Algorithmes pour l'arithmétique II
Cours 1

Julien Lavauzelle

Université Paris 8

Master 2 ACC et CSSD – Algorithmes pour l'arithmétique

24/11/2020

Présentation

Quelques informations personnelles :

- Julien Lavauzelle, maître de conférence à l'Université Paris 8 (depuis cette année)
- email : `julien.lavauzelle@univ-paris8.fr`
- Ma recherche : codes correcteurs, applications en cryptographie
- Enseigne aussi : Théorie de l'information (M1), clé publique (M1), codes correcteurs (M2, à Paris 13)

La page web de ce cours :

`www.math.univ-paris13.fr/~lavauzelle/teaching/2020-21/algorithmes-arithmetiques.html`

Contiendra :

- informations sur le cours
- slides, notes de cours
- exercices

Le passé

Suite d'Algorithmes pour l'arithmétique I (cours de M1)

Ce que vous y avez vu :

- Multiplication rapide (Karatsuba, Toom-Cook)
- Transformée de Fourier discrète
- Division (algorithme binaire, algorithme de Knuth, division rapide, itération de Newton)
- Exponentiation (binaires, chaines d'addition)
- Théorème chinois des restes
- Test de primalité (divisions, cribles, test de Fermat, d'Euler, génération de nombre premiers)

Le futur

Ce que vous allez voir dans ce cours (programme prévisionnel) :

- Factorisation matricielle, suites récurrentes linéaires, résolution de systèmes creux
- Chaînes de Lucas, tests de primalité avancés (Lucas, Frobenius)
- Extraction de racines carrées
- Factorisation de polynômes
- Factorisation d'entiers
- Logarithme discret dans un groupe générique, dans un corps fini

Votre travail (régulier) :

- implémenter et tester les algorithmes
- comprendre les résultats théoriques sous-jacents

Aujourd'hui

1. Factorisations matricielles

Factorisation LU
Factorisation QR
Factorisation de Cholesky

2. Reconstruction de suites récurrentes linéaires

Suites récurrentes linéaires
Algorithme de Berlekamp-Massey

Plan

Plan

Le problème

Résolution d'un système d'équations linéaires.

Notations.

$\mathbb{F}$ un corps.
$\mathbb{F}^n$ espace vectoriel.
$\mathbb{F}^{n\times m}$ espace des matrices de taille $(n \times m)$ sur $\mathbb{F}$.
$\mathrm{rk}(\boldsymbol{a})$ le rang de $\boldsymbol{A} \in \mathbb{F}^{n\times n}$

Problème. Étant donné $\boldsymbol{A} \in \mathbb{F}^{n\times m}$ et $\boldsymbol{b} \in \mathbb{F}^n$, trouver tous les $\boldsymbol{x} \in \mathbb{F}^m$ tels que

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}\,.$$

Théorème. L'espace des solutions est de la forme

$$\boldsymbol{v} + \mathcal{E}$$

où

- $\boldsymbol{v} \in \mathbb{F}^m$ est une solution quelconque de $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$,
- $\mathcal{E} \subseteq \mathbb{F}^m$ est l'espace vectoriel solution de $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}$.

Remarque. On a donc une description courte de l'espace des solutions, avec $n - \mathrm{rk}(\boldsymbol{A}) + 1$ vecteurs de $\mathbb{F}^m$.

Méthode du pivot de Gauss : idée

On se concentre ici sur le cas $n = m$ (matrices carrées).

Idée. on sait bien résoudre quand la matrice $\boldsymbol{A}$ est triangulaire (supérieure).

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ 0 & a_{22} & & & a_{2n} \\ \vdots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 & a_{nn} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ \vdots \\ b_n \end{bmatrix}$$

On calcule itérativement :

$$\begin{aligned} x_n &= \tfrac{1}{a_{nn}} b_n \\ x_{n-1} &= \tfrac{1}{a_{n-1,n-1}} (b_{n-1} - a_{n-1,n} x_n) \\ &\vdots \\ x_1 &= \tfrac{1}{a_{1,1}} (b_1 - \sum_{j=2}^{n} a_{1,j} x_j) \end{aligned}$$

Complexité : $O(n^2)$ opérations dans $\mathbb{F}$.

But : transformer le système original $\boldsymbol{Ax} = \boldsymbol{b}$ en système triangulaire, en sachant relier les solutions des deux systèmes.

Méthode du pivot de Gauss : la théorie (1)

But : transformer A en une matrice triangulaire supérieure.

- Matrices élémentaires
$$M_{ij}\left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right) := \begin{bmatrix} 1 & & & & \\ & a & & b & \\ & & 1 & & \\ & c & & d & \\ & & & & 1 \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}$$
- Action à gauche de $P_{ij} := M_{ij}\left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$ sur A : permutation des colonnes i, j de A
- Action à droite de P_{ij} sur A : permutation des lignes i, j de A
- Action à gauche de $E_{ij}(\alpha) := M_{ij}\left(\begin{smallmatrix} 1 & 0 \\ \alpha & 1 \end{smallmatrix}\right)$ sur A : élimination de coefficients (lorsque α est bien choisi).
  Par exemple
$$E_{2,1}\left(\tfrac{-t}{r}\right) \cdot \begin{bmatrix} r & s \\ t & u \end{bmatrix} = \begin{bmatrix} r & s \\ 0 & * \end{bmatrix}$$

Toutes ces matrices sont inversibles.

Méthode du pivot de Gauss : algorithme

Algorithme de Gauss.

Entrée. Une matrice $\boldsymbol{A} \in \mathbb{F}^{n \times n}$ inversible
Sortie. Trois matrices $\boldsymbol{P}$, $\boldsymbol{Q}$ et $\boldsymbol{A}'$ telles que $\boldsymbol{P}$ est inversible, $\boldsymbol{Q}$ est une matrice d'échange de colonnes et $\boldsymbol{A}'$ est triangulaire supérieure, avec $\boldsymbol{PAQ} = \boldsymbol{A}'$

1. Initialiser $\boldsymbol{P} = \boldsymbol{Q} = \boldsymbol{I}$ et $\boldsymbol{A}' = \boldsymbol{A}$
2. Pour $i = 1 \ldots n$:
   - 2.1 Trouver j, la position d'un pivot inversible dans la i-ème ligne
   - 2.2 Appliquer $\boldsymbol{P}_{ij}$ à droite de $\boldsymbol{Q}$ et à droite de $\boldsymbol{A}'$ (permutation des colonnes de $\boldsymbol{A}$)
   - 2.3 Pour tout $k > i$:
     - 2.3.1 Si $a'_{k,i} \neq 0$, appliquer la matrice $\boldsymbol{E}_{ki}(-\frac{a'_{k,i}}{a'_{ii}})$ à gauche de $\boldsymbol{P}$ et à gauche de $\boldsymbol{A}'$, pour éliminer des coefficients non-nuls de $\boldsymbol{A}$
3. Retourner $\boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{A}'$

Remarque d'implémentation : plutôt que de retourner $\boldsymbol{Q}$, on peut simplement retourner les échanges de colonnes effectués

Complexité en $O(n^3)$ opérations sur $\mathbb{F}$.

Méthode du pivot de Gauss : la théorie (2)

Lemme. On a

$$\boldsymbol{Ax} = \boldsymbol{b} \iff \begin{cases} \boldsymbol{A'y} = \boldsymbol{Pb} \\ \boldsymbol{x} = \boldsymbol{Qy} \end{cases}$$

Par conséquent, pour résoudre le système linéaire $\boldsymbol{Ax} = \boldsymbol{b}$, il suffit

1. d'appliquer l'algorithme de Gauss pour produire $\boldsymbol{A'}$ et $\boldsymbol{P}$ et $\boldsymbol{Q}$
2. de résoudre $\boldsymbol{A'y} = \boldsymbol{Pb}$ (système triangulaire)
3. de calculer $\boldsymbol{x} = \boldsymbol{Qy}$

Remarques.

- On a $\det(\boldsymbol{A}) = \prod_{i=1}^{n} a'_{ii}$.
- Si $\boldsymbol{A}$ n'est pas inversible, l'étape 2.1 de l'algorithme de Gauss peut échouer. Alors, on peut appliquer échanger des lignes de $\boldsymbol{A'}$ en appliquant une matrice $\boldsymbol{P}_{ik}$ à gauche, et ainsi faire remonter une ligne non-nulle de $\boldsymbol{A'}$ en position i.
- On a alors $\mathrm{rk}(\boldsymbol{A}) = \mathrm{rk}(\boldsymbol{A'})$.

Factorisation LU

L'algorithme de Gauss construit donc P et A' tels que $AQ = P^{-1}A'$.

Remarquons que la matrice P ainsi construite est un produit fini de matrices $E_{ij}(\alpha)$.

Exercice. Démontrer que tout produit fini de matrices élémentaires $E_{ij}(\alpha)$ donne une matrice triangulaire inférieure, dont les coefficients diagonaux sont égaux à 1.

Que dire de P^{-1} ?

Exercice. Que vaut l'inverse de $E_{ij}(\alpha)$?

Solution. $E_{ij}(\alpha)^{-1} = E_{ij}(-\alpha)$.

Donc P^{-1} est aussi triangulaire inférieure.

Autrement dit, à permutation de colonnes près, on a factorisé A en produit de deux matrices L, U telles que :

- L est triangulaire inférieure avec des coefficients diagonaux égaux à 1
- U est triangulaire supérieure

Remarques.

1. La factorisation $LU = AQ$ n'est pas unique
2. La factorisation $LU = A$ (i.e., avec $Q = I$) n'est pas toujours possible

Plan

Problème

On se place dans le cas où $\mathbb{F} = \mathbb{R}$.

But : calculer une matrice orthogonale Q et une matrice triangulaire supérieure R telles que

$$A = QR\,.$$

Rappel : Q orthogonale $\iff$ $QQ^T = I$.

Théorème. Si A est inversible, alors il existe une unique factorisation QR telle que la matrice R a ses coefficients diagonaux positifs.

Remarque. Pour $\mathbb{F} = \mathbb{C}$, il existe des résultats similaires. Remplacer les matrices orthogonales par des matrices unitaires, le produit scalaire par le produit hermitien, la transposée par la matrice adjointe.

Motivation.

- Méthode des moindres carrés linéaires (optimisation)
- Réseaux euclidiens (cryptographie)

Procédé d'orthonormalisation de Gram-Schmidt

Définition. Opérateur de projection suivant un vecteur $\boldsymbol{u} \in \mathbb{R}^n$:

$$\pi_{\boldsymbol{u}}(\boldsymbol{v}) = \frac{\langle \boldsymbol{u}, \boldsymbol{v} \rangle}{\|\boldsymbol{u}\|^2} \boldsymbol{u}\,.$$

Algorithme de Gram-Schmidt.

Entrée. n vecteurs $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n$ formant une base de $\mathbb{F}^n$
Sortie. n vecteurs $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ formant une base orthonormale de $\mathbb{F}^n$

1. Pour $i = 1, \ldots, n$,
   1.1 Calculer $\boldsymbol{u}_i = \boldsymbol{v}_i - \sum_{j=1}^{i-1} \pi_{\boldsymbol{u}_j}(\boldsymbol{v}_i)$.
   1.2 Calculer $\boldsymbol{e}_i = \frac{1}{\|\boldsymbol{u}_i\|} \boldsymbol{u}_i$.
2. Retourner $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$

Interprétation géométrique : À l'étape i, on projette $\boldsymbol{v}_i$ orthogonalement sur l'espace $\mathrm{Vect}(\boldsymbol{u}_1, \ldots, \boldsymbol{u}_{i-1})$. Puis on normalise le vecteur de projection.

Méthode de factorisation QR (non optimisée) :

Pour $A \in \mathbb{F}^{n \times n}$ une matrice inversible, on définit $a_1, \ldots, a_n$ les vecteurs colonnes de A.

1. Obtenir une base orthornormée $e_1, \ldots, e_n$ en appliquant le procédé d'orthonormalisation de Gram-Schmidt aux vecteurs $a_1, \ldots, a_n$.
2. Calculer

$$Q = \begin{bmatrix} \vdots & \vdots & & & \vdots \\ e_1 & e_2 & \cdots & \cdots & e_n \\ \vdots & \vdots & & & \vdots \end{bmatrix}$$

et

$$R = \begin{bmatrix} \langle e_1, a_1 \rangle & \langle e_1, a_2 \rangle & \cdots & \cdots & \langle e_1, a_n \rangle \\ 0 & \langle e_2, a_2 \rangle & & & \langle e_2, a_n \rangle \\ \vdots & & \ddots & & \vdots \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & \cdots & 0 & \langle e_n, a_n \rangle \end{bmatrix}$$

Factorisation QR : discussion

Proposition. L'algorithme produit des matrices $\boldsymbol{Q}, \boldsymbol{R}$ telles que $\boldsymbol{QR} = \boldsymbol{A}$.

Preuve. On a généralement

$$\boldsymbol{a}_i = \sum_{j=1}^{n} \langle \boldsymbol{a}_i, \boldsymbol{e}_j \rangle \, \boldsymbol{e}_j \, .$$

Pour observer que $\boldsymbol{R}$ est triangulaire supérieure, on note que le procédé d'orthonormalisation de Gram-Schmidt produit $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n$ tels que $\boldsymbol{e}_j$ est orthogonal à $\boldsymbol{a}_i$ pour tout $j > i$. En effet, $\boldsymbol{a}_i \in \mathrm{Vect}(\boldsymbol{e}_1, \ldots, \boldsymbol{e}_i)$.

Remarques additionnelles.

- L'algoirthme demande le calcul de beaucoup de racines carrées.
- La stabilité numérique de Gram-Schmidt n'est pas très bonne.

Plan

Le problème

On se place dans le cas où $\mathbb{F} = \mathbb{R}$ (s'étend encore à $\mathbb{C}$).

But : étant donnée A symétrique, **définie positive**, calculer une matrice triangulaire inférieure L telle que

$$A = LL^\top .$$

Rappel : A définie positive $\iff$ $x^\top Ax > 0$ pour tout $x \neq 0$ $\iff$ valeurs propres > 0.

Applications. Dans les (nombreux) cas où les systèmes son linéaires :

- optimisation linéaire
- résolution numérique d'équations différentielles
- méthode de Monte Carlo (matrice de covariance)

Algorithme

Si $\boldsymbol{L} = (\ell_{i,j})_{1 \leq i,j \leq n} \in \mathbb{R}^{n \times n}$ triangulaire inférieure vérifie $\boldsymbol{L}\boldsymbol{L}^\top = \boldsymbol{A}$, alors on a

$$\boldsymbol{A} = \begin{bmatrix} \ell_{11}^2 & & & (\text{sym.}) \\ \ell_{21}\ell_{11} & \ell_{21}^2 + \ell_{22}^2 & & \\ \vdots & & \ddots & \\ \ell_{n,1}\ell_{11} & \ell_{n,1}\ell_{2,1} + \ell_{n,2}\ell_{2,2} & \cdots & \sum_{i=1}^{n} \ell_{n,i}^2 \end{bmatrix}$$

On en déduit :

1. calcul de $\ell_{11} = \sqrt{a_{11}}$
2. calcul de $\ell_{21} = a_{21}/\ell_{11}$ puis $\ell_{22} = \sqrt{a_{22} - \ell_{21}^2}$
3. en général, pour $i > j$, calcul de

$$\ell_{ij} = \frac{1}{\ell_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} \ell_{ik}\ell_{jk} \right)$$

puis de

$$\ell_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} \ell_{jk}^2}$$

Discussion

Avantage. Les deux étapes de résolution du système linéaire se font avec les mêmes coefficients $\implies$ gain en complexité : $/2$

Inconvénient. Problème de stabilité dû à l'extraction de racines carrées.

On peut y répondre en calculant une forme alternative

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\top}$$

où $\boldsymbol{D}$ est diagonale (avec éléments diagonaux positifs), et les éléments diagonaux de $\boldsymbol{L}$ valent 1.

Exercice. Décrire un algorithme qui calcule la décomposition LDL.

Plan

Plan

Définitions

Définition. Une suite $\boldsymbol{u} = (u_n)_{n\in\mathbb{N}}$ est récurrente linéaire d'ordre d s'il existe $L \geq d$ et $\boldsymbol{c} = (c_1, \ldots, c_d) \in \mathbb{F}^d$ avec $c_d \neq 0$, tel que

$$\forall n \geq L, \quad u_n + \sum_{i=1}^{d} c_i u_{n-i} = 0\,.$$

Exemple. Considérons la suite définie sur $\mathbb{F}_2$ par

$$u_0 = 1,\ u_0 = 0,\ , u_2 = 1,\ , u_3 = 1,$$

et pour tout $n \geq 4$,

$$u_n = u_{n-1} + u_{n-3}\,.$$

Son ordre est $d = 3$ et les premiers termes de la suite sont :

$$1, \underbrace{0, 1, 1, 1, 0, 1, 0}_{\text{période}}, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1$$

Reformulation en terme de polynômes et de séries formelles.
Pour $\boldsymbol{u} = (u_n)_{n\in\mathbb{N}} \in \mathbb{F}^{\mathbb{N}}$, on note $U \in \mathbb{F}[\![X]\!]$ la série formelle $\sum_{n\geq 0} u_n X^n$.

Fait. La série formelle U d'une suite récurrente linéaire $\boldsymbol{u}$ satisfait

$$UP = Q$$

où $P, Q \in \mathbb{F}[X]$. Un tel polynôme P est appelé *polynôme de connexion*.

Pour $n \geq L$,

$$u_n + \sum_{i=1}^{d} c_i u_{n-i} = 0 \iff \mathrm{Coeff}_n(UP) = 0 \text{ avec } P(X) = 1 + \sum_{i=1}^{d} c_i X^i$$

Les autres coefficients ($n < L$) sont ceux de Q.

Exercice. Démontrer que l'ensemble des polynômes de connexion d'une suite récurrente linéaire forme un idéal de $\mathbb{F}[X]$. En déduire l'existence et l'unicité d'un polynôme de connexion de degré minimal satisfaisant $P(0) = 1$.

Intérêt de la formulation en séries entières

La série formelle d'une suite récurrente linéaire s'écrit donc

$$U = \frac{Q}{P}, \quad \text{où } Q, P \in \mathbb{F}[X]$$

avec $P(0) = 1$.

Réciproquement, on peut montrer que :

Exercice (hors de ce cours). Toute fraction rationnelle $\frac{Q}{P} \in \mathbb{F}(X)$ sans pôle en 0 admet un développement en série formelle $U \in \mathbb{F}[\![X]\!]$.

Exemple typique.

$$\frac{1}{1-X} = \sum_{n \geq 0} X^n$$

Avantage. Permet d'obtenir une description courte de suites périodiques.
Par exemple, $\frac{1}{1-X^6}$ est la série formelle de la suite

$$1\underbrace{00000}_{\text{5 zéros}}100000100000100\ldots$$

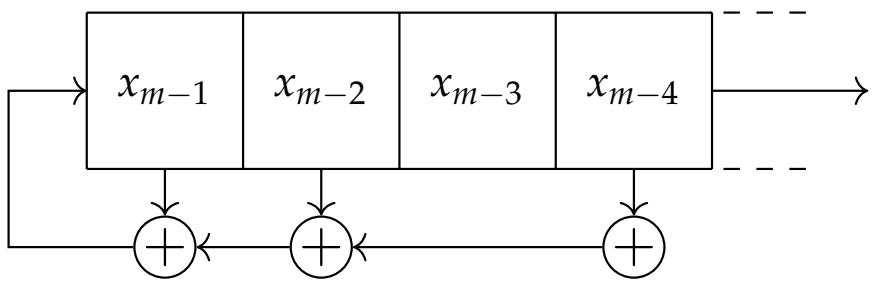
Des suites récurrentes en pratique

Les suites récurrentes linéaires sont fréquemment utilisées en informatique.

Un modèle sous forme de circuit : les **LFSR (*linear feedback shift register*, registre à décalage rebouclé linéairement)**.

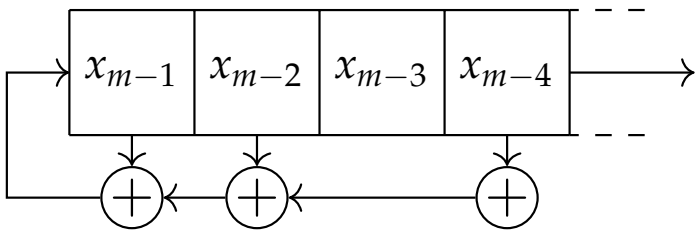
Essentiellement, un LFSR est la donnée d'une suite récurrente linéaire.

Exemple pour $x_m = x_{m-1} + x_{m-2} + x_{m-4}$:



On va également adopter un formalisme polynomial pour définir les LFSR.

Définition. Un LFSR est la donnée d'un polynôme $P \in \mathbb{F}[X]$ et d'un entier $L \geq 1$ (nommé la *dimension*) tel que $\deg(P) \leq L$.



Ici, $P(X) = 1 + X + X^2 + X^4$ et $L = \deg(P) = 4$.

Définition. Soit $L < N$ et $\boldsymbol{u} \in \mathbb{F}^{\mathbb{N}}$ une suite de série formelle $U \in \mathbb{F}[\![X]\!]$. Un LFSR (P, L) engendre $\boldsymbol{u}$ sur N termes si le polynôme

$$PU \bmod X^N$$

est de degré $< L$.

Remarques.

- Un même LFSR (P, L) peut engendrer plusieurs suites finies.
- Le LFSR $(1, L)$ engendre toutes les suites finies de taille L.

Complexité linéaire

Soit $\boldsymbol{u} = (u_n)_{n\in\mathbb{N}}$ une suite de série formelle $U \in \mathbb{F}[\![X]\!]$.
On note $T_k(\boldsymbol{u}) = (u_0, \ldots, u_{k-1})$. Le polynôme associé à $T_k(\boldsymbol{u})$ est $U \mod X^k$.

Définition. La complexité linéaire d'une suite **finie** $\boldsymbol{v} = (v_0, \ldots, v_{n-1}) \in \mathbb{F}^n$ est la plus petite dimension L telle qu'il existe un LFSR (P, L) qui engendre $\boldsymbol{v}$ (sur n termes). On la note $\ell(\boldsymbol{v})$.

Exemple. La suite $\boldsymbol{v} = (1010101010....)$ peut être vue comme

$$v_0 = 1, v_1 = 0 \qquad v_n = v_{n-2} \quad \forall n \geq 2$$

ou comme

$$v_0 = 1, v_1 = 0, v_2 = 1 \qquad v_n = v_{n-1} + v_{n-2} + v_{n-3} \quad \forall n \geq 3$$

Ici, il est clair $\ell(\boldsymbol{v}) = 2$, car on ne peut pas engendrer $\boldsymbol{v}$ avec une suite récurrente d'ordre 1.

Exercice. Quelle est la complexité linéaire de $(\underbrace{0, 0, \ldots, 0}_{k \text{ fois}}, 1)$?

Solution. C'est $k + 1$.

Complexité linéaire

Définition. La complexité linéaire d'une suite finie $\boldsymbol{v} = (v_0, \ldots, v_{n-1})$ est la plus petite dimension L telle qu'il existe un LFSR (P, L) qui engendre $\boldsymbol{v}$ (sur n termes). On la note $\ell(\boldsymbol{v})$.

Définition. Le profil de complexité linéaire de $\boldsymbol{u}$ est la suite $\boldsymbol{\ell} = (\ell_k)_{k \leq N}$ donnée par

$$\ell_k = \ell(T_k(\boldsymbol{u}))\,.$$

Exemple. Pour $\boldsymbol{u} = (1110110110)$ on a

ℓ_0	$= 0$	$(P, L) = (1, 0)$ par convention	ℓ_4	$= 3$	$(P, L) = (1 + X + X^2, 3)$
ℓ_1	$= 1$	$(P, L) = (1, 1)$	ℓ_5	$= 3$	$(P, L) = (1 + X + X^2, 3)$
ℓ_2	$= 1$	$(P, L) = (1 + X, 1)$	ℓ_6	$= 3$	$(P, L) = (1 + X + X^2, 3)$
ℓ_3	$= 1$	$(P, L) = (1 + X, 1)$	ℓ_7	$= 3$	$(P, L) = (1 + X + X^2, 3)$

Exercice. Le profil de complexité linéaire $\boldsymbol{\ell} = (\ell_k)$ d'une suite $\boldsymbol{u}$ est une suite croissante telle que $\ell_0 = 0$.

Problème du LFSR minimal

Problème. Étant donnée une suite $\boldsymbol{u}$ de taille $\leq N$, trouver des LFSR *de dimension minimale* qui engendrent $\boldsymbol{u}$ sur k termes, pour tout $k \leq N$.

Applications.

- LFSR utiles en cryptographie : chiffrement, génération de nombres pseudo-aléatoires
- décodage de codes BCH
- résolution de systèmes linéaires creux (voir cours suivant)

On va voir deux **solutions** à ce problème :

- Aujourd'hui en cours : algorithme de Berlekamp-Massey
- En TD la semaine prochaine : avec l'algorithme d'Euclide étendu (ne donne que le dernier LFSR)

L'algorithme de Berlekamp-Massey est itératif : on construit le LFSR minimal (P_{k+1}, L_{k+1}) à partir de LFSR minimaux obtenus aux ordres antérieurs.

On note

$$\mathcal{R}_k(\boldsymbol{u}) := \{\text{LFSR}(P, L) \text{ qui engendrent } \boldsymbol{u} \text{ sur } k \text{ termes}\}.$$

Exercice. Montrer que $\mathcal{R}_k(\boldsymbol{u}) \subseteq \mathcal{R}_{k-1}(\boldsymbol{u})$.

Lemme. Soient $(P, L), (P, L')$ deux LFSR. Supposons que

- $(P, L), (P, L') \in \mathcal{R}_k(\boldsymbol{u})$ pour une suite $\boldsymbol{u} \in \mathbb{F}^{\mathbb{N}}$,
- $k \geq L + L'$.

Alors il existe $\boldsymbol{v} \in \mathbb{F}^{\mathbb{N}}$ telle que $(P, L), (P, L') \in \mathcal{R}_m(\boldsymbol{v})$ pour tout $m > L, L'$.

Preuve. On écrit $PU = Q + X^k R$ et $P'U = Q' + X^k R'$ avec $\deg(Q) < L$ et $\deg(Q') < L'$. Donc,

$$PP'U = QP' + X^k RP' = Q'P + X^k R'P$$

donne $QP' - Q'P = X^k(R'P - RP')$. En comparant des degrés, $QP' - Q'P = 0$.
On pose alors la suite $\boldsymbol{v}$ définie par la série formelle associée à $\frac{Q}{P} = \frac{Q'}{P'}$. On a alors

$$PV = Q \quad \text{et} \quad P'V = Q'.$$

Premiers résultats

Lemme. Soient $(P, L), (P, L')$ deux LFSR tels que $(P, L), (P, L') \in \mathcal{R}_k(\boldsymbol{u})$ pour une suite $\boldsymbol{u} \in \mathbb{F}^{\mathbb{N}}$ et un entier $k \geq L + L'$.
Alors il existe $\boldsymbol{v} \in \mathbb{F}^{\mathbb{N}}$ telle que $(P, L), (P, L') \in \mathcal{R}_m(\boldsymbol{v})$ pour tout $m > L, L'$.

Corollaire. Si $(P, L) \in \mathcal{R}_k(\boldsymbol{u}) \setminus \mathcal{R}_{k+1}(\boldsymbol{u})$, alors pour tout $(P', L') \in \mathcal{R}_{k+1}(\boldsymbol{u})$ on a $L' \geq k + 1 - L$.

Preuve. C'est la contraposée.

Conséquence : Supposons que l'on connaisse un élément de $\mathcal{R}_k(\boldsymbol{u}) \setminus \mathcal{R}_{k+1}(\boldsymbol{u})$. Si l'on veut chercher un élément (P', L') dans $\mathcal{R}_{k+1}(\boldsymbol{u})$, il faut que $L' \geq k + 1 - L$.

Questions : Comment trouver le polynôme P' ? Peut-on avoir $L' = k + 1 - L$?

C'est l'objet de l'étape d'itération de l'algorithme de Berlekamp-Massey.

Plan

Le lemme central

L'itération de Berlekamp-Massey repose sur le résultat suivant.

Lemme. Soient $(P, L) \in \mathcal{R}_k(\boldsymbol{u}) \setminus \mathcal{R}_{k+1}(\boldsymbol{u})$ et $(P', L') \in \mathcal{R}_{k'}(\boldsymbol{u}) \setminus \mathcal{R}_{k'+1}(\boldsymbol{u})$ où $k' < k$. On note α le coefficient de degré k de UP et α' le coefficient de degré k' de UP'.
Alors, le LFSR donné par

$$\left(P - \frac{\alpha}{\alpha'} X^{k-k'} P' \ , \ \max\{L, L' + k - k'\} \right)$$

est dans $\mathcal{R}_{k+1}(\boldsymbol{u})$.

Preuve. On a

$$UP = Q + \alpha X^k + X^{k+1} R$$
$$UP' = Q' + \alpha' X^{k'} + X^{k'+1} R'$$

Donc

$$U(P - \tfrac{\alpha}{\alpha'} X^{k-k'} P') = \underbrace{Q - \tfrac{\alpha}{\alpha'} X^{k-k'} Q'}_{\deg < \max\{L, L'+k-k'\}} + X^{k+1}(R - \tfrac{\alpha}{\alpha'} R')$$

Lemme. Si $\ell_k(\boldsymbol{u}) < \ell_{k+1}(\boldsymbol{u})$, alors $\ell_{k+1}(\boldsymbol{u}) = k+1-\ell_k(\boldsymbol{u})$.

Preuve. D'après le Corollaire (quelques slides antérieures), on a $\ell_{k+1}(\boldsymbol{u}) \geq k+1-\ell_k(\boldsymbol{u})$.
L'autre sens se montre par récurrence.
Au premier terme non-nul de $\boldsymbol{u}$ on a $\ell_k(\boldsymbol{u}) + \ell_{k+1}(\boldsymbol{u}) = k+1$.
Pour l'induction : soient k', k tels que

$$\ell_{k'}(\boldsymbol{u}) < \ell_{k'+1}(\boldsymbol{u}) = \ell_k(\boldsymbol{u}) < \ell_{k+1}(\boldsymbol{u})$$

Par hypothèse de récurrence $\ell_{k'}(\boldsymbol{u}) = k'+1-\ell_{k'+1}(\boldsymbol{u}) = k'+1-\ell_k(\boldsymbol{u})$.
D'après le lemme précédent, $\ell_{k+1}(\boldsymbol{u}) \leq \max\{\ell_k(\boldsymbol{u}), \ell_{k'}(\boldsymbol{u})+k-k'\} = \ell_{k'}(\boldsymbol{u})+k-k'$. Donc $\ell_{k+1} = k+1-\ell_k(\boldsymbol{u})$.

Théorème. Si $(P_k, L_k = \ell_k(\boldsymbol{u})) \in \mathcal{R}_k(\boldsymbol{u}) \setminus \mathcal{R}_{k+1}(\boldsymbol{u})$, alors

$$\ell_{k+1}(\boldsymbol{u}) = \max\{\ell_k(\boldsymbol{u}), k+1-\ell_k(\boldsymbol{u})\}.$$

Preuve. Reprendre tous les résultats précédents.

Conséquence : si $\ell_k(\boldsymbol{u}) > k/2$, alors $\ell_{k+1}(\boldsymbol{u}) = \ell_k(\boldsymbol{u})$.

L'algorithme de Berlekamp-Massey

Entrée. Une suite finie $\boldsymbol{u}$ commençant par m zéros, donnée par $U \in \mathbb{F}[X]$.
Sortie. Une suite de LFSR $((P_k, L_k))_k$ telle que $(P_k, L_k) \in \mathcal{R}_k(\boldsymbol{u})$.

1. Initialisation :
   - $P_0 = \cdots = P_m = P_{m+1} = 1$.
   - $L_0 = \cdots = L_m = 0$ et $L_{m+1} = m + 1$.
   - $k' = m + 1$ et $\alpha' = u_m$
2. Pour $m + 1 \leq k \leq \deg(U)$
   2.1 Calculer α le terme de degré k de UP_k
   2.2 Si $\alpha = 0$:
      - $P_{k+1} = P_k$
      - $\ell_{k+1} = L_k$

   2.3 Si $\alpha \neq 0$:
      - $P_{k+1} = P_k - \frac{\alpha}{\alpha'} X^{k-k'} P_{k'}$
      - Si $L_k > k/2$, poser $L_{k+1} = L_k$
      - Sinon , poser $L_{k+1} = k + 1 - L_k$ et $k' = k$
      - $\alpha' = \alpha$
3. Retourner la séquence $((P_k, L_k))_k$.

Conclusion

La prochaine fois :

- Algorithmes de Coppersmith/Wiedemann pour la résolution de systèmes linéaires creux
- TD : Euclide étendu pour le problème du LFSR + autres exercices (racines carrées ?)

À faire (optionnel) :

- Essayez d'implémenter Berlekamp-Massey.
- Observez (expérimentalement, théoriquement) la croissance de $\ell_k(\boldsymbol{u})$.

Séance à rattraper : vos disponiblités ?

Documents à (re)trouver sur :

`www.math.univ-paris13.fr/~lavauzelle/teaching/2020-21/algorithmes-arithmetiques.html`