaluant la plus courte distance d entre des sommets du Simplexe et en évaluant le ratio $\frac{d^N}{|Vol(D)|}$. Il est possible de faire une mesure combinée de ces deux qualités en calculant $\frac{d^N}{|Vol(D)*d_Q|}$.

Travail à faire Munir la classe Simplexe d'une fonction qui teste l'inclusion d'un point Q dans son intérieur et d'une fonction qui calcule la iième coordonnée barycentrique d'un point Q dans le simplexe.

Testez l'implémentation des coordonnées barycentriques dans un petit programme de test (mainTestCoordonnées) et servez-vous en pour interpoler des valeurs définies au sommet d'un triangle (simplexe de dimension 2).

2.3 Interpolation d'un nombre $n >> N$ de valeurs dans un espace de dimension N

Etant donné un ensemble de valeurs f_i associées à des points P_i, on souhaite interpoler ces valeurs au point Q si c'est possible.

Pour cela on répète le processus suivant :

- Tirage au sort de $N+1$ points parmi les n pour former un Simplexe D de dimension N.
- Le point Q est-il inclus dans le Simplexe D tiré au sort ?
- Si OUI : Calcul de l'interpolation au point Q des valeurs f_i associées aux sommets de D, et mesure de la qualité *qual* de cette interpolation : $qual = \frac{d^N}{|Vol(D)*d_Q|}$

On fait la moyenne des interpolations obtenues au point Q en les pondérant en fonction de leur qualité (bien penser à normaliser la somme des pondérations à 1).

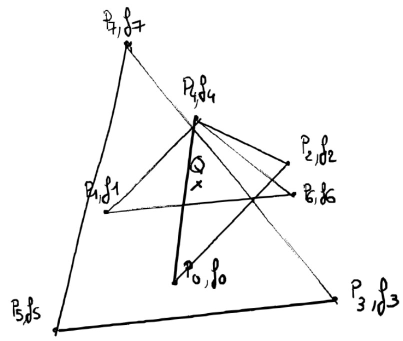

FIGURE 3 – On réalise plusieurs interpolations de $f(Q)$ en utilisant des Simplexes formés au hasard. On moyenne ensuite les valeurs obtenues, en pondérant par la qualité des Siimplexes.

Si le point Q n'appartient à aucun des Simplexes tirés au sort, c'est sans doute qu'il est à l'extérieur de l'enveloppe convexe de l'ensemble des points P_i, on ne peut alors pas fournir d'interpolation.

Rappel : La fonction `int std::rand(void)` (bibliothèque `stdlib`) permet de renvoyer une valeur entre 0 et `RAND_MAX`.

Travail à faire Etant donné un ensemble de valeurs f_i aux points P_i, écrire un petit programme qui fournit l'interpolation des f_i en n'importe quel point Q proposé par l'utilisateur.

Travail à faire Ecrire un petit programme où vous tirez au sort un ensemble de points P_i en 2D avec pour valeur de f_i le cosinus de l'abscisse de P_i. Interpolez ces valeurs sur tous les points Q d'une grille 2D, et affichez les sur la sortie standard.

2.4 Recommandations

Votre travail est décomposé en plusieurs étapes. Merci de bien placer le code des programmes de test résultant de chacune des étapes dans des fichiers différents. Chacun de ces programme principaux devra illustrer le bon fonctionnement des éléments mis en place. Il devra en particulier permettre l'instanciation et l'appel de l'ensemble des fonctions membres. Vous détaillerez ce que vous avez eu le temps de faire dans un README. Attention de bien vérifier que votre code correspond à une gestion saine de l'espace mémoire.

2.5 Après dépot dans Tomuss

Maintenant vous avez bien mérité de passer une bonne soirée !