Mathématiques avec Sage

Vincent Delecroix

Mars 2018

Introduction

Cette courte note a pour but de vous familiariser avec le logiciel de calcul formel Sage (ou SageMath). Il contient un bref historique, des notes pour son installation et une liste de ressources. Ce texte est extrait d'un texte plus long du même auteur écrit pour l'École Jeunes Chercheurs en Informatique édition 2018, voir `https://ejcim2018.sciencesconf.org`.

Sage est un logiciel qui évolue rapidement depuis sa création en 2005. Les informations contenues dans ce document concernent la version Sage 8.0 datant de juillet 2017.

1 Histoire et idéologie

Cette courte section donne quelques points de repère sur l'écosystème du logiciel Sage. Il ne vous apprendra pas à utiliser Sage et vous pouvez sauter à la section suivante si c'est cela que vous cherchez.

Le début de l'histoire. Sage est un logiciel de mathématiques qui a été créé autour de 2005 par William Stein. Sa motivation originale était de faire interagir des bibliothèques et logiciels pour faire des calculs autour des formes modulaires (voir par exemple le livre [3] datant de 2006 et contenant les premiers exemples de code Sage). La première version de Sage (qui s'appelait alors Manin) faisait interagir PARI/GP (bibliothèque et logiciel de théorie des nombres), Pyrex (compilateur permettant d'interfacer simplement du code C en Python), GMP (pour les entiers en précision arbitraire) et NTL (bibliothèque pour la théorie des nombres). Sage, ou plutôt Manin, offrait aussi une interface à Mathematica et Magma, deux logiciels propriétaires.

Peu après, sous l'impulsion de David Joyner et David Kohel qui rejoignirent l'équipe de développement, une interface à GAP (pour pouvoir travailler avec les groupes de permutations), Maxima (pour le calcul symbolique) et Singular (pour les polynômes multivariés) sont crées. La structure interne de Manin évolue considérablement. Le nom change de Manin à SAGE comme acronyme à "Software for Algebra and Geometry Experimentation". Aujourd'hui, le logiciel s'appelle Sage ou SageMath et l'acronyme a été laissé de côté.

Pour plus de détails bibliographiques sur les débuts, vous pouvez consulter le document [4].

Aujourd'hui. Sage est actuellement developpé par des centaines de personnes et permet l'interaction d'environ 200 bibliothèques et logiciels. Il a un cycle de développement rapide avec entre 4 et 5 versions par an ces dernières années, voir Figure 1. Ce choix de cycle rapide permet aux utilisateurs de profiter des améliorations rapidement et simplement.

L'écosystème Sage. Comme il a été mentionné dans les deux paragraphes précédents, la partie principale de Sage est une interface commune à des bibliothèques spécialisées. Il contient aussi du code propre essentiellement écrit en Python (1.5 millions de lignes) et Cython (0.5 million de lignes).

année	mois	version
2013	déc.	**6.0**
2014	jan.	6.1
	mai	6.2
	août	6.3
	nov.	6.4
2015	fév.	6.5
	avr.	6.6
	mai	6.7
	juil.	6.8
	oct.	6.9
2016	jan.	**7.0**
	mars	7.1
	mai	7.2
	août	7.3
	oct.	7.4
2017	jan.	7.5
	mars	7.6
	juil.	**8.0**

FIGURE 1 – Dates de sorties des versions de Sage depuis la version 6.0.

Notons que le projet Pyrex faisant partie de Sage à ses débuts a été repris par certains développeurs de Sage pour donner naissance à Cython (un des mainteneurs est R. Bradshaw, employé de Google et ancien développeur de Sage). Ce dernier est largement utilisé en dehors de Sage. Les développeurs sont ainsi encouragés à participer au développement d'outils dont la portée dépasse simplement l'usage interne de Sage.

Licence. Dès sa création, un des objectifs du projet SageMath est de proposer un logiciel de mathématiques générales sous licence libre (et donc gratuit). Des logiciels propriétaires (souvent chers) sont surreprésentés dans la recherche ainsi que les cursus universitaires. Sage est publié sous licence GPL [1].

Python. Une autre particularité de Sage est d'avoir adopté le langage de programmation Python plutôt que d'en créer un nouveau (de même que Maxima est basé sur Lisp, mais contrairement à Mathematica ou PARI/GP qui nécessitent d'apprendre leur propre langage). Le langage Python est très facile à prendre en main même pour quelqu'un qui n'a jamais programmé. Cette simplicité du langage est probablement pour beaucoup dans la popularité de Sage. Par exemple, le code suivant devrait être lisible par la plupart des scientifiques

```
sage: V = ZZ^3
sage: v1 = V([1, 3, 5])
sage: v2 = V([-1, 1, 0])
sage: v1.dot_product(v2).is_prime()
True
```

Bien que pour utiliser Sage il vous faut écrire en Python, les bibliothèques utilisées en internes sont écrites en C/C++, Lisp ou Fortran.

Il est à noter que Python n'est pas un langage performant (attendez vous à des boucles jusqu'à 60 fois plus lentes que dans un langage compilé [2]). Cependant ces écueils de Python sont faciles à identifier et, lorsque c'est nécessaire, il est facile d'insérer du code en C/C++ en utilisant Cython.

Développement. Le développement de Sage est assez unique car il regroupe plusieurs centaines de développeurs sur un pied d'égalité. Chacun est libre de proposer une modification du code via un ticket sur le site `https://trac.sagemath.org`. Le ticket passe ensuite par un relecteur qui est un des autres développeurs. Suite à des discussions entre l'auteur et le relecteur des modifications peuvent être apportées au ticket. Lorsqu'un consensus est atteint le code est alors inclus dans Sage et sera disponible dans la prochaine version.

1. voir par exemple `https://fr.wikipedia.org/wiki/Licence_publique_générale_GNU`.
2. Les comparatifs ne manquent pas, voir par exemple `http://www.osnews.com/story/5602/Nine_Language_Performance_Round-up_Benchmarking_Math_File_I_O`.

Il est relativement aisé de contribuer à Sage et toutes les utilisatrices y sont invitées.

Brève liste de logiciels et bibliothèques. Nous donnons une petite liste des bibliothèques et logiciels utilisés par Sage et qui sont cités dans ce document. Vous trouverez une liste exhaustive sur le site `http://www.sagemath.org/`. Nous choisissons de présenter seulement les 4 logiciels de calcul formels à partir desquels Sage a été construit à ces débuts (GAP, Maxima, PARI/GP et Singular) ainsi que les librairies prenant une importance plus grande dans le cadre de ce texte.

Cryptominisat (`https://github.com/msoos/cryptominisat`)
Bibliothèque C et module Python pour résoudre les problèmes SAT. Il s'agit d'un paquetage optionnel de Sage.

Cython (`http://cython.org/`)
Compilateur d'extension C/C++ pour Python (successeur de Pyrex).

GAP (`http://www.gap-system.org/`)
Logiciel de calcul formel avec un très fort développement vers la théorie des groupes. Les premières versions datent de 1988 [3].

GMP/MPIR (`https://gmplib.org/` et `http://mpir.org/`)
Librairie C pour les entiers en précision arbitraire (avec de nombreuses optimisations). La fourche MPIR de GMP a été crée pour offrir des optimisations pour plus d'architectures et un support Windows.

Maxima (`http://maxima.sourceforge.net/`)
Logiciel en licence libre pour la manipulation d'expressions symboliques. Le projet découle de Macsyma qui a commencé dans les années 1960 !

NTL (`http://www.shoup.net/ntl/`)
Librairie C++ pour la théorie des nombres.

Python (`https://www.python.org/`)
Langage de programmation sur lequel Sage se base.

Pyrex
Compilateur anciennement utilisé par Sage pour créer des extensions Python en C.

PARI/GP (`https://pari.math.u-bordeaux.fr/`)
PARI est une bibliothèque C de théorie des nombres et GP un interpréteur qui permet de l'utiliser. Il est distribué sous licence libre et ses débuts datent de 1983 [4].

Parma Polyhedra Library (PPL) (`http://bugseng.com/products/ppl/`)
Bibliothèque pour les problèmes linéaires en nombres rationnels (polytopes, optimisation, etc).

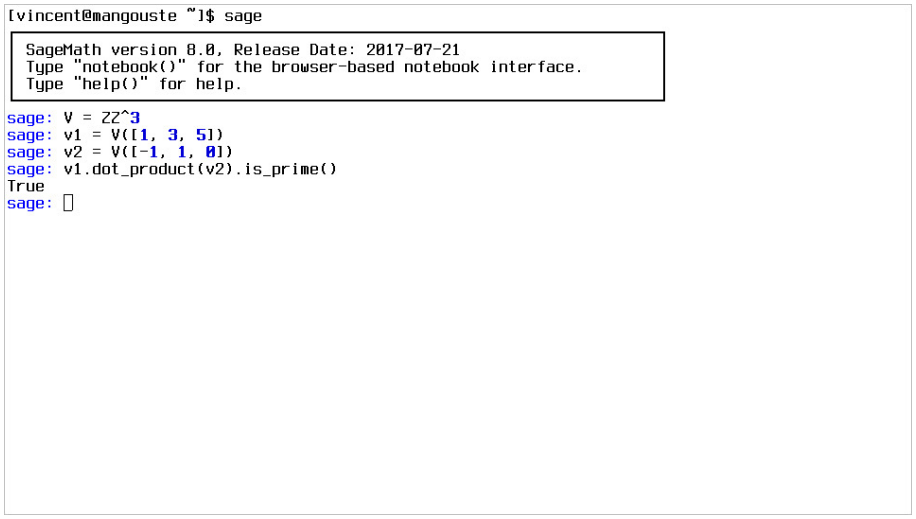
Singular (`http://www.singular.uni-kl.de/`)
Bibliothèque et logiciel de calcul spécialisé dans les systèmes polynomiaux et les bases de Gröbner. Le projet a commencé en 1984 [5].

2 Installer et démarrer Sage

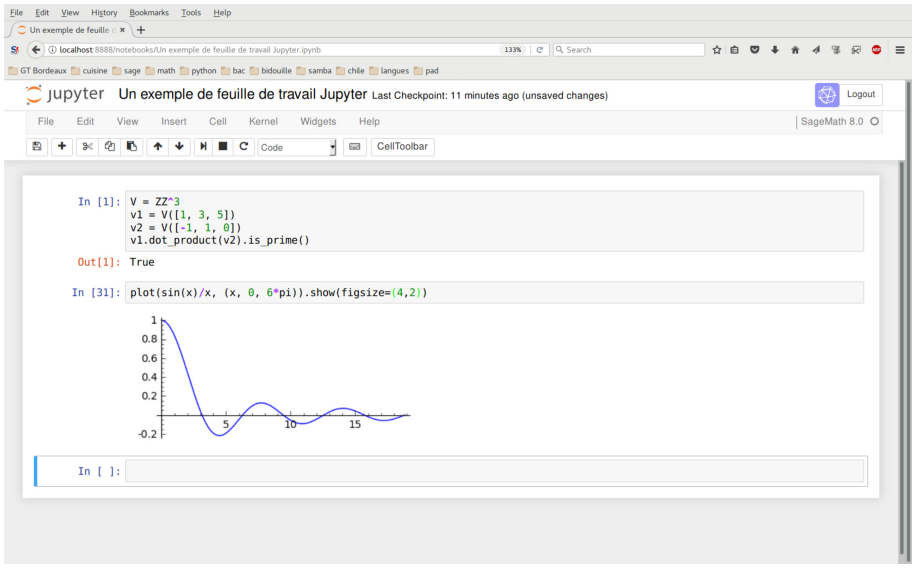
Sage peut s'utiliser d'au moins cinq façons

3. Voir `http://www.gap-system.org/Doc/History/history.html`.
4. Voir `http://pari.math.u-bordeaux.fr/timeline.html`
5. Voir `http://www.singular.uni-kl.de/index.php/background/history.html` .

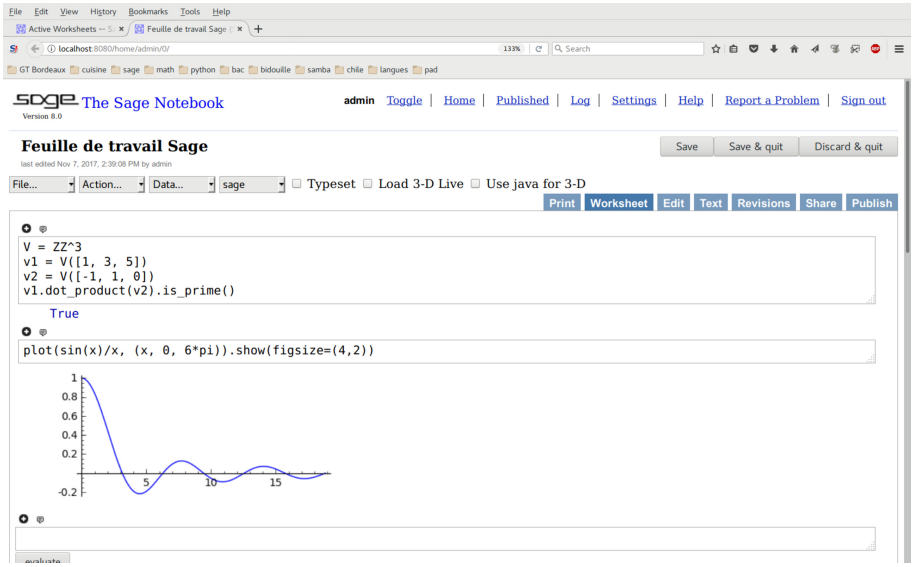
1. sur son propre ordinateur via la console (la commande pour lancer l'exécutable est appelée `sage`),



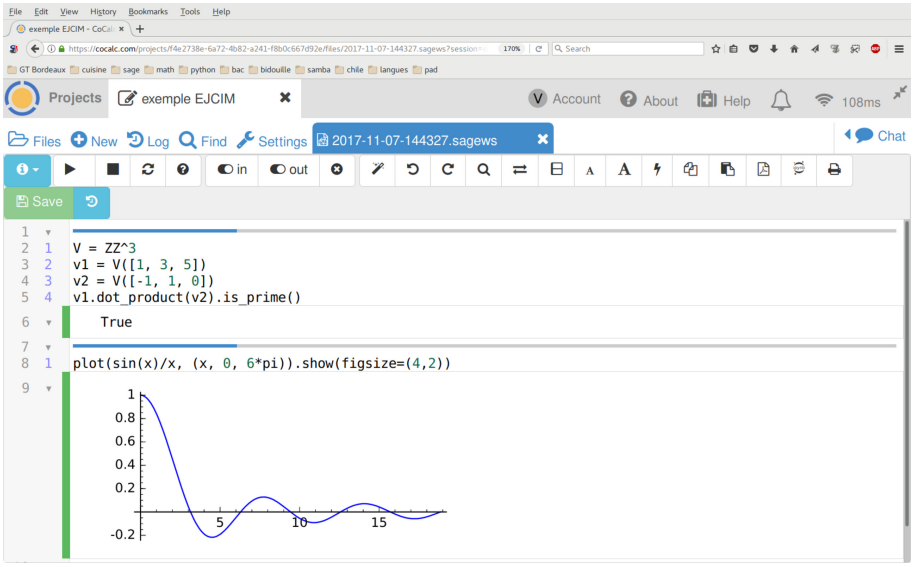
2. sur son propre ordinateur via Jupyter (interface graphique par défaut à partir de Sage 8.0) – la commande pour lancer Jupyter est `sage -n jupyter`,



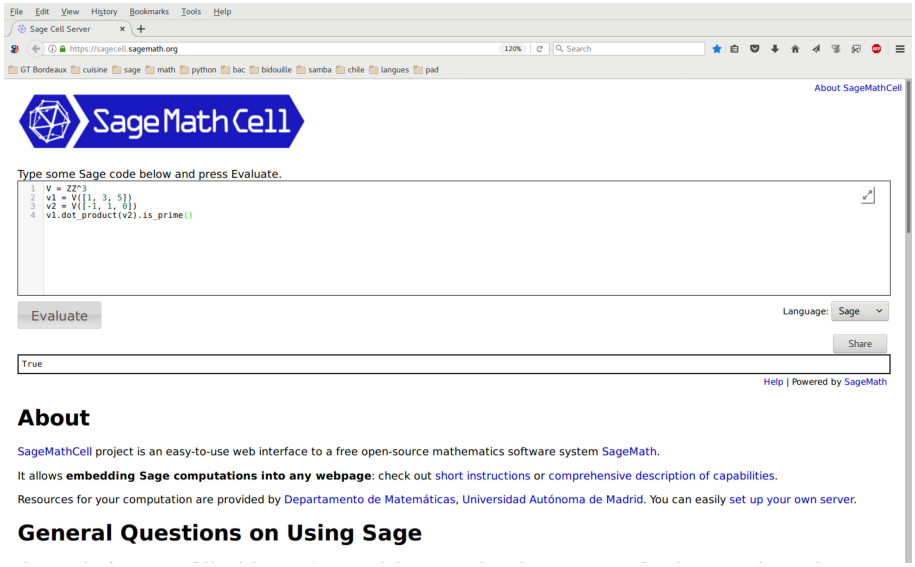
3. sur son propre ordinateur via le Sage Notebook (interface graphique par défaut jusqu'à Sage 7.6 et qui va progressivement disparaître) – la commande pour lancer le Sage Notebook est `sage -n sagenb`,



4. en ligne sur Cocalc (`https://cocalc.com/`),



5. en ligne via une cellule Sage `https://sagecell.sagemath.org/`).



La manière d'installer Sage sur votre machine dépend de votre système.

1. Debian ($\geq$ Stretch, 9.2) et Ubuntu ($\geq$ 17.04, Zesty) : installez le paquetage `sagemath`.
2. Fedora : il y a un support limité pour un paquetage `sagemath` non officiel (voir `http://fedoraproject.org/wiki/SIGs/SciTech/SAGE` et `https://ask.sagemath.org/question/27158`).
3. Archlinux : installez le paquetage `sagemath`, voir `https://wiki.archlinux.org/index.php/SageMath`.
4. Gentoo : consultez `https://github.com/cschwan/sage-on-gentoo/`.
5. Windows : utilisez `https://github.com/embray/sage-windows/`.
6. Pour tous les autres systèmes, téléchargez le binaire correspondant. sur `http://www.sagemath.org/download`

Formats de fichiers, sauvegardes. Pour sauvegarder votre travail de programmation vous avez plusieurs possibilités.

extension	explication
`.py`	fichiers Python, pour lire un fichier Python depuis la console ou dans Jupyter vous pouvez utiliser `%runfile` ou `%attach`
`.pyx`	fichier source Cython, fonctionne également avec `%runfile` et `%attach`
`.sage`	fichiers Sage. Équivalent à un fichier Python sauf lorsque vous exécuterez votre fichier dans un shell via `$ sage mon_fichier.sage` auquel cas le préparseur de Sage est appliqué
`.ipynb`	feuille de travail ("notebook") Jupyter (le nom ipynb découle de "IPython notebook")
`.sws`	feuille de travail Sage (obsolète)
`.rst`	fichier au format ReST, permet d'écrire des fichiers mélangeant du texte et du code Sage. Conversion possible vers pdf ou feuille de travail Jupyter.

3 Cinq principes de base

Casse. Il y a deux conventions de nommage dans Sage
— notation serpent ("snake case") : `vector`, `matrix`, `is_prime`, ...
— notation chameau ("camel case") : `VectorSpace`, `PolynomialRing`, `Matrix`, ...
La notation chameau est réservée aux noms des objets (les espaces vectoriels, les anneaux de polynômes, les matrices, ...). La notation serpent est a priori réservé aux fonctions (comme `is_prime`) mais il y a parfois des fonctions qui construisent des objets (comme `vector` et `matrix`).

Orientation objet. Python est un langage objet. La plupart des fonctions qui s'appliquent à un objet `x` sont appelées via `x.f()` plutôt que `f(x)`.

```
sage: 42.bits()
[0, 1, 0, 1, 0, 1]
sage: bits(42)
Traceback (most recent call last)
...
NameError: name 'bits' is not defined
```

Parfois les deux syntaxes sont possibles

```
sage: 101.is_prime()
True
sage: is_prime(101)
True
```

Complétion automatique. Pour trouver une fonction, une manière pratique consiste à utiliser la complétion automatique par la touche tabulation. Cela consiste à écrire une partie d'un mot et appuyer sur la touche ⟨tab⟩.

```
sage: prim<tab>
           prime_divisors  prime_powers    primes
           prime_factors   prime_range     primes_first_n
           prime_pi        prime_to_m_part primitive_root
```

Lorsqu'une seule complétion est possible le mot est automatiquement complété. Cette remarque s'applique aussi aux méthodes d'un objet préalablement construit

```
sage: n = 42
sage: n.f<tab>
          n.factor
          n.factorial
          n.floor
```

Documentation. Il est possible d'accéder à la documentation directement dans la console ou une feuille de travail. Pour cela, on ajoute un point d'interrogation à la fin avant d'appuyer sur

```
sage: bernoulli?
Signature:      bernoulli(n, algorithm='default', num_threads=1)
Docstring:
   Return the n-th Bernoulli number, as a rational number.

   INPUT:
```

```
* "n" - an integer

* "algorithm":

  * "'default'" -- use 'flint' for n <= 300000, and 'bernmm'
    otherwise (this is just a heuristic, and not guaranteed
    to be optimal on all hardware)

  * "'arb'" -- use the arb library

  * "'flint'" -- use the FLINT library
```

Pour accéder au code source, on utilise deux points d'interrogation au lieu d'un.

Langage Python. Pour aller plus loin en programmation dans Sage il est indispensable de bien connaître la syntaxe et les structure de données Python. Python est avant tout un langage impératif et on retrouve les flots de contrôle `if`/`elif`/`else`, `for` et `while`. Par ailleurs, Python possède quelques structures de données avancées dont voici la liste.

nom	type	délimiteurs	modifiable
liste	`list`	`[` et `]`	oui
tuple	`tuple`	`(` et `)`	
chaîne	`str`	', ", ''', """	non
ensemble	`set`	`{` et `}`	oui
ensemble gelé	`frozenset`		non
dictionnaire	`dict`	`{` et `}`	oui

Pour apprendre Python, je recommande
- le tutoriel en français de la documentation officielle Python `https://docs.python.org/fr/2/tutorial/`,
- les premiers chapitres des "Scipy lecture notes" (en anglais) `http://www.scipy-lectures.org/`
- le tutoriel thématique "Tutorial : Programming in Python and Sage" de la documentation officielle de Sage (en anglais) `http://doc.sagemath.org/html/en/thematic_tutorials/tutorial-programming-python.html`.
- le tutoriel "How to Think Like a Computer Scientist" (en anglais) `http://interactivepython.org/runestone/static/thinkcspy/index.html`

4 Ressources

Livres.
- Calcul Mathématique avec Sage [2] : ouvrage de référence en français qui couvre les différents aspects du calcul formel et scientifique. Couvre grosso-modo les mathématiques rencontrées dans un cursus de licence. Mentionnons également qu'une version en anglais pour Sage 8.0 est disponible. Toutes les versions et correctifs sont disponibles à `https://members.loria.fr/PZimmermann/sagebook/`.
- Sage for Undergraduates [1] : ouvrage en anglais qui couvre le programme de licence de mathématiques avec beaucoup de détails concernant le graphisme.

Documentation en ligne. Voir la fin de la Section précédente concernant le langage Python.

Autre ressources en ligne.

— Un forum de questions/réponses `http://ask.sagemath.org/`.

— Une liste de discussion `https://groups.google.com/forum/#!forum/sage-support`.

References

[1] G. V. Bard. *Sage for Undergraduates*. American Mathematical Society, Providence, RI, 2015.

[2] A. Casamayou. *Calcul mathématique avec Sage*. 2013.

[3] W. Stein. *Modular forms, a computational approach*, volume 79 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2007. With an appendix by Paul E. Gunnells.

[4] W. Stein. Mathematical software and me: A very personal recollection, 2009. disponible à `http://wstein.org/mathsoftbio/history.pdf`.