Cryptographie
&
Calcul symbolique

– introduction –

Sandrine Julia, Bruno Martin

Université Côte d'Azur

Option – Licence 3 Info./ Math.-Info./ Sc. & Techn. – Automne 2023

Organisation

1h30 de cours, 2h TP hebdomadaires sur 12 semaines, 6 TD de 2h

https://lms.univ-cotedazur.fr/2023/course/view.php?id=13766

https://upinfo.univ-cotedazur.fr/~julia/Crypto

Selon les semaines, soit :

Cours	vendredi	10h15-11h45	amphi biologie
TP	vendredi	13h15-15h15	salles P.V. 314 , P.V. 315

soit :

Cours	vendredi	10h15-11h45	amphi biologie
TP	vendredi	13h15-15h15	salles P.V. 314 , P.V. 315
TD	vendredi	15h30-17h30	salle M.1.6

Evaluation :

partiel (30%) + travail et tests en TP (20%) + examen (50%)

Plan du cours

https://lms.univ-cotedazur.fr/2023/course/view.php?id=13766

https://upinfo.univ-cotedazur.fr/~julia/Crypto

0 - Introduction au calcul symbolique
1 - Introduction à la théorie de l'information
2 - Codes de source - codes d'Huffman
3 - Compression de textes
4 - Codes de canal - codes de Hamming
5 - Cryptographie à clef secrète
6 - Cryptographie à clef publique I
7 - Cryptographie à clef publique II
8 - Fonctions de hachage, intégrité
9 - Signatures digitales
10 - Certificats, échange de clefs, protocoles sécurisés
11 - La cryptographie (d'aujourd'hui et) de demain

(Les TP utilisent `Sympy` et la librairie `OpenSSL` sous Jupyter)

Bibliographie

- Handbook of Applied Cryptography, A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, CRC Press, 1996
  (*en ligne :* https://cacr.uwaterloo.ca/hac/)
- La science du secret, J. Stern, Odile Jacob, 1998
- Histoire des codes secrets, S. Singh, LFG, 2001
- Codage, cryptologie et applications, B. Martin, PPUR, 2004
- A Classical Introduction to Cryptography, S. Vaudenay, Springer, 2005
- Théorie des codes, J.-G. Dumas, J.-L. Roch, E. Tannier, S. Varette, Dunod, 2007
- La fracture cryptographique, S. Vaudenay, PPUR, 2010
- Exercices et problèmes de cryptographie, D. Vergnaud, Dunod, 2018.

Introduction au calcul symbolique

Calcul symbolique (ou formel)

À l'intersection des mathématiques et de l'informatique.

Étudie et propose des algorithmes qui travaillent de manière symbolique sur des expressions mathématiques ayant une représentation finie et exacte : *Computer Algebra System*.

Diffère du calcul scientifique (ou numérique) qui travaille sur des nombres ayant une représentation numérique approchée (en virgule flottante).

Exemple (de calcul symbolique)

calculer la dérivée, la primitive d'une fonction, simplifier une expression algébrique, faire tous les calculs algébriques habituels (matriciel,...)

Bref historique

1950 algorithmes de calcul de dérivée d'une fonction

1970 premiers systèmes de calcul formel : `Reduce` et `Macsyma`, écrits en `LISP`

1980 systèmes modernes (avec GUI) : `Maple` et `Mathematica`, écrits en `C`

2000 calcul formel dans le libre : `Sage` qui utilise `Python` (et d'autres langages) ou `SymPy` un module de `Python`

Objets du calcul formel

- les nombres
  - les entiers (en précision arbitraire, $100!$)
  - les rationnels par un couple p/q de deux entiers
  - les entiers modulo p : un élément de l'ensemble $\{0,\ldots, p-1\}$
- les polynômes
- les matrices
- et bien d'autres objets

Que fait un système de calcul formel ?

Résolution d'équations, factorisation, simplification ou réécriture d'expressions

Calcul symbolique dans des structures algébriques
groupes, anneaux, corps,...

Plus généralement, travailler sur des expressions de façon symbolique (par opposition à numérique)

SymPy comme système de calcul symbolique

Deux logiciels libres récents de calcul symbolique :

- `SageMath` rassemble un certain nombre de systèmes de calcul symbolique au sein d'une interface standardisée avec un gros noyau
- `SymPy`, un module `Python`. Il est léger et fonctionne sur tout système capable d'exécuter `Python`. Il utilise
  - `NumPy` pour le calcul numérique
  - `Matplotlib` pour le graphisme.

  On privilégiera l'utilisation des feuilles de calcul fournies par `Jupyter`.

Structure de SymPy

- interface web graphique : interagit avec l'utilisateur ; gère le fichier de travail (feuille de calcul ou notebook `Jupyter`), permet de saisir les instructions et d'afficher les résultats, y compris l'affichage de graphiques
- noyau (kernel) : interprète les instructions écrites en `SymPy` ou en `Python`, effectue les calculs et retourne le résultat

Pour lancer la session (et un navigateur), saisir la commande :

```
$ jupyter notebook
$ /opt/anaconda3/bin/jupyter-notebook (au PV)
```

dans un terminal (sous `linux`, `*BSD`, `MacOS`) ou dans une invite de commande sous `Windows`.

Structure d'un notebook

Fichier `SymPy`
Ou notebook, d'extension `.ipynb` est strucuré en cellules (cells) d'une ou plusieurs lignes.

Chaque cellule peut contenir des instructions, des résultats, du texte ou du texte mis en forme en `markdown`.

Une cellule est évaluée par MAJ+ENTRÉE

```
import sympy as symb
symb.init_printing()   # sortie formatée pour l'interface
x = symb.symbols('x')
r = sqrt(x)
r
```

$$\sqrt{x}$$

UTILISER L'AIDE

Le menu `Help` propose des tutoriels et de l'aide. Il donne accès aux pages des différents modules (`NumPy, SciPy, SymPy,...`)

On peut demander de l'aide sur une fonction spécifique par

```
?factorial
```

A suivre ...