Master IFMA
2016-2017

Programmation en C++ : projets

Sujet 1 : Marche sur les sphères et problème de Dirichlet
Mots-clés : *Mouvement Brownien, temps d'arrêt, équations aux dérivées partielles eliptiques.*

On cherche à approcher numériquement la solution de l'équation

$$\begin{cases} \Delta u(x) = 0 & \text{si } x \in \Omega, \\ u(x) = f(x) & \text{si } x \in \partial\Omega, \end{cases}$$

où Ω est un ouvert connexe borné de $\mathbb{R}^d$, f est une fonction continue sur $\partial\Omega$ et Δ désigne l'opérateur Laplacien. On admettra que cette équation admet une unique solution u. La fonction u admet alors une interprétation probabiliste. En effet $u(x)$ est donné par $u(x) = E_x W_\tau$, où W est un mouvement Brownien, τ désigne le temps de sortie de l'ouvert Ω pour W, et E_x désigne l'espérance conditionnelle sachant $W_0 = x$. On est donc capable de donner une approximation de $u(x)$ en tout point par une méthode de Monte-Carlo, si on sait simuler W_τ.

Pour simuler W_τ, on peut utiliser la méthode de *marche sur les sphères* :

1. On part d'un point $x \in \Omega$;
2. On calcule le plus grand r tel que $B(x, r) \subset \Omega$;
3. Le mouvement Brownien issu de x quitte $B(x, r)$ en un point uniformément réparti sur la sphère de centre x et de rayon r. On tire donc un point uniformément sur cette sphère ;
4. On recommence en partant du nouveau point.
5. On arrête l'algorithme quand on est à une distance suffisamment petite du bord.

On pourra par exemple donner une approximation de la solution pour $\Omega = [0, 1]^2$, avec $f(x_1, x_2) = x_1$, $f(x_1, x_2) = x_1 x_2$ ou $f(x_1, x_2) = |x_1 - x_2|$. Remarque : en fait, on a pour tout x l'encadrement $\inf_{\partial\Omega} f \leq u(x) \leq \sup_{\partial\Omega} f$, de sorte qu'il est simple de donner un intervalle de confiance pour $u(x)$.

Sujet 2 : Simulation exacte d'EDS par méthode du rejet
Mots-clés : *Processus de Poisson, mouvement Brownien, équations différentielles stochastiques*

Il est possible de simuler exactement la solution de l'équation différentielle stochastique

$$dX_t = dW_t + b(X_t)dt,$$

à l'aide d'un méthode de rejet, sous certaines hypothèses sur la fonction b. Ici, on supposera $\varphi = (b^2 + b')/2$ bornée : $\alpha \leq \varphi \leq \beta$. On note également B une primitive de b. L'algorithme est le suivant pour simuler une trajectoire issue de x_0 jusqu'au temps T :

1. On tire une réalisation $(t_i, x_i)_{i=1,\ldots,N}$ d'un processus de Poisson dans $[0, T] \times [\alpha, \beta]$;
2. On tire une réalisation W_T de la loi $Ce^{B(x)}e^{-(x-x_0)^2/2T}$;
3. On simule les valeurs en les t_i d'un mouvement Brownien W valant W_T au temps T ;
4. (a) Si tous les W_{t_i}, x_i vérifient $\varphi(W_{t_i}) \leq x_i$, la trajectoire W obtenue est une solution de l'équation (une fois complétée par des ponts Browniens).

(b) Sinon, on recommence du début.

On pourra par exemple simuler exactement une trajectoire dans le cas $b = \cos$.

Sujet 3 : Développements limités
Mots-clés : *Développement limités, calcul formel*

Toute fonction de classe $\mathcal{C}^\infty$ admet un développement limité à un ordre quelconque. Le but sera de programmer une classe correspondant aux développements limités : un objet de la classe contiendra une valeur n correspondant à l'ordre auquel le développement limité est poussé, et un tableau de taille $n+1$ correspondant au coefficients du développement.

On implémentera les développements limités des fonctions exp, ln, sin, cos, $\frac{1}{1-x}$, des polynômes, de la fonction réciproque d'une fonction donnée.

Par exemple, on pourra écrire un programme permettant de calculer la limite de

$$\frac{\sin(\arcsin(x)) - \tan(\arctan(x))}{\arcsin(\sin(x)) - \arctan(\tan(x))}.$$

Sujet 4 : Calcul différentiel formel
Mots-clés : *Dérivation*

Le calcul de la dérivée d'une fonction "usuelle" se fait de manière algorithmique. En effet, les formules $(f \circ g)' = (f' \circ g) \times g'$, $(f \times g)' = f' \times g + f \times g'$ et $(\lambda f + \mu g)' = \lambda f' + \mu g'$ permettent de calculer la dérivée de toute fonction construite à partir de fonctions de base (par exemple, les polynômes, les fonctions trigonométriques, les exponentielles et les logarithes). En représentant une fonction quelconque sous la forme d'un arbre, on pourra donc calculer la dérivée d'une fonction quelconque.

Sujet 5 : Matrices creuses
Mots-clés : *Algèbre linéaire*

Dans les applications, on peut être amenés à faire de l'analyse numérique sur des matrices de très grande taille. Par exemple, une matrice 1000×1000 possède un million de coefficients. Un coefficient étant usuellement codé sur 8 octets, une seule matrice occupe donc 8 méga-octets.

Or en pratique, on a souvent des hypothèses sur la forme de la matrice. Par exemple, la matrice du Laplacien discret en dimension 1 est tridiagonale. Une telle matrice, de taille 1000×1000 n'aura donc que 2998 coefficients non nuls (sur 1000000 de coefficients).

On programmera une classe permettant de représenter les matrices tridiagonales, de calculer leur déterminant, d'inverser un système linéaire les faisant intervenir, et de calculer leurs valeurs/vecteurs propres.

Sujet 6 : Algorithme de Propp-Wilson
Mots-clés : *Chaînes de Markov, mesures invariantes*

L'algorithme de "couplage par le passé", de Propp-Wilson permet de simuler *exactement* une variable aléatoire dont la loi est la loi invariante d'une chaîne de Markov (supposée irréductible, sur un espace d'états E fini). Le principe est le suivant : on se donne une chaîne de Markov dont les transitions sont de la forme

$$X_{n+1} = f(X_n, \Theta_n) = f_{\Theta_n}(X_n),$$

où la fonction $f : E \times [0,1] \to E$ est mesurable, et Θ_n est une suite de variables aléatoires indépendantes et de même loi.

On définit alors la suite de fonctions de E dans E :

$$\begin{cases} F_0 & = id, \\ F_{n+1} & = F_n \circ f_{\Theta_n}. \end{cases}$$

On calcule alors la suite F_n jusqu'à obtenir une fonction constante. La valeur de cette constante est alors une variable aléatoire dont la loi est la loi invariante de (X_n).

On peut par exemple simuler la loi invariante du modèle de l'urne d'Ehrenfest (la loi invariante doit être la loi binomiale).

Sujet 7 : Voyageur de commerce et recuit simulé
Mots-clés : *Minimisation de fonction, chaîne de Markov*

De nombreux problèmes de minimisation parcourent un ensemble trop grand pour être parcouru intégralement. Un exemple est le problème dit du *voyageur de commerce* : on doit parcourir n villes, en minimisant le coût du voyage. Plus formellement, on dispose de réels $(A_{ij})_{1\leq i<j\leq n}$, où A_{ij} représente la distance entre les villes i et j. On se donne un trajet σ (= une permutation de $\{1,\dots,n\}$, $\sigma(i)$ est la $i$ème ville visitée), et on cherche à minimiser $E(\sigma) = \sum_{k=1}^{n} A_{\sigma(i)\sigma(i+1)}$.

Une méthode probabiliste efficace est l'algorithme du *recuit simulé*, dont le principe est le suivant :

1. On choisit un trajet σ quelconque ;
2. On choisit au hasard deux villes i et j, et on définit le trajet $\tilde{\sigma}_{ij}$ obtenu en échangeant les villes i et j dans σ.
3. (a) Si $E(\tilde{\sigma}_{ij}) < E(\sigma)$ on recommence avec $\tilde{\sigma}_{ij}$ au lieu de σ ;
   (b) Sinon, on garde $\tilde{\sigma}_{ij}$ avec probabilité $p = \exp(\beta(E(\sigma) - E(\tilde{\sigma}_{ij})))$ et σ avec probabilité $1-p$ (β étant à choisir judicieusement).

Quand β tend progressivement vers ∞ (en $c\log(t)$, où t est le nombre d'itérations effectuées, et c suffisamment petit), l'algorithme converge presque sûrement vers le minimum de la fonction E.

Sujet 8 : Marches aléatoires ordonnées
Mots-clés : *Équation aux dérivées partielles, équation différentielle stochastique*

La solution de l'équation aux dérivées partielles

$$\partial_t u_t(x) = \sigma^2 \Delta u_t(x) + \partial_x(f(u_t(x)))$$

avec une condition initiale u_0 croissante peut être approchée par un système de particules de la manière suivante. Pour simplifier, on supposera que u_0 est une fonction de répartition. On considère $(\tilde{X}_0^1,\dots,\tilde{X}_0^n)$ des variables aléatoires indépendantes dont la loi a pour fonction de répartition u_0. On obtient $(X_0,\dots,X_0^n)$ à partir de $(\tilde{X}_0^1,\dots,\tilde{X}_0^n)$ en permutant les coordonnées de sorte à obtenir un vecteur ordonné :

$$X_0^1 \leq X_0^2 \leq \dots \leq X_0^n.$$

On effectue ensuite un schéma d'Euler de pas δ

$$\tilde{X}^j_{(i+1)\delta} = X^j_{i\delta} + \sigma\sqrt{2}(W^j_{(i+1)\delta} - W_{i\delta}) + \delta f'(X^j_{i\delta}),$$

puis on réordonne encore pour obtenir un vecteur $(X^1_{(i+1)\delta},\dots,X^n_{(i+1)\delta})$. On note $u_t^\delta(x) = \frac{1}{n}\sum_{j=1}^n \mathbf{1}_{X_t^j\leq x}$ la fonction de répartition empirique du système. Dans la limite $\delta \to 0$, cette fonction de répartition

converge vers la solution de l'équation. De plus une particule choisie aléatoirement converge vers un processus (X_t) vérifiant

$$dX_t = \sigma\sqrt{2}dW_t + f'(u_t(X_t))dt,$$

où u_t est la fonction de répartition de X_t et est la solution de l'EDP.

Sujet 9 : Compression d'images
Mots-clés : *Séries de Fourier*

La plupart des formats d'image classiques (jpg, png) sont des formats compressés, qui ne représentent pas une image pixel par pixel. À la place, on a calculé les coefficients de Fourier de l'image, vue comme une fonction de $[0,1]^2$ dans $[0,1]^3$ (l'image est un carré, et en chaque point, on donne la quantité de bleu, de rouge, et de vert), et on ne stocke que la série de Fourier tronquée à un certain ordre.

On programmera le calcul des coefficients de Fourier d'une image (noir et blanc, pour simplifiée) représentée sous forme de matrice. On pourra illustrer la décroissance des coefficients de Fourier de l'image, ainsi que l'image obtenue en supprimant un certain nombre de coefficients.

Sujet 10 : Réduction de variance
Mots-clés : *Variables Gaussiennes, méthode de Monte Carlo, algorithme du gradient*

On remarque que si G est une variable de loi normale standard d-dimensionnelle, alors on a, pour tout vecteur θ de $\mathbb{R}^d$

$$\mathbb{E}\left(f(G)\right) = \mathbb{E}\left(f(G+\theta)e^{-|\theta|^2/2-\theta\cdot G}\right).$$

On a donc deux variables aléatoires ($X_0 = f(G)$ et $X_\theta = f(G+\theta)e^{-|\theta|^2/2-\theta\cdot G}$) de mêmes moyennes. Cependant, la variance de X_θ est donnée par

$$\begin{aligned}\mathbb{E}X_\theta^2 - (\mathbb{E}X_\theta)^2 &= \mathbb{E}X_\theta^2 - (\mathbb{E}X_0)^2 \\ &= \mathbb{E}\left(f(G+\theta)^2 e^{-|\theta|^2-2\theta\cdot G}\right) - (\mathbb{E}X_0)^2 \\ &= \mathbb{E}\left(f(G)^2 e^{|\theta|^2/2-\theta\cdot G}\right) - (\mathbb{E}X_0)^2.\end{aligned}$$

Notamment, la variance de X_θ est minimale si

$$\Phi(\theta) = \mathbb{E}\left(f(G)^2 e^{|\theta|^2/2-\theta\cdot G}\right)$$

l'est. On remarque que la fonction Φ est une fonction convexe de θ, dont le gradient est

$$\nabla\Phi(\theta) = \mathbb{E}\left((\theta - G)f(G)^2 e^{|\theta|^2/2-\theta\cdot G}\right).$$

On calculera l'espérance $\mathbb{E}[f(G)]$ par une méthode de Monte-Carlo avec ou sans réduction de variance, et on comparera les intervalles de confiance obtenus. On pourra aussi chercher le θ optimal en effectuant une descente de gradient.

Sujet 11 : Percolation
Mots-clés : *Variables de Bernoulli, graphes*

On considère le réseau $\mathbb{Z}^2$, sur lequel point peut être "ouvert" ou "fermé". Pour cela, on représente un état du réseau par une fonction $\eta : \mathbb{Z}^2 \to \{0,1\}$, ou les points x tels que $\eta(x) = 1$ sont les points ouverts. On suppose que les $\eta(x)$ sont aléatoires de loi de Bernoulli de paramètre $0 < p < 1$.

Un *chemin issu de l'origine* est une suite $(x_n)_{n\in\mathbb{N}}$ de points de $\mathbb{Z}^2$ tels que $x_0 = (0,0)$, que $|x_{n+1} - x_n| = 1$ et que tous les x_n soient ouverts.

On a le résultat théorique suivant : pour $p <\leq 1/2$, on a une probabilité nulle qu'il existe un chemin issu de l'origine allant jusqu'à l'infini, alors qu'un tel chemin existe avec probabilité strictement positive si $p > 1/2$. On écrira un programme qui vérifie empiriquement ce fait (on remplacera les chemins allant jusqu'à l'infini par des chemins quittant une boîte $\{-N, N\}^2$).

Sujet 12 : Marche aléatoire à boucles supprimées
Mots-clés : *Marche aléatoire*

On considère une marche aléatoire symétrique $(X_n)_{n\in\mathbb{N}}$ dans $\mathbb{Z}^2$, dont on enlève les "boucle". Plus précisément, on calcule successivement les suites finies $(X_0, \dots, X_n)$, et à chaque fois que X_n est une valeur qui a déjà été prise par la boucle, on supprime la portion du trajet se situant entre ces deux instants. On se retrouve donc avec une marche aléatoire qui ne passe jamais deux fois au même point.

On arrête la marche au moment où elle sort d'un ensemble borné, par exemple $\{(x,y) \in \mathbb{Z}^2, x^2+y^2 \leq N^2\}$. Il se trouve que pour $N \to \infty$, la marche, correctement normalisée converge vers un processus à valeurs dans le disque de centre 0 et de rayon 1. On illustrera cette convergence, et on pourra estimer la dimension fractale de la courbe aobtenue.

Sujet 13 : Algorithme de Strassen
Mots-clés : *Matrices*

L'algorithme naïf de multiplication de deux matrices de taille n a un coût de l'ordre de n^3. En effet, on applique n^2 fois la formule $(ab)_{ij} = \sum_{k=1}^{n} a_{ik}b_{kj}$, qui est une somme de n termes.

L'algorithme de Strassen est basé sur la remarque suivante :

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} P_1 + P_4 - P_5 + P_7 & P_3 + P_5 \\ P_2 + P_4 & P_1 - P_2 + P_3 + P_6 \end{pmatrix},$$

où les P_i sont donnés par

$$\begin{aligned} P_1 &= (A_{11} + A_{22})(B_{11} + B_{22}), \\ P_2 &= (A_{21} + A_{22})B_{11}, \\ P_3 &= A_{11}(B_{12} + B_{22}), \\ P_4 &= A_{22}(B_{21} - B_{11}), \\ P_5 &= (A_{11} + A_{12})B_{22}, \\ P_6 &= (A_{21} - A_{11})(B_{11} + B_{12}), \\ P_7 &= (A_{12} - A_{22})(B_{21} + B_{22}). \end{aligned}$$

On a donc remplacé les huit multiplications de l'algorithme naïf par *sept* multiplications et 18 additions/soustractions. Comme l'addition de deux matrices de taille n se fait en n^2 étapes, il est possible de gagner à effecture le calcul précédent. Le calcul des produits P_i peut alors aussi se faire par l'algorithme de Strassen, jusqu'à manipuler des matrices de "petites" tailles.

Le but est d'implémenter l'algorithme de Strassen, et éventuellement d'illustrer une accélération du calcul pour de grandes matrices. Pour simplifier la partie récursive, on ne considéra que des matrices dont la dimension est une puissance de 2.

Sujet 14 : Recuit simulé en temps continu
Mots-clés : *Équations différentielles stochastiques*

On cherche à minimiser une fonction $V : \mathbb{R}^d \to \mathbb{R}$ que l'on supposera suffisamment régulière, et suffisamment croissante à l'infini.

La mesure invariante du processus

$$dX_t = -\nabla V(X_t)dt + \sqrt{\frac{2}{\beta}}dW_t$$

est la mesure $Z_\beta e^{-\beta V(x)}dx$. En particulier, pour β grand, cette mesure ce concentre autour des minima de la fonction V. Par conséquent, il est naturel d'attendre qu'en remplaçant β par un β_t tendant vers l'infini, le processus converge vers le minimum de V. Il se trouve que pour β tendant suffisamment lentement vers ∞ (par exemple $\beta_t = c\log(t)$ pour c une constante suffisamment grande), on a convergence presque sûre de X_t vers le minimum de V (que l'on suppose unique).

On illustrera cette convergence d'abord sur un exemple en dimension 1 (par exemple $V(x) = \sin(10x) + x^2$), en étudiant l'influence du paramètre c. On pourra ensuite étudier un exemple en dimension supérieure.

Sujet 15 : Un schéma d'Euler qui "ne marche pas"
Mots-clés : *Méthode de Monte Carlo, équation différentielles stochastiques*

Le schéma d'Euler pour la résolution de l'équation

$$dX_t = b(X_t)dt + dW_t$$

consiste en l'approximation $X_t \simeq X_t^\delta$, où le processus X^δ est affine sur chaque segment $[k\delta, (k+1)\delta]$, avec des valeurs aux bords définies par

$$\begin{cases} X^\delta_{(k+1)\delta} & = X^\delta_{k\delta} + b(X^\delta_{k\delta})\delta + (W_{(k+1)\delta} - W_{k\delta}), \\ X^\delta_0 & = X_0. \end{cases}$$

Sous de bonnes hypothèses sur b, on a convergence de X_t^δ vers X_t, par exemple presque sûrement et dans L^p.

Il se trouve que le cas $b(x) = -x^3$ ne rentre pas dans ce cadre d'hypothèses. En effet, dans ce cas X^δ converge presque sûrement uniformément sur les $[0, T]$ vers X, mais on n'a pas de convergence L^1. En fait, $E|X_t^\delta|$ tend vers l'infini quand δ tend vers 0, pour tout $t > 0$. On illustrera (toujours dans ce cas $b(x) = -x^3$) cette convergence presque sûre, ainsi que l'absence de convergence L^1. On pourra aussi donner une approximation à la solution de cette équation par un autre schéma utilisant la solution exacte de l'équation $y' = -y^3$.