Corrigé du partiel de M.A.O. Calcul Formel Durée : 3 heures

Lundi 15 avril 2019
Master 1 M.F., Orsay

1. Pour calculer la i-ème ligne de M on doit calculer les puissances de x_i jusqu'à x_i^{n-1} : chacune est obtenue en multipliant la précédente par x_i. Cela nécessite $n-2$ multiplications pour chaque valeur de i, donc au total $O(n^2)$ opérations. *L'exponentiation rapide n'est d'aucune utilité ici : elle permettrait de calculer rapidement, par exemple, x_i^{n-1}; mais comme on a besoin de calculer aussi toutes les puissances intermédiaires, l'algorithme naïf est optimal.*

2. On a $P(x_k) = \sum_{\ell=0}^{n-1} a_\ell x_k^\ell = \sum_{j=1}^{n} x_k^{j-1} a_{j-1}$ en posant $j = \ell+1$. On a donc $P(x_k) = y_k$ pour tout k si, et seulement si, $Y = MA$.

3. Pour calculer les $P(x_k)$, il suffit de déterminer la matrice Y définie par $Y = MA$. Effectuer naïvement ce produit d'une matrice $n \times n$ par un vecteur colonne coûte $O(n^2)$ opérations (car chacun des n coefficients qu'on calcule est une somme de n produits de deux facteurs). Compte tenu de la question 1, le coût du calcul de Y à partir de A et des x_i est donc de $O(n^2)$ opérations arithmétiques dans $\mathbb{K}$.

4. Cette fois on connaît Y et on cherche A. Comme les x_i sont deux à deux distincts, M est inversible et $A = M^{-1}Y$. Calculer M^{-1} coûte $O(n^3)$ opérations, puis calculer le produit de M^{-1} par Y coûte $O(n^2)$ opérations (comme à la question précédente). Cela fournit donc un algorithme d'interpolation en $O(n^3)$ opérations. *En fait si on utilise les mêmes points x_i pour interpoler de nombreuses fois avec des y_i différents, le calcul de M^{-1} n'est à faire qu'une seule fois : si on le néglige on n'a besoin que de $O(n^2)$ opérations pour chaque donnée des y_i. Mais ce n'est pas le point de vue adopté par l'énoncé.*

5. ($\star$)

```
sage: M = matrix(QQ, 2, 2, [[1, 1], [1, 2]])
sage: Y = matrix(QQ, 2, 1, [[3], [9]])
sage: A = M^(-1) * Y
sage: A[1]
(6)
sage: A
[-3]
[ 6]
```

   Le coefficient de X dans P vaut 6. On peut vérifier le résultat du calcul : $P(X) = 6X - 3$ vérifie bien $P(1) = 3$ et $P(2) = 9$.

6. ($\star$) Il faut être attentif, comme toujours, au fait que les indices des matrices commencent à 0.

```
sage: M = matrix(QQ, 10, 10)
sage: for i in range(10):
....:     for j in range(10):
....:         M[i, j]=(i+1)^j
....:
sage: Y = matrix(QQ, 10, 1)
sage: for j in range(10):
....:     Y[j] = 3^(j+1)
....:
sage: A = M^(-1) * Y
sage: A[1]
(308116/105)
```

Le coefficient de X dans P est $\frac{308116}{105}$.

7. (a) Si il existe un élément ω d'ordre n dans le groupe multiplicatif $\mathbb{F}_q^*$ alors n divise l'ordre de ce groupe, qui est $q-1$, d'où $q \equiv 1 \bmod n$. Réciproquement, si $q \equiv 1 \bmod n$ alors n divise $q-1$. Comme $\mathbb{F}_q^*$ est cyclique, il existe alors un élément d'ordre n dans $\mathbb{F}_q^*$.

   (b) Soient $k, \ell \in \{1, \ldots, n\}$. Le coefficient (k, ℓ) de $M_\omega M_{\omega^{-1}}$ est

$$\sum_{j=1}^{n} \omega^{(k-1)(j-1)} \omega^{-(j-1)(\ell-1)} = \sum_{j=1}^{n} \left(\omega^{k-\ell}\right)^{j-1} = \sum_{t=0}^{n-1} \left(\omega^{k-\ell}\right)^{t}.$$

   Or $\omega^{k-\ell}$ est une racine n-ième de l'unité, puisque ω en est une. Si $\omega^{k-\ell} \neq 1$, cette somme est égale à $\frac{1-\omega^{(k-\ell)n}}{1-\omega^{k-\ell}} = 0$. Dans le cas contraire, elle vaut n car chaque terme vaut 1. Comme ω est une racine primitive n-ième de l'unité, cette deuxième situation se produit si, et seulement si, $k-\ell$ est multiple de n. Comme on a pris $k, \ell \in \{1, \ldots, n\}$ c'est le cas si, et seulement si, $k = \ell$. On a donc démontré que $M_\omega M_{\omega^{-1}} = nI_n$.

   (c) Comme on travaille avec des nombres complexes et qu'on suppose $n \geq 1$, on obtient immédiatement que M_ω est inversible avec $M_\omega^{-1} = \frac{1}{n} M_{\omega^{-1}}$.

   (d) Si la caractéristique de $\mathbb{K}$ est nulle (ou si c'est un nombre premier qui ne divise pas n), l'élément $n = n1_{\mathbb{K}}$ vu dans $\mathbb{K}$ est inversible, et la conclusion est la même que dans la question précédente : M_ω est inversible avec $M_\omega^{-1} = \frac{1}{n} M_{\omega^{-1}}$.

   (e) On applique deux fois la question 2, ainsi que la question (c) ou (d). Etant donné Y, on veut calculer $M_\omega^{-1} Y = \frac{1}{n} M_{\omega^{-1}} Y$. Or calculer $M_{\omega^{-1}} Y$ revient à résoudre le problème de l'évaluation aux points $x'_i = \omega^{-(i-1)}$ du polynôme $y_n X^{n-1} + \ldots + y_2 X + y_1$: par hypothèse on sait le faire efficacement puisque $x'_i = \omega^{n+1-i} = x_{n+2-i}$ pour $2 \leq i \leq n$, et $x'_1 = 1 = x_1$. Il ne reste plus qu'à diviser par n tous les coefficients de $M_{\omega^{-1}} Y$ pour conclure. Ces n opérations sont forcément négligeables par rapport à celles nécessaires pour le reste du calcul, ou à la limite du même ordre.

8. (⋆) Dans le programme ci-dessous la variable `Ltemp` désigne L_k qu'on calcule pour chaque valeur de k. Il faut faire attention à bien écrire `R(3)^(k+1)` plutôt que `R(3^(k+1))` : mieux vaut mettre 3 à la puissance $k+1$ dans $\mathbb{F}_p$ plutôt que de calculer 3^{k+1} (qui est un entier énorme) et de le réduire ensuite modulo p.

```
sage: R = GF(641)
sage: S.<x> = PolynomialRing(R, 'x')
sage: P = 0
sage: n = 100
sage: for k in range(n):
....:     Ltemp = 1
....:     for j in range(k):
....:         Ltemp = Ltemp * (x - R(j + 1))
....:     for j in range(k+1, n):
....:         Ltemp = Ltemp * (x - R(j + 1))
....:     P = P + Ltemp * R(3)^(k + 1) / Ltemp(R(k + 1))
....:
sage: P[2]
130
```

   Le coefficient de X^2 dans P est $\overline{130}$.

9. On effectue la division euclidienne de $L(X)$ par $X - x_k$: le quotient est $L_k(X)$ (et le reste est nul). Le coût de ce calcul est $O(n)$ opérations arithmétiques dans $\mathbb{K}$, puisque $\deg(L)\deg(X - x_k) = n$.

10. Multiplier un polynôme de degré 1 par un polynôme de degré j coûte $O(j)$ opérations arithmétiques dans $\mathbb{K}$.

11. On commence par calculer $L(X)$. Pour cela on calcule $L^{[j]} = \prod_{\ell=1}^{j}(X - x_\ell)$ pour j allant de 1 à n. D'après la question précédente, on calcule $L^{[j]}$ à partir de $L^{[j-1]}$ en $O(j)$ opérations arithmétiques dans $\mathbb{K}$, puisqu'il s'agit simplement de multiplier par $X - x_j$. Comme $L^{[1]}$ est connu, le coût du calcul de $L = L^{[n]}$ est de $O(n^2)$ opérations.

    Fixons maintenant un entier $k \in \{1, \ldots, n\}$. D'après la question 9 on calcule $L_k(X)$ en $O(n)$ opérations. Puis évaluer $L_k(X)$ en x_k nécessite également $O(n)$ opérations : si on ne connaît pas le schéma de Hörner, on peut calculer les puissances de x_k jusqu'à x_k^{n-1}, multiplier chacune par son coefficient, et faire la somme. Enfin on peut multiplier le polynôme $L_k(X)$ par la constante $\frac{y_k}{L_k(x_k)}$ en $O(n)$ opérations. Finalement, on calcule donc $\frac{y_k}{L_k(x_k)}L_k(X)$ en $O(n)$ opérations, pour chaque valeur de k. Quand k varie on obtient donc ces termes en $O(n^2)$ opérations, et on n'a plus qu'à les ajouter (ce qui coûte à nouveau $O(n^2)$ opérations). En conclusion, on a calculé $P(X)$ en $O(n^2)$ opérations.

12. La relation $\zeta^{ij} = \frac{\lambda_{i+j}}{\lambda_i \lambda_j}$ découle immédiatement, en prenant l'exponentielle en base ζ, de la formule suivante qu'on vérifie en développant le membre de gauche :

$$\frac{(i+j)(i+j-1)}{2} - \frac{i(i-1)}{2} - \frac{j(j-1)}{2} = ij.$$

13. Pour tout $i \in \{1, \ldots, n\}$ on a, en utilisant la définition de b_j et la question précédente :

$$P(x_i) = \sum_{j=0}^{n-1} a_j \zeta^{(i-1)j} = \sum_{j=0}^{n-1} \lambda_j b_j \frac{\lambda_{i+j-1}}{\lambda_{i-1}\lambda_j} = \frac{1}{\lambda_{i-1}} \sum_{j=0}^{n-1} b_j \lambda_{i+j-1}.$$

14. L'algorithme de Karatsuba permet de calculer le produit de deux polynômes de degré $\leq t$ à coefficients dans $\mathbb{K}$ en $O(t^\alpha)$ opérations arithmétiques dans $\mathbb{K}$, avec $\alpha = \log_2(3) \approx 1.58$.

15. Pour commencer on calcule les λ_i pour $0 \leq i \leq 2n-2$. La formule $\frac{i(i+1)}{2} = \frac{i(i-1)}{2} + i$ montre que $\lambda_{i+1} = \lambda_i \zeta^i$ pour tout i. On calcule donc simultanément, de proche en proche, les λ_i et les ζ^i (en utilisant pour ces derniers la formule $\zeta^{i+1} = \zeta^i \zeta$). En $O(n)$ opérations on arrive ainsi à calculer tous les λ_i pour $0 \leq i \leq 2n-2$.

    Le calcul de $b_j = a_j/\lambda_j$ pour $0 \leq j \leq n-1$ se fait alors aussi en $O(n)$ opérations. On dispose donc des polynômes $B(X) = \sum_{j=0}^{n-1} b_j X^j$ et $T(X) = \sum_{k=0}^{2n-2} \lambda_{2n-2-k} X^k$. Or on a

$$B(X)T(X) = \sum_{\ell=0}^{3n-3} c_\ell X^\ell$$

    avec

$$c_\ell = \sum_{\substack{0 \leq j \leq n-1 \\ 0 \leq k \leq 2n-2 \\ j+k=\ell}} b_j \lambda_{2n-2-k} = \sum_{\max(0,\ell-2n+2) \leq j \leq \min(n-1,\ell)} b_j \lambda_{2n-2-\ell+j}.$$

    La question 13 donne alors

$$P(x_i) = \frac{c_{2n-1-i}}{\lambda_{i-1}} \text{ pour tout } i \in \{1, \ldots, n\}.$$

    Comme $B(X)$ et $T(X)$ sont de degrés $\leq 2n$, la question 14 montre qu'on peut calculer tous les coefficients c_ℓ du produit BT en $O(n^\alpha)$ opérations arithmétiques dans $\mathbb{K}$. Comme $\alpha > 1$, les n divisions qu'il reste à faire sont négligeables : on calcule les $P(x_i)$ en $O(n^\alpha)$ opérations.

16. Chacun des n^2 coefficients de la matrice RS s'obtient de la manière suivante : on calcule n produits de deux polynômes de degré $< d$, et on effectue $n-1$ additions entre polynômes de degré $< 2d$ pour calculer leur somme. Chaque produit coûte $O(d^\alpha)$ opérations arithmétiques dans $\mathbb{K}$, et chaque somme $O(d)$ opérations. Donc chaque coefficient se calcule en $O(nd^\alpha)$ opérations (puisque $\alpha \geq 1$), et on calcule la matrice RS en $O(n^3 d^\alpha)$ opérations arithmétiques dans $\mathbb{K}$.

17. Pour tout $1 \leq i \leq 2d$, on a $(RS)(x_i) = R(x_i)S(x_i)$, c'est-à-dire que le produit matriciel commute à l'évaluation en x_i de tous les coefficients des matrices mises en jeu. On calcule d'abord les matrices $R(x_i)$ et $S(x_i)$ pour tout $1 \leq i \leq 2d$: chacune possède n^2 coefficients (qui sont des polynômes de degré $< d < 2d$), et il s'agit d'appliquer un algorithme d'évaluation en ces $2d$ points à chacun de ces coefficients. Par hypothèse cela coûte $O(d^\alpha)$ opérations par coefficient, donc $O(n^2 d^\alpha)$ opérations au total. La deuxième étape consiste à calculer pour tout $1 \leq i \leq 2d$ le produit $R(x_i)S(x_i)$. Il s'agit du produit de deux matrices $n \times n$ *à coefficients dans* $\mathbb{K}$; on le fait naïvement en $O(n^3)$ opérations pour chaque valeur de i, donc $O(n^3 d)$ opérations au total. Enfin, la troisième étape consiste à interpoler : chaque coefficient de RS est un polynôme de degré $< 2d$, et on connaît sa valeur en chacun des points x_i, $1 \leq i \leq 2d$ (c'est le coefficient correspondant du produit $R(x_i)S(x_i)$). Par hypothèse, cette étape a le même coût que la première : $O(d^\alpha)$ opérations par coefficient, donc $O(n^2 d^\alpha)$ opérations au total. Finalement, les trois étapes cumulées coûtent $O(n^2 d^\alpha + n^3 d)$ opérations arithmétiques dans $\mathbb{K}$.

18. Une matrice carrée à coefficients dans A est inversible si, et seulement si, son déterminant est un élément inversible de A. Etant donnée la formule du déterminant de la matrice de Vandermonde (qui reste valable sur un anneau), il est donc suffisant de supposer que $x_i - x_j$ est inversible dans A pour tous i et j distincts.

19. Soit $Q \in \mathbb{K}[X]$ de degré d, avec $\mathbb{K} = \mathbb{F}_q$ et d beaucoup plus grand que q. Notons R le reste dans la division euclidienne de $Q(X)$ par $X^q - X$. Comme $x^q - x = 0$ pour tout $x \in \mathbb{F}_q$ on a $Q(x) = R(x)$ pour tout x : il suffit d'évaluer R pour évaluer Q. Comme $\deg R < q$, on sait le faire par hypothèse. Il ne reste donc qu'à calculer R : le cours affirme que cette division euclidienne nécessite $O(dq)$ opérations. En fait R n'a que deux coefficients non nuls donc elle beaucoup plus rapide : $O(d)$ opérations suffisent.

20. Les points x_i correspondent à des nombres premiers p_i, deux à deux distincts. On peut effectuer les calculs sur les entiers modulo chacun des p_i, puis utiliser le théorème chinois pour retrouver le résultat à partir de ses classes de congruence modulo chacun des p_i. L'intérêt est de ne pas devoir manipuler des entiers trop grands. En notant x le résultat du calcul, il faut être sûr a priori que $2|x|$ est strictement inférieur au produit des p_i, pour être sûr que x est le plus petit élément (en valeur absolue) de sa classe de congruence modulo le produit des p_i. Cette condition correspond à l'hypothèse selon laquelle le degré de P est strictement plus petit que le nombre de points x_i. La transformée de Fourier rapide permet d'obtenir des algorithmes très efficaces fondés sur ces idées (voir le dernier chapitre du poly).