

TP1 : Introduction à SageMath

Présentation du cours & résumé du TP

Bertrand Meyer

24 novembre 2020

Survol du cours

De quoi traite ce cours ?

Cours d'outils de calcul mathématique .

Logiciel :  SageMath

En filigrane :

- La cryptographie du passé (RSA, chiffrement par flot),
- La cryptographie du présent (le chiffrement par courbes elliptiques)
- La cryptographie du futur (compétition du NIST pour la standardisation de cryptographie post quantique)
- Les communications (codes correcteurs d'erreurs)

Mode de travail

Chaque séance porte sur un **sujet distinct**.

Avant : préparer le TP en lisant le sujet et en résolvant certains exercices.

Pendant : faire le point sur la leçon du jour et résoudre le TP
présenter le TP de la séance précédente.

Après : me déposer les exercices à traiter.

La programmation doit rester un **outil** .

Objectif : **apprendre des mathématiques** et non pas à devenir un virtuose de programmation.

La **note finale**  est formée de

- Une note de **contrôle continu** (12 points).
Exercices à rendre pour chaque TP.
- Une note d'**examen final** (8 points).
Examen traditionnel sans ordinateur.

Utiliser l'enseignant comme **ressource** pour apprendre et non pas comme examinateur.

Contenu

Deux parties

Le cours se divise en

Partie A Algorithmes pour l'algèbre

Partie B Algorithmes pour l'arithmétique

Partie A (période P2) : résoudre des équations

TP1 : Initiation à SageMath

TP2 : Algèbre linéaire sur un anneau

TP3 : Réseau euclidien, algorithme LLL
(prérequis du TP 12)

TP4 : Factorisation sur les corps finis

TP5 : Factorisation en général

TP6 : Systèmes polynomiaux (applications :
cryptographie post-quantique, robotique)

TP7 : Systèmes polynomiaux (préparation à
ACCQ205)






Partie B (période P4) : théorie des nombres algorithmique

- TP8 : Nombres premiers
- TP9 : Factorisation (application : cryptographie traditionnelle)
- TP10 : Production de bits (application : cryptographie par flots)
- TP11 : Code correcteurs (complète ACCQ204, applications : communications, cryptographie post-quantique)
- TP12 : Cryptographie des réseaux (applications : cryptographie post-quantique, chiffrement homomorphe)
- TP13 : Courbe elliptique (complète ACCQ207, application : cryptographie ECC)
- TP14 : Logarithme discret & couplage (complète ACCQ207, application : protocoles Diffie-Hellman)

Introduction à SageMath

Quoi Logiciel de **calcul mathématique** gratuit et ouvert.

Comment S'appuie sur des **logiciels préexistants**,
Utilise  python™ comme langage de programmation.

Trois faces Distribution  de logiciels,
Interface  vers ceux-ci,
Bibliothèque de méthodes mathématiques .

- Lancer SageMath depuis un terminal `sage` ou `sage -n jupyter`.
- Quand un objet est défini : la touche tab permet d'afficher la liste des méthodes.
- Aide d'une méthode : `objet.methode?` ou `objet.methode??`.
- Le langage est riche : `exécuter` le code `souvent` plutôt que de déboguer.
- Les bibliothèques Python sont inutiles : oubliez `numpy`, `scipy`, `random`. Utilisez les structures de SageMath.

Corrigé de l'exercice 1

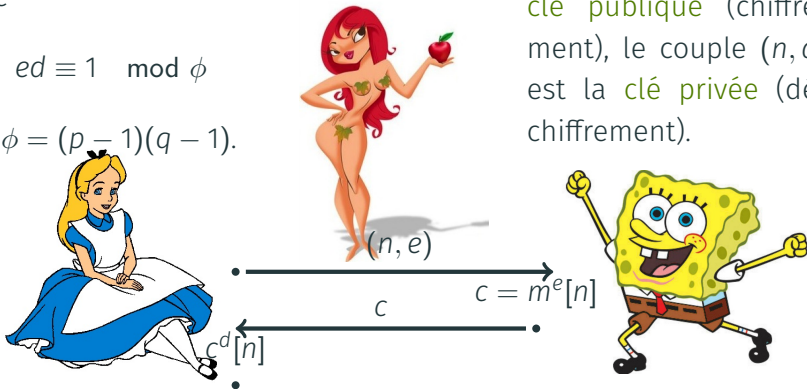
Le cryptosystème de Rivest, Shamir & Adleman (RSA)

Alice génère p , q premiers, puis e et d tels que

$$ed \equiv 1 \pmod{\phi}$$

où $\phi = (p-1)(q-1)$.

Alice pose $n = pq$. Le couple (n, e) est la **clé publique** (chiffrement), le couple (n, d) est la **clé privée** (déchiffrement).



On a bien $c^d = (m^e)^d = m^{ed} = m[n]$ car $\mathbb{Z}/n\mathbb{Z}^\times$ est cyclique d'ordre ϕ .