

Python pour les SHS

Introduction à la programmation scientifique

Émilien Schultz

emilien.schultz@gmail.com

SESSTIM

Ce que j'aimerai faire aujourd'hui

- ▶ Montrer l'intérêt du langage Python pour les SHS
- ▶ Vous mettre le pied à l'étrier avec les bases du langage
- ▶ Permettre (éventuellement) une autonomie dans l'apprentissage
- ▶ Éclairer certains aspects plus avancés.
- ▶ Échanger !

Ce que je ne ferai pas

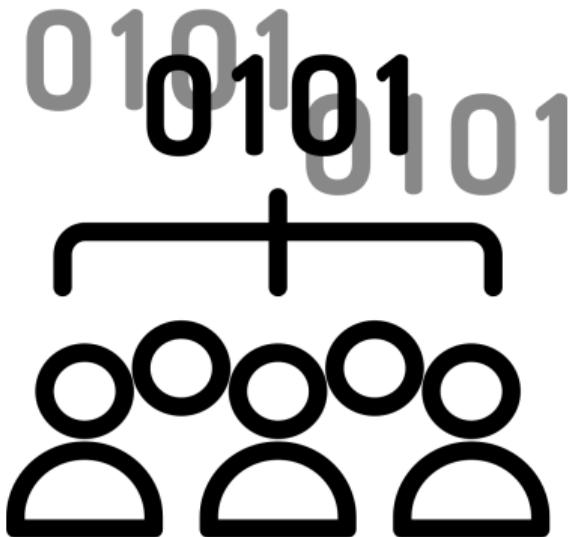
- ▶ De la théorie sur la programmation
- ▶ Épuiser toutes les notions du langage de programmation Python
- ▶ Faire de vous des développeurs.ses

Le dépôt Github

<https://github.com/pyshs/Formation-URFIST-Lyon-2023-automne>

Répondre à 3 questions

1. Pourquoi programmer (en recherche) ?
2. Pourquoi Python ?
3. Pourquoi penser les usages spécifiques aux SHS ?



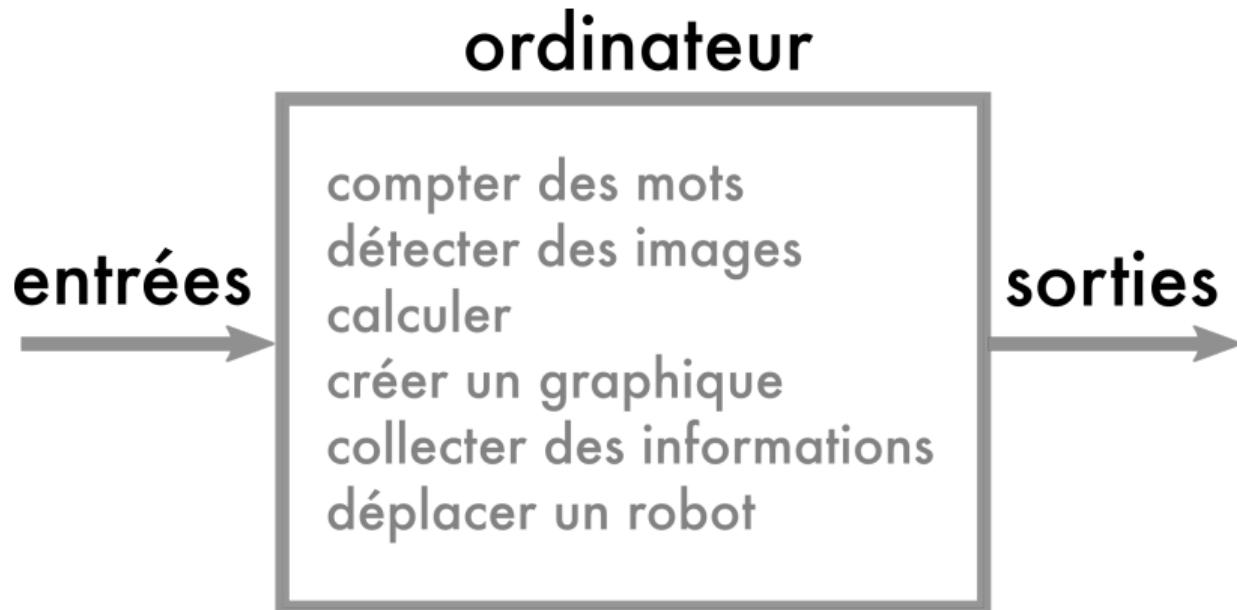
1. Pourquoi programmer ?

La numérisation de la recherche

- ▶ Traitement numérique comme point de passage obligé du•de la chercheur•se
 - ▶ *digital turn*
- ▶ Explosion de la disponibilité des données
 - ▶ *manipulation données*
- ▶ Courant profond et puissant de la science ouverte
 - ▶ *reproductibilité traitements*
- ▶ Apparition d'objets/méthodes liés aux pratiques numériques
 - ▶ *nouveaux terrain(s)*

Programmer !

Programmer[Définition pratique] : utiliser un ensemble de commandes (code) dans un langage (de programmation) pour faire réaliser (exécuter) à l'ordinateur des tâches.



Un **algorithme** est la description d'une suite d'étapes permettant d'obtenir un résultat à partir d'éléments fournis en entrée

Pour le faire : un ensemble de savoirs interdépendants

- ▶ Notions générales sur le fonctionnement d'un ordinateur (stockage, calcul, périphériques)
- ▶ Environnements spécifiques (OS et logiciels)
- ▶ Penser la logique des instructions : algorithmiques
 - ▶ ensemble ordonné d'instructions
- ▶ Exprimer ces instructions : langages de programmation
- ▶ Formats spécifiques des données
- ▶ Diversité d'outils/savoirs associés
 - ▶ Debugger

Les langages de programmation

Abstractions permettant de réaliser des opérations

- ▶ Des langages différents (plus ou moins abstraits)
- ▶ Des opérations partagées par tous les langages (opérations mathématiques)
- ▶ Infrastructure pour passer de l'opération à sa réalisation
 - ▶ Compilation (logiciel)
 - ▶ Interprétation

Cinquante nuance de programmation

- ▶ Des *styles* de programmation différentes (paradigmes)
 - ▶ Impératif/Procédural
 - ▶ Orienté objet
 - ▶ ...
- ▶ Un usage spécifique pour la recherche : **la programmation scientifique**
 - ▶ Orientation **script** : réaliser des petites tâches spécifiques
 - ▶ Orientation **interactive** : tester et expérimenter
 - ▶ Orientation **recherche** : des outils spécifiques
- ▶ Usage compatible avec des logiciels et le reste des pratiques

Proto-programmation : la ligne de commande (bash)

Donner des instructions à l'ordinateur dans un *terminal*. Permet d'interagir rapidement quand il n'y a pas d'environnement graphique (serveur) ou de lister les instructions

- ▶ Se déplacer dans un dossier : **cd ./dossier/**
- ▶ Supprimer un fichier : **rm nom_du_fichier**
- ▶ Créer un dossier : **mkdir nom_du_dossier**
- ▶ ...

Mais aussi appeler des petits programmes : **wc -w fichier.txt**, et de combiner les commandes ...

Pour plus de lecture : <https://melaniewalsh.github.io/Intro-Cultural-Analytics/01-Command-Line/01-The-Command-Line.html>

Programmer ≠ Construire un logiciel

E.-M. Arvanitou, A. Ampatzoglou, A. Chatzigeorgiou et al.

The Journal of Systems & Software 172 (2021) 110848

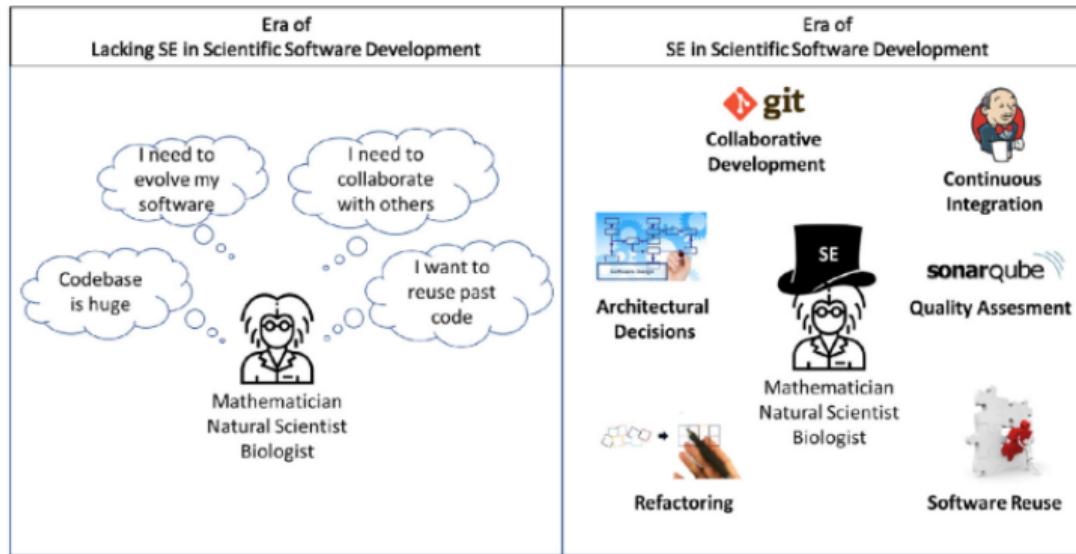


Fig. 1. State of practice and envisioned practices in scientific software development.

Pas forcément besoin de choisir



JANE-CLARK.TUMBLR

Programmer pour la recherche : entre standardisation et adaptation

- ▶ créer un dialogue interactif avec l'ordinateur (exploration et stabilisation)
- ▶ formaliser des manipulations pour les partager (reproductibilité des traitements)
- ▶ adapter à des tâches non prévues par les logiciels (flexibilité)
- ▶ interconnecter des opérations sinon séparées (glue)

Des approches ancrées dans la recherche : *literate programming*

Une pratique largement orientée data science, *plus "légère"*, avec ses outils dédiés.

Intégration du code et du texte (Knuth, 1992) puis des résultats dans la *literate computing*.

Casual Notebooks and Rigid Scripts: Understanding Data Science Programming

Krishna Subramanian, Nur Hamdan, Jan Borchers
RWTH Aachen University
52074 Aachen, Germany
{krishna, hamdan, borchers}@cs.rwth-aachen.de

Abstract—Data workers are non-professional data scientists who often use scripting languages like R, Python, or MATLAB, and employ an exploratory programming workflow. Current IDEs offer them two main programming modalities: script files and computational notebooks. To understand how these modalities impact work practice, we conducted a study with 21 data workers, and a subsequent larger survey with 62 respondents. Through interviews, walkthroughs, and screen recordings, we collected information about their workflows. Our analysis shows a tension between scripts and computational notebooks. Scripts are more common, better support storage and execution of previous analyses, but hinder experimentation. Notebooks better suit the actual data science workflow, but can become easily unorganized. We discuss how this dual nature of modality usage leads to several issues that affect data workers' workflows, and discuss implications for the design of programming IDEs.

Index Terms—scripting languages, exploratory programming, programming interfaces, data science, notebooks

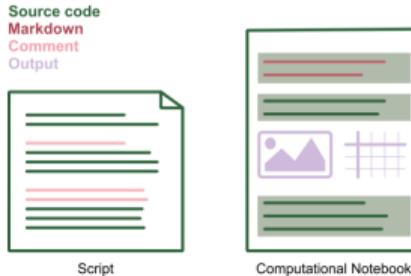


Fig. 1. Current scripting language IDEs support writing and executing code via two programming modalities: scripts (left) and computational notebooks (right). In this paper we investigate how these modalities are used in data

Programmer comme une formation par la pratique

Un effet **oignon** :

- ▶ Familiarise aux environnements informatiques
 - ▶ Ligne de commande, chemin d'accès, requête HTTP, ...
- ▶ Penser la structures des données et leurs diversité
 - ▶ Format de fichier : csv ou xls ? Passage vers du relationnel ?
- ▶ Penser la matérialité de nos pratiques et les solutions possibles
 - ▶ Stockage mémoire vive, cloud ou disque dur ?
- ▶ Favorise les nouvelles collaborations
 - ▶ Une langue commune entre spécialités

Un exemple : découvrir qu'une image est en fait un tableau de points, chaque point décrit par trois valeurs (rouge, vert, bleu), et qu'on peut manip

Renforcer les bonnes pratiques informatiques en recherche

De nouveaux mondes à explorer à partir de la programmation.



software carpentry

Teaching basic lab skills
for research computing

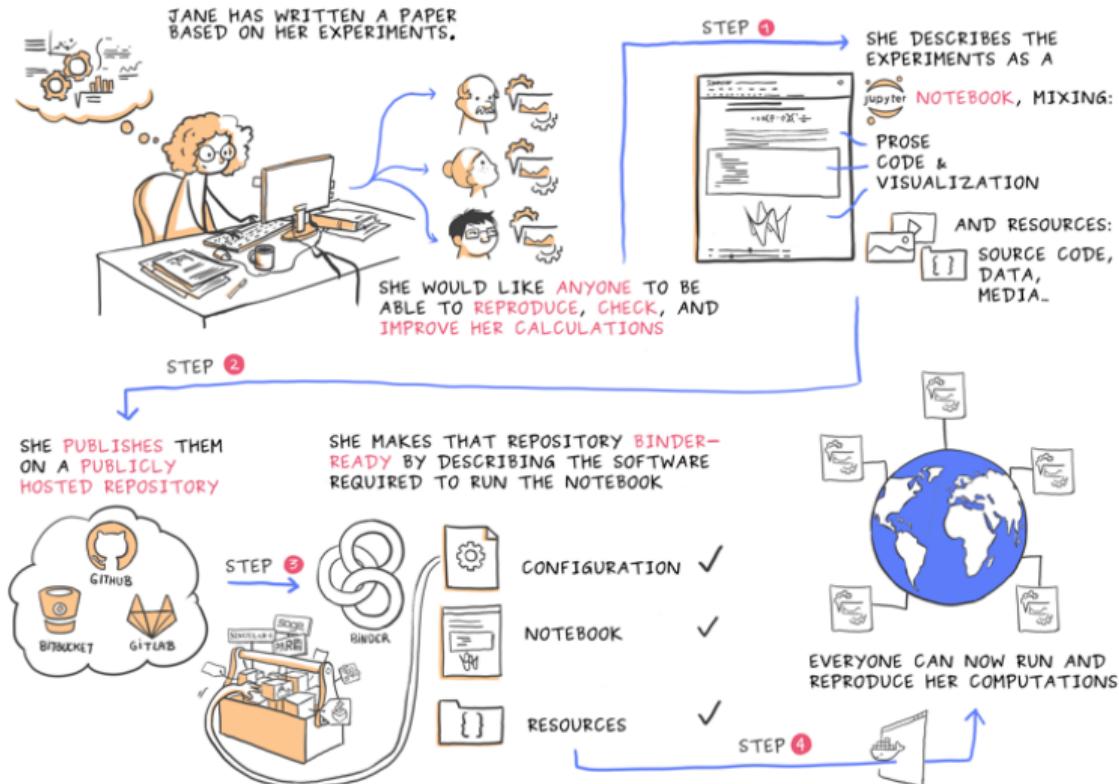
About Us

Since 1998, Software Carpentry has been teaching researchers the computing skills they need to get more done in less time and with less pain. Our [volunteer instructors](#) have run [hundreds of events](#) for more than 34,000 researchers since 2012. All of our [lesson materials](#) are freely reusable under the [Creative Commons - Attribution license](#).

The [Software Carpentry Foundation](#) and its sibling lesson project, [Data Carpentry](#), have merged to become The Carpentries, a fiscally sponsored project of [Community Initiatives](#), a 501(c)3 non-profit incorporated in the United States. See the [staff page for The Carpentries](#).

Wilson, Greg, Jennifer Bryan, Karen Cranston, Justin Kitzes, Lex Nederbragt, and Tracy K. Teal. 2017. "Good Enough Practices in Scientific Computing." Edited by Francis Ouellette. PLOS Computational Biology 13 (6) : e1005510.

Plus généralement, les enjeux de reproductibilité



Open and reproducible scientific workflow using Jupyter notebook and related open-source tools. Source : Juliette Taka and Nicolas M. Thiery. Publishing reproducible logbooks explainer comic strip. Zenodo. DOI : 10.5281/zenodo.4421040 (2018).

2. Pourquoi Python ?

Un monde de langages

Liste de langages de programmation

49 langues

Article Discussion

Lire Modifier Modifier le code Voir l'historique

Le but de cette liste de langages de programmation est d'inclure tous les langages de programmation existants, qu'ils soient actuellement utilisés ou historiques, par ordre alphabétique. Ne sont pas listés ici les langages informatiques de représentation de données tels que XML, HTML, XHTML ou YAML. Un langage de programmation doit permettre d'écrire des algorithmes, mais il n'est pas nécessaire qu'il soit Turing-complet (par exemple Gallina, le langage de programmation de Coq, ne l'est pas).

Par ailleurs, cette liste répertorie les langages de programmation, et non leurs implémentations (par exemple, JRuby et IronRuby sont deux implémentations différentes du même langage Ruby).

Sommaire : Haut - A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

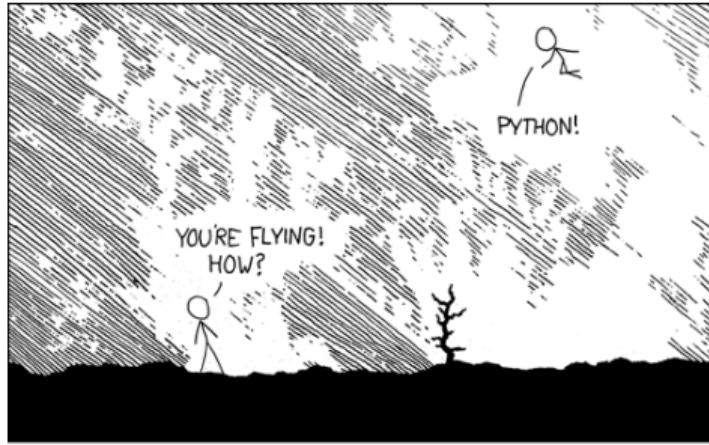
A [modifier | modifier le code]

- A+
- A++
- A# .NET
- A# (Axiom) (en)
- A-0 System
- ABAL
- ABAL++
- ABAP
- ABC
- ABCL/I
- ABCL/c+
- ABCL/R
- ABCL/R2
- Abel
- ABSET (en)
- ABSYS
- ALI
- Abundance
- ACC (programming language) (en)
- Accent
- ActForex
- Ace DASL
- ACT-III
- Ada
- Adenine
- Afrix
- Agora (programming language) (en)
- AIS Balise
- Aikido
- Alef
- Algebraic Logic Functional programming language (en)
- Algol 60
- Algol 68
- Algol W
- Alice (programming language) (en)
- Ambi
- Amiga E (en)
- AML
- AMOS
- AMPLE
- Anubis
- APDL
- APL
- AppleScript
- Arc
- Ariberion
- Arobase (langage)
- Assembleur
- ASP.NET
- ATS
- AUPL
- AutoHotkey
- AutoIt
- Averest
- awk
- axe parser
- Axum (programming language) (en)

B [modifier | modifier le code]

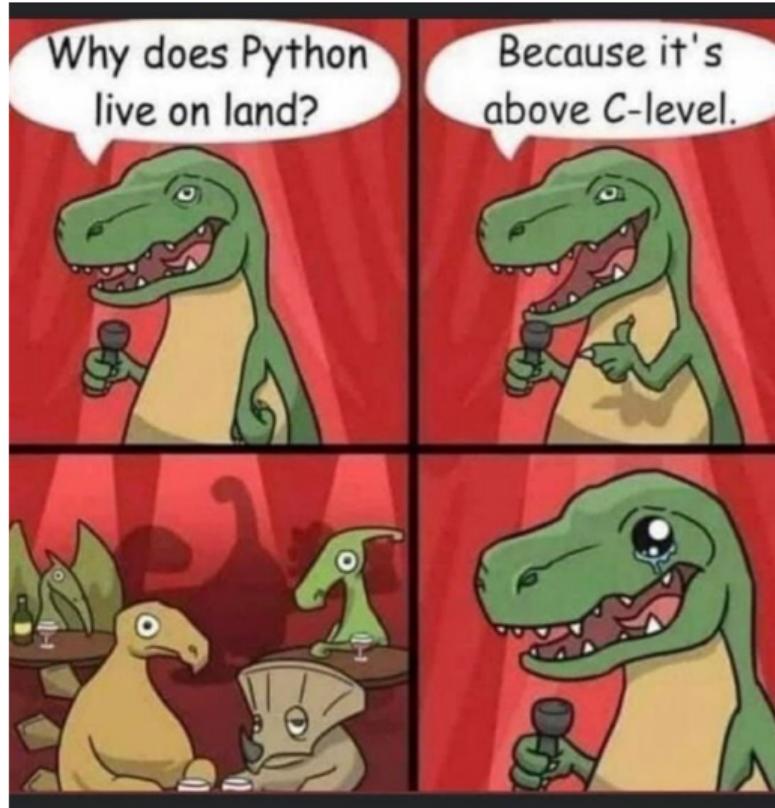
- B
- Bah-Lang
- BASIC
- BASICa
- Basic Nspire
- QuickBasic
- SmallBasic
- TI-Basic
- True Basic
- Turbo Basic
- Beef
- Befunge
- Bennu
- Bertrand
- BETA
- Bon
- Boo
- Boomerang
- Bosque
- Bourne shell (sh)

Tout est possible avec Python (sur un ordinateur)



<https://xkcd.com/353/>

Un langage de haut niveau



Propriétés de Python

- ▶ Libre et interopérable (interprété)
- ▶ *versatile* par rapport aux manières de l'utiliser
- ▶ Pédagogique *by design*
- ▶ De nombreuses ressources / documentation
- ▶ Favorise les bonnes pratiques de programmation
- ▶ En croissance d'usage (recherche et privé)
- ▶ Un avenir brillant : enseigné dès le lycée



Le produit d'une histoire et d'un écosystème

Une histoire qui construit sur les autres langages :

<https://inference-review.com/article/the-origins-of-python>

"It makes sense to think of the realm of programming languages as an ecosystem in which languages occupy their own niches. FORTRAN's niche is high-performance scientific programming, involving heavy-duty numerical computation ; that of COBOL is administration, based on files of data records. The C language is designed for systems programming, originally developed specifically for the Unix operating system. Just as there is no such thing as a general-purpose transportation vehicle, a truly one-size-fits-all general-purpose programming language does not exist ; for a given highly specialized application domain it will always be possible to design a language tailored to, and better suited for, the specific needs of that domain [...] Python was originally designed to serve as a high-level scripting language for the Amoeba project. ABC was completely unsuitable for this purpose ; it lived in a world of its own, shielding its users—by design—from the outside world. Python was expressly designed to interface with that outside world."

The Zen of Python

Tim Peters a résumé les principes du BDFL en 19 aphorismes :

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Flat is better than nested.

Sparse is better than dense.

Readability counts.

Special cases aren't special enough to break the rules.

Although practicality beats purity.

Errors should never pass silently.

Unless explicitly silenced.

In the face of ambiguity, refuse the temptation to guess.

There should be one— and preferably only one —obvious way to do it.

Although that way may not be obvious at first unless you're Dutch.

Now is better than never.

Although never is often better than *right* now.

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

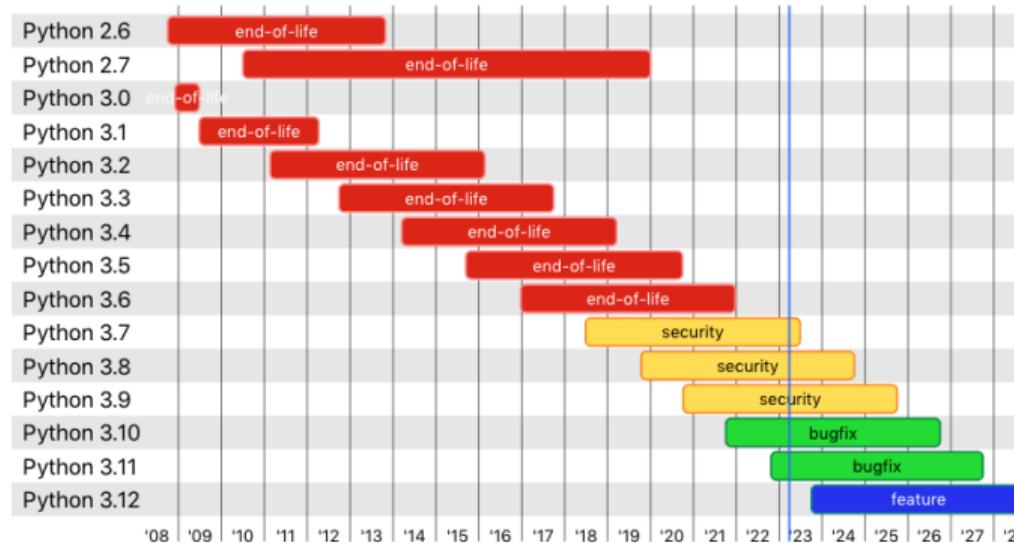
Namespaces are one honking great idea – let's do more of those !

Un langage qui a évolué et intégré les bonnes pratiques

Status of Python Versions

The main branch is currently the future Python 3.12, and is the only branch that accepts new features.
The latest release for each Python version can be found on the [download page](#).

Python Release Cycle



Plus qu'un langage : une communauté de chercheurs



SciPy:

- Home
- News
- Downloads
- Build/Test Scoreboard
- Bug Tracker
- View CVS
- Mailing Lists
- Members
- Search
- Envoyer

SciPy Help:

- Documentation and Tutorials
- FAQ
- Screen Shots
- Using CVS
- Doing a binary install
- Building from source
- Membership Tips

Chaco:

- Home

Guest:

- Log in
- Join



Status: published

- View

SciPy '02



Python for Scientific Computing Workshop

CalTech, Pasadena, CA
September 5-6, 2002

This workshop provides a unique opportunity to learn and affect what is happening in the realm of scientific computing with Python. Attendees will have the opportunity to review the available tools and how they apply to specific problems. By providing a forum for developers to share their Python expertise with the wider industrial, academic, and research communities, this workshop will foster collaboration and facilitate the sharing of software components, techniques and a vision for high level language use in scientific computing.

The two-day workshop will be a mix of invited talks and training sessions in the morning. The afternoons will be breakout sessions with the intent of getting standardization of tools and interfaces.

By all (mostly biased) accounts, the workshop was a great success. Thanks to all who helped organize and sponsor the meeting. [Click Here](#) to see results of the workshop as they become available.

- [Campus Map with Parking and Meeting Locations](#)
- [Agenda](#)
- [Tutorials Descriptions](#)
- [Registration](#)
- [Directions & Lodging](#)

Discussion about the workshop may be directed to the [SciPy-user mailing list](#) to which you may post messages at scipy-user@scipy.org

Co-Hosted By:

[The National Biomedical Computation Resource](#) (NBCR, SDSC, San Diego, CA)

The mission of the National Biomedical Computation Resource at the San Diego Supercomputer Center is to conduct, catalyze, and enable biomedical research by harnessing advanced computational technology.

[The Center for Advanced Computing Research](#) (CACR, CalTech, Pasadena, CA)

CACR is dedicated to the pursuit of excellence in the field of high-performance computing, communication, and data engineering. Major activities include carrying out large-scale scientific and engineering applications on parallel supercomputers and coordinating collaborative research projects on high-speed network technologies, distributed computing and database methodologies, and related topics. Our goal is to help further the state of the art in scientific computing.



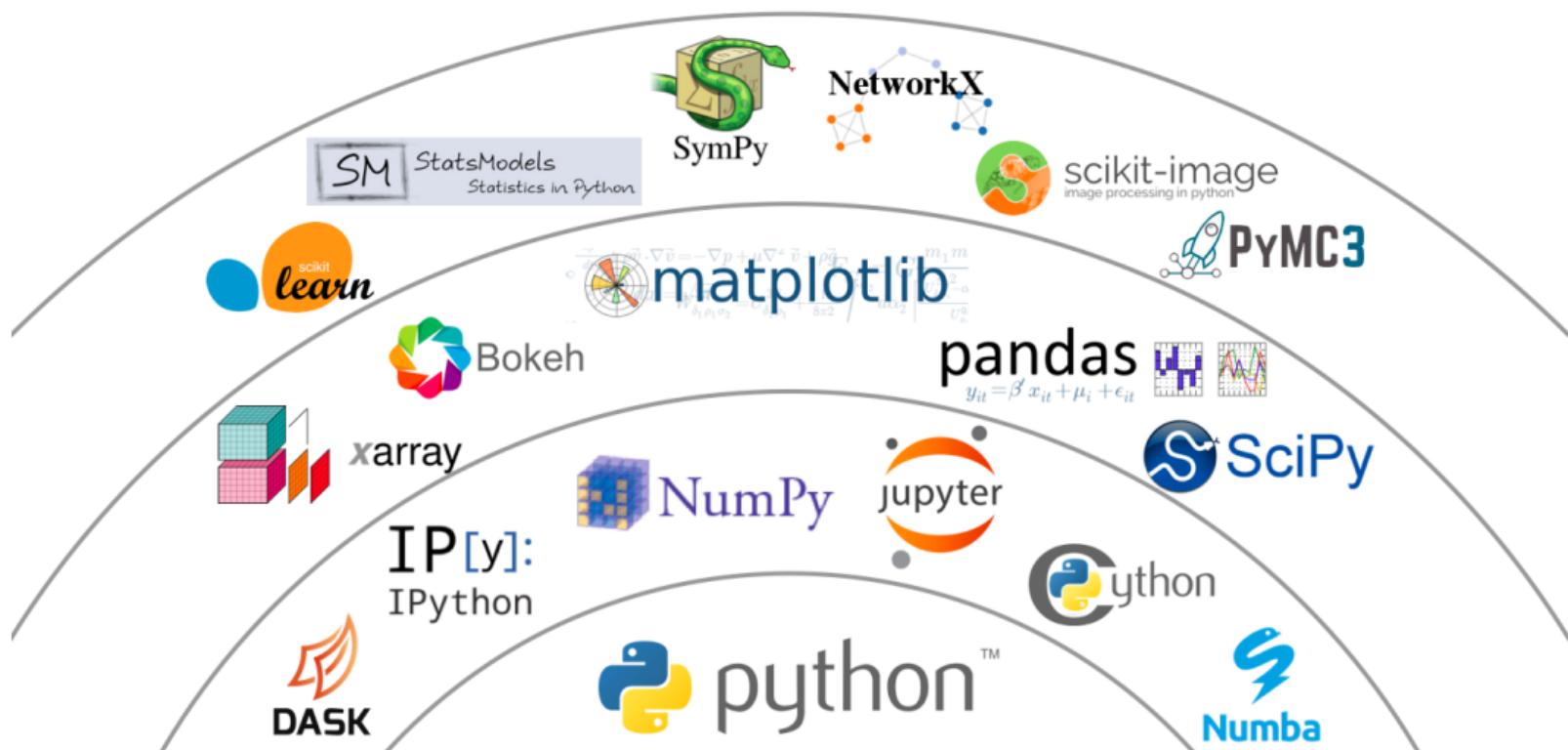
[Enthought, Inc.](#) (Austin, TX)

Enthought, Inc. provides business and scientific computing solutions through software development, consulting and training.

Plus qu'un langage : un univers d'outils

Python's Scientific Stack

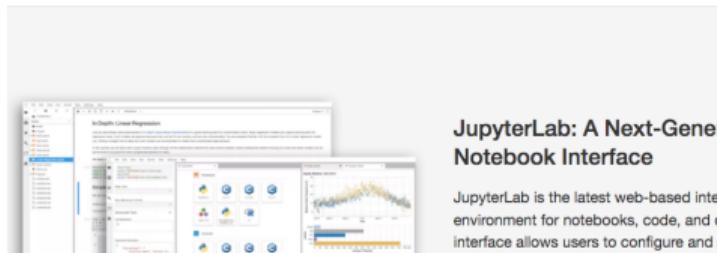
Jake Vanderplas PyCon 2017 Keynote



De nombreux outils

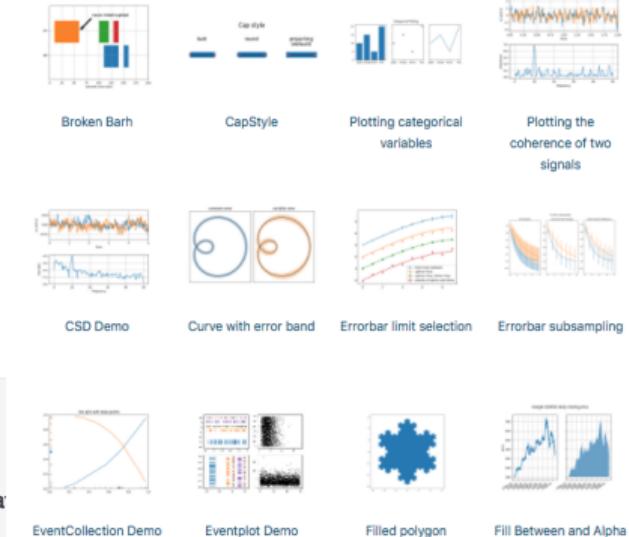


Free software, open standards, and web services for interactive computing across all programming languages



JupyterLab: A Next-Generation Notebook Interface

JupyterLab is the latest web-based interface for notebooks, code, and data. This interface allows users to configure and arrange workflows in



Gallerie Matplotlib

Lines, bars and markers
Images, contours and fields
Subplots, axes and figures
Statistics
Pie and polar charts
Text, labels and annotations
pyplot
Color
Shapes and collections
Style sheets
axes_grid1
axisartist
Showcase
Animation
Event handling
Front Page
Miscellaneous
3D plotting
Scales
Specialty Plots
Spines
Ticks
Units
Embedding Matplotlib in graphical user interfaces
Userdemo
Widgets

Des bibliothèques puissantes

learn Install User Guide API Examples Community More ▾

scikit-learn

Machine Learning in Python

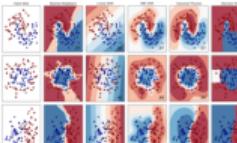
Getting Started Release Highlights for 1.0 GitHub

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.

Algorithms: SVM, nearest neighbors, random forest, and more...

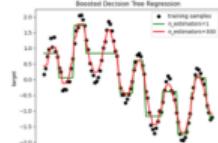


Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.

Algorithms: SVR, nearest neighbors, random forest, and more...



Clustering

Automatic grouping of objects into sets.

Applications: Customer segmentation, Grouping experiment on digits.

Algorithms: k-Means, hierarchical clustering, mean-shift, and more...



Simple and efficient tools for predictive data analysis
Accessible to everybody, and reusable in contexts
Built on NumPy, SciPy, and matplotlib
Open source, commercially usable -

spaCy *Out now: spaCy v3.3

USAGE MODELS API UNIVERSE 23,242 Search docs

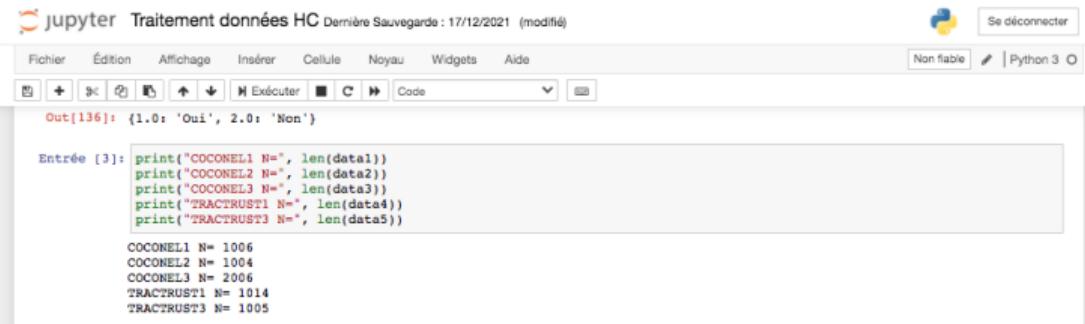
Industrial-Strength Natural Language Processing

IN PYTHON

Get things done

spaCy is designed to help you do real work — to build real products, or gather real insights. The library respects your time, and tries to avoid wasting it. It's easy to install, and its API is simple and productive.

Permettant de construire des workflows complets



A screenshot of a Jupyter Notebook interface. The top bar shows the title "jupyter Traitement données HC Dernière Sauvegarde: 17/12/2021 (modifié)". The menu bar includes Fichier, Édition, Affichage, Insérer, Cellule, Noyau, Widgets, Aide. On the right, there are buttons for "Se déconnecter", "Non fiable", and "Python 3". Below the menu is a toolbar with various icons. The main area shows a code cell and its output.

```
Out[136]: {1.0: 'Oui', 2.0: 'Non'}
```

```
Entrée [3]: print("COCONEL1 N=", len(data1))
print("COCONEL2 N=", len(data2))
print("COCONEL3 N=", len(data3))
print("TRACTRUST1 N=", len(data4))
print("TRACTRUST3 N=", len(data5))

COCONEL1 N= 1006
COCONEL2 N= 1004
COCONEL3 N= 2006
TRACTRUST1 N= 1014
TRACTRUST3 N= 1005
```

[FIGURE 1] Evolution de l'attitude en France



```
Entrée [9]: # Tableau par enquête
d = {'04-07-2020': pyhs.tri_a_plat(data1,"HC_c","RED")["Pourcentage (%)"],
      '04-19-2020': pyhs.tri_a_plat(data1,"HC_c","RED")["Pourcentage (%)"],
      '06-23-2020': pyhs.tri_a_plat(data1,"HC_c","RED")["Pourcentage (%)"],
      '11-03-2020': pyhs.tri_a_plat(data1,"HC_c","RED")["Pourcentage (%)"],
      '06-08-2021': pyhs.tri_a_plat(data1,"HC_c","RED")["Pourcentage (%)"]
t = pd.concat(d, axis=1).drop("Total").T

# Données Google Trends
hc = pd.read_csv("./multiTimeline.csv").replace({'\xa0':0})
hc["chloroquine: (France)"] = hc["chloroquine: (France)"].apply(int)
hc["hydroxychloroquine: (France)"] = hc["hydroxychloroquine: (France)"].apply(int)
hc["Semaine"] = pd.to_datetime(hc["Semaine"])
hc = hc.set_index("Semaine")["chloroquine: (France)"]

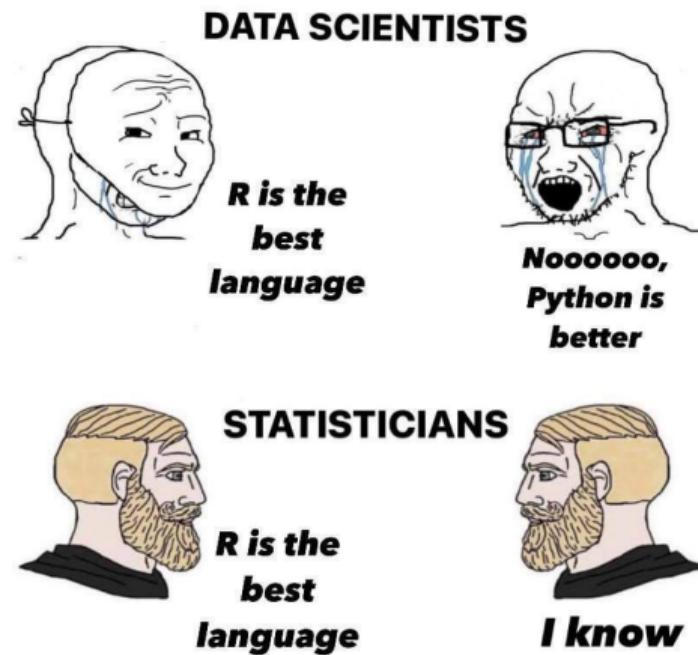
# Graphique
t.index = pd.to_datetime(t.index)
ax = t.plot(color=['r','g','b'], figsize=(10,5),marker='o', linestyle='--')
pd.DataFrame(hc.resample('w').sum()).plot(ax=ax,color="gray")
plt.xlim("2020-02-01","2021-06-20")
plt.xlabel("Date (per week)")
plt.ylabel("Percentage (%)")
plt.legend(["HC is effective","HC is ineffective","Uncertain","Intensity of Google searches using Google Trends"])
plt.title("Figure 1. Evolution of attitudes toward HC in France and media coverage between April 2020 and June 2021")

plt.tight_layout()
plt.savefig("../figures/Figure 1 - evolution.png",dpi=1000)
```



Mais pas le seul choix...

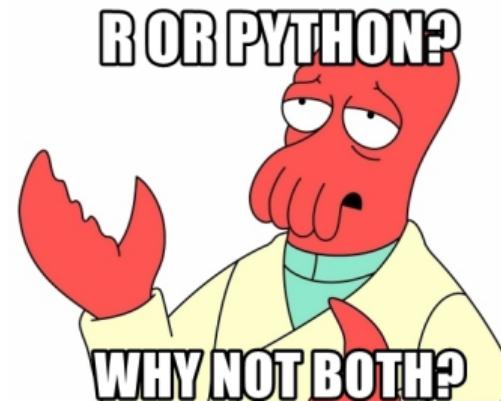
Convergence et divergences avec d'autres langages, R en premier lieu



Qui mène à la question centrale : dois-je choisir Python ?

Python ou R ? Python et R ? Tous ensemble dans la programmation scientifique

- ▶ Python et R permettent la majorité des traitements associés à la collecte des données, au traitement, et à la visualisation, et évoluent en permanence.
- ▶ Python est davantage compris par les informaticiens et assimilés + secteur privé
- ▶ R excellent pour les statistiques
- ▶ Python est en avance pour les applications en machine learning
- ▶ Python permet de déployer
- ▶ Python semble avoir une meilleure logique de documentation



Dans tous les cas, important d'initier d'un usage et importance des ressources disponibles pour apprendre : collègues, etc.

3. Pourquoi réfléchir les usages spécifiques aux SHS ?

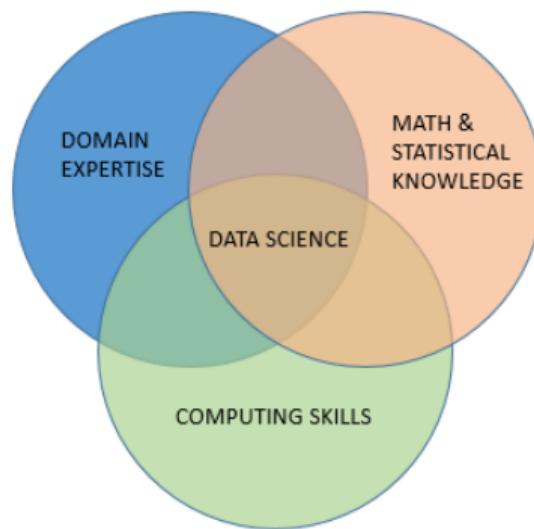
Des identités en transformation autour du numérique

Revenir à la poussière ? L'identité professionnelle des historiens et historiennes

Le livre d'Arlette Farge (1989) a connu un tel succès national et international qu'il semble avoir contribué à stabiliser la définition même du métier d'historien et d'historienne autour de celui ou celle qui noircit ses mains de poussière, qui « descend aux archives », etc. C'est la raison pour laquelle les médiations numériques sont très peu évoquées dans les remerciements de thèse, les blogs ou, plus simplement, les livres : historiens et historiennes seraient prisonniers de « faux récits de l'archive » qui le conduisent à valoriser la mise en scène du contact physique au document plutôt que la réalité du travail derrière l'écran ou la fouille *via* les moteurs de recherche⁸. Un certain « récit de l'archive », déphasé par rapport aux pratiques réelles, reste central dans la construction de l'identité professionnelle. La numérisation du métier est pourtant bien avancée : rares sont les gestes qui ne sont pas médiés par l'ordinateur ou l'instrument, scanner, téléphone ou encore appareil photo. Comment expliquer ce décalage entre récit de l'archive et pratiques concrètes ? Le déni de la numérisation du métier dans la présentation des coulisses des enquêtes historiques révèle la force des représentations qui lient empathie, imprégnation du passé et immersion dans des cartons de documents physiques. Quels seraient des récits d'archive plus proches des

L'autonomisation de la "data science"

- ▶ De plus en plus autonome comme littérature (manuels dédiés, beaucoup tournés vers l'opérationnel)
- ▶ Toujours relatif à des domaines spécifiques



Hétérogénéité des SHS

- ▶ Rôle central de la problématique (perspectivisme)
- ▶ Méthodologies très variées
- ▶ Données plus ou moins accessibles et normalisées
- ▶ Culture du numérique variable



Même entre les usagers de statistiques

Une multitude d'usages et de logiciels, avec leurs cultures



Des dynamiques en cours

4. FOCUS SUR 3 OUTILS NUMÉRIQUES ET 3 LOGIQUES D'INNOVATION

Nous procédons à une analyse plus approfondie des 3 premiers outils les plus cités : Excel, R et Python. Leurs caractéristiques propres en font à la fois des « concurrents » et des outils complémentaires. Notre analyse tente d'évaluer si l'on peut trouver des profils de chercheurs, qui par leurs caractéristiques propres peuvent être associés à chacun de ces trois outils. Nous constatons que nous rencontrons trois configurations. Nous rencontrons l'innovation : en voie d'institutionnalisation (N. Alter, 2015) symbolisée par R; le logiciel institutionnalisé représenté par Excel; et la pratique en émergence avec Python.

Les utilisateurs de R (n = 244): la voie de l'institutionnalisation

Une moyenne d'âge des utilisateurs de R plus jeune

Les utilisateurs de R se caractérisent par une moyenne d'âge et un âge médian inférieur d'environ 4 ans à la POP. L'usage de R est lié à des chercheurs parmi les plus jeunes, les écarts étant sensibles pour les 35-45 ans et nettement plus marqués pour les chercheurs de moins de 35 ans.

Constats (à discuter)

- ▶ Une division persistante quanti/quali que la programmation permet de dépasser
- ▶ Des usages "discrets" plus que "computationnels" à identifier
- ▶ Constat d'une limite des exemples disponibles : que faire ?
- ▶ Programmation souvent ramenée aux statistiques (et à R)
- ▶ Encore peu de bibliothèques Python dédiées SHS (donc de la place pour en développer de nouvelles)
- ▶ Des usages encore peu stabilisés (Notebooks, etc.)
- ▶ Division du travail vs. culture partagée

Un gros potentiel d'interface entre les pratiques. Mais un état des lieux en cours.

4. En pratique, ça sert à quoi ?

Cas : format de données

Passer d'un fichier *.html* à un *.txt* mis en forme pour l'ramuteq

Les Echos, no. 23183
événement, vendredi 20 mars 2020 613 mots, p. 3

Coronavirus

Aussi perdu dans le temps - 19 mars 2020 - lesechoes.fr

Les cliniques privées à la rescoussse
SOLVEIG GODELUCK

En Alsace, où les hôpitaux publics sont débordés, les établissements privés sont également sollicités

Certains sont dans la tempête; d'autres l'attendent. Ainsi, à Strasbourg, au sein de l'hôpital Saint-Vincent, le directeur général de la Fondation Saint-Vincent à Strasbourg, Christophe M...

Des lits transformés pour la réanimation

Ces disponibilités ont pourtant été signalées par le directeur de l'établissement : « Nous avons pu ouvrir des lits de réanimation dans les salles d'opération », explique Christophe M...

« Nous ne sommes pas sollicités à hauteur du service de réanimation », assure Samu : on oriente les malades vers le secteur public. Lorsque les deux jours, on a déprogrammé toutes nos opérations.

100.000 soins déprogrammés dans le privé lucratif



renforcement » dans d'autres. Le lendemain, le ministre de l'Intérieur a lui-même été infecté, a annoncé l'extension des tests de dépistage et a déclaré qu'il se lancerait dans le déconfinement Sophie Amsili et Tiffen Clémentine.

***** *num 618 *journal LeFigaro

«Pendant trois heures, Emmanuel Macron a pris connaissance à résultats obtenus par l'équipe du Pr Raoult», se réjouit la **Martine Wonner**, seule parlementaire LREM à «**Covid-19-Laissons les médecins prescrire**.» **LIRE AUSSI -** **Rapport** : les dessous d'une rencontre surprise. Cette psychologue, dont les positions souvent plus tranchées que celles de ses collègues. Elle s'était aussi engagée avec les écologistes, contre le tournoiement ouest de Strasbourg, dont l'énorme chantier a

 IRaMuTeQ

Ou encore : passer d'un fichier .pdf à un .txt pour faire du TAL

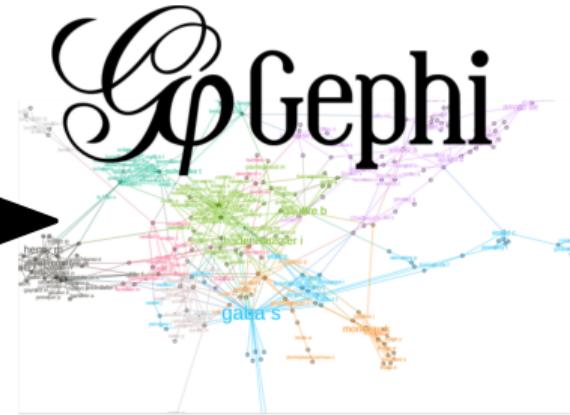
Cas : construire un réseau

Créer la bonne structure relationnelle (ici auteur/auteur) et l'exporter dans un format compatible avec Gephi

A	B	C	D	E
ID	ANNEE	AUTEURS	TITRE	JOURNAL
2	35	1996 LEROUX A., BRETAGNOLLE V	Sex ratio variations in broods of Montagu's harrier	Journal of Avian Biology
3	37	1996 A RECORDER	SALAMOLARD M., BRETAGNOLLE V.	
4	44	1998 ARROYO B.E., LEROUX A.B.A.	Egg and clutch	Journal of Reproduction and Development
5	47	1998 de CORNUNIER T., BERNARD R.	Nidification of Redstarts	
6	52	1999 ARROYO B.E., BRETAGNOLLE V.	Breeding biology	Journal of Reproduction and Development
7	55	1999 SALAMOLARD M., MOREAU C.	Habitat selection	Bird Study
8	58	2000 AMAR A., ARROYO B.E., BRETAGNOLLE V.	Post-fledging Ibis	Ibis
9	59	2000 ARROYO B.E., DECORNULIER T.	Sex and age of Condors	Condor
10	62	2000 GUILLEMAIN M., HOUTTE S., FRIJ	Activities and Revue d'Ecologie	Ecology
11	63	2000 JIGUET F., ARROYO B., BRETAGNOLLE V.	Lek mating systems	Behavioural Ecology
12	68	2000 SALAMOLARD M., BUTET A.	Responses of Ecology	Ecology
13	69	2001 ARROYO B., MOUGEOU F., BRETAGNOLLE V.	Colonial birds	Behavioral Ecology
14	70	2001 CLERE E., BRETAGNOLLE V.	Disponibilité	Revue d'Ecologie
15	71	2001 JIGUET F., BRETAGNOLLE V.	Courtship behaviour	Behavioural Ecology



```
<?xml version="1.0" encoding="utf-8"?>
graph TD
    subgraph "http://graphml.graphdrawing.org/mlns"
        min["min<br>http://graphml.graphdrawing.org/mlns"]
        max["max<br>http://graphml.graphdrawing.org/mlns"]
        title["title<br>http://graphml.graphdrawing.org/mlns"]
        id["id<br>http://graphml.graphdrawing.org/mlns"]
        type["type<br>http://graphml.graphdrawing.org/mlns"]
        name["name<br>http://graphml.graphdrawing.org/mlns"]
        cluster["cluster<br>http://graphml.graphdrawing.org/mlns"]
        string["string<br>http://graphml.graphdrawing.org/mlns"]
        long["long<br>http://graphml.graphdrawing.org/mlns"]
        label["label<br>http://graphml.graphdrawing.org/mlns"]
    end
    graph TD
        node0[Sex ratio variations in broods of Montagu's harrier] --- node1[1996]
        node1 --- node2[Article]
        node2 --- node3[5c]
        node3 --- node4[Sex ratio]
        node4 --- node5[Sex ratio]
    
```



Cas : utiliser un tutorial existant

Avec l'arrivée des modèles (Huggingface & co), beaucoup d'usages émergent comme la retranscription.



Transcription d'entretiens avec Whisper et le SSP Cloud ↗

Ce tutoriel cherche à proposer une **solution simple et rapide** aux personnes souhaitant **retranscrire des enregistrements audio** grâce au module [Whisper](#) de [OpenAI](#) qui ne nécessite ni l'installation de logiciel, ni de disposer d'un ordinateur puissant, ni une bonne connaissance de [Python](#) et [Bash](#). Il ne présente pas toutes les potentialités de Whisper. Les informations complètes sur toutes ces fonctionnalités sont disponibles sur le dépôt [Github d'OpenAI](#). Par ailleurs, la **retranscription automatisée d'entretiens sociologiques** avec WhisperOpenAI soulève un certain nombre de questions très bien évoquées par Yacine Chitour dans [ce tutoriel](#) dont je recommande vivement la lecture.

<https://github.com/anoukmartin/Transcription-Whisper-x-SSP-Cloud>

Cas : construction de tableaux adaptés

Produire des sorties de tableaux adaptés à l'objet (et possibilité ensuite d'aller sur Excel ou Latex)

```
Entrée [64]: var_ind = {"sexe":"1 - Sex","age2":"2 - Age","diplome":"3 - Education", "revenus":"4 - Incomes",
                     "PROXPARTI":"5 - Political orientation"}

t = {"COCONEL1":pyshs.tableau_croise_multiple(data1,"HC_c",var_ind,chi2=False)[["1 - HC effective","2 - HC not effect",
                     "COCONEL2":pyshs.tableau_croise_multiple(data2,"HC_c",var_ind,chi2=False)[["1 - HC effective","2 - HC not effect",
                     "COCONEL3":pyshs.tableau_croise_multiple(data3,"HC_c",var_ind,chi2=False)[["1 - HC effective","2 - HC not effect",
                     "TRACTRUST1":pyshs.tableau_croise_multiple(data4,"HC_c",var_ind,chi2=False)[["1 - HC effective","2 - HC not effect",
                     "TRACTRUST2":pyshs.tableau_croise_multiple(data5,"HC_c",var_ind,chi2=False)[["1 - HC effective","2 - HC not effect

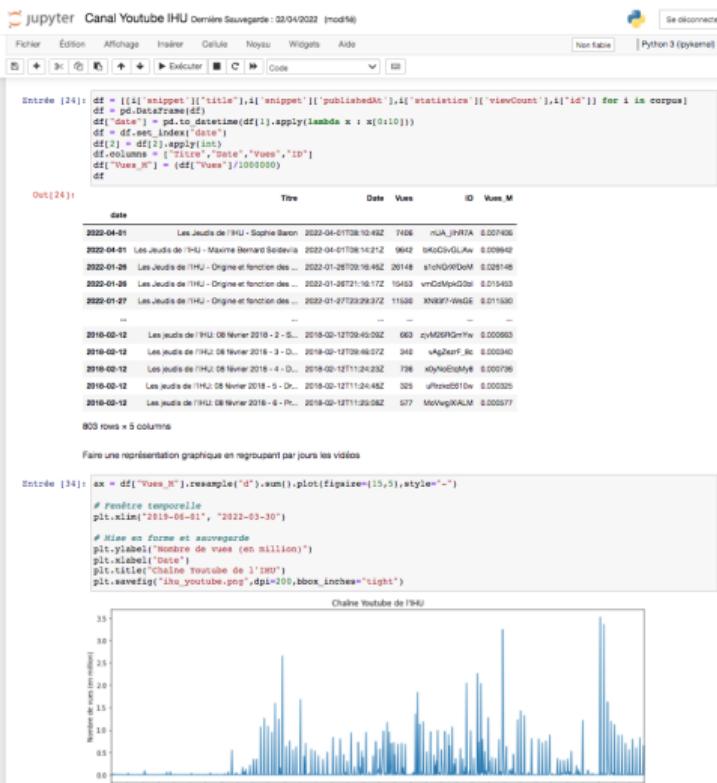
t = pd.concat(t,axis=1)
t.applymap(lambda x : re.findall("\((.*?)%\)",x)[0])
```

Out[64]:

Variable	Modalités	COCONEL1		COCONEL2		COCONEL3		TRACTRUST1		TRACTRUST2	
		1 - HC effective	2 - HC not effective	1 - HC effective	2 - HC not effective	1 - HC effective	2 - HC not effective	1 - HC effective	2 - HC not effective	1 - HC effective	2 - HC not effective
1 - Sex	Femme	38.3	3.9	34.0	9.1	17.8	9.0	14.2	13.4	15.8	18.8
	Homme	36.8	7.4	27.2	13.6	21.6	14.7	19.5	19.0	16.2	29.1
	Total	37.6	5.6	30.8	11.3	19.6	11.7	16.7	16.1	16.0	23.9
2 - Age	17-34	36.7	8.9	27.8	15.4	16.8	14.7	14.6	20.4	14.4	25.8
	35-54	41.1	4.5	31.3	10.1	19.9	11.8	18.4	14.2	15.8	23.9
	55-79	36.8	4.0	33.3	10.2	23.3	8.9	17.7	16.7	18.8	20.1
3 - Education	70-100	33.3	4.5	31.0	8.4	19.1	9.6	14.9	11.8	15.5	25.7
	Total	37.6	5.6	30.8	11.3	19.6	11.7	16.7	16.1	16.0	23.9
	1 - inf bac	33.2	5.3	34.8	8.4	21.3	8.0	18.7	8.3	14.8	15.1
4 - Revenus	2 - bac	42.3	4.7	33.5	9.3	21.4	9.9	17.5	14.0	19.0	21.3
	3 - sup bac	37.5	6.1	27.0	13.9	17.5	15.0	15.0	22.0	15.2	30.8

Cas : exploration de données de l'API aux statistiques

Exploration d'un tableau de données (ici le nombre de vues par vidéos de la chaîne Youtube de l'IHU)



Cas : codage de matériel qualitatif

TAGUETTE

TAGUETTE

NOURRIT VOS BESOINS QUALITATIFS

Avez-vous déjà cherché "analyse qualitative gratuit" et solution qui vous permet de taguer vos documents ? Taguette est un logiciel libre et gratuit pour l'analyse de contenu. Vous pouvez importer vos documents, marquer des extraits de texte et exporter les résultats !

[En savoir plus »](#) [Essayez Taguette sur notre serveur](#)

doccano

code quality A doccano CI passing

doccano is an open source text annotation tool for humans. It provides annotation features for text classification, sequence labeling and sequence to sequence tasks. So, you can create labeled data for sentiment analysis, named entity recognition, text summarization and so on. Just create a project, upload data and start annotating. You can build a dataset in hours.

Demo

You can try the [annotation demo](#).

Key	Value
id	4940272
Item	1946
Political party	Republican
Surname	Melania Knauss
Parents	Fred Trump, Mary Anne MacLeod

Cas : figures d'un article faciles à reproduire

Production des statistiques et des figures facile à relancer en cas de révision de l'article.

Open Access Article

French Public Familiarity and Attitudes toward Clinical Research during the COVID-19 Pandemic

by  Émilien Schultz 1,2,*  Jeremy K. Ward 3,4  Laëtitia Atlani-Duault 1,5,6  Seth M. Holmes 2,7,8  and  Julien Mancini 2,9

¹ CEPED (UMR 196), Université de Paris, IRD, 75006 Paris, France
² SESSTIM, Sciences Économiques & Sociales de la Santé & Traitement de l'Information Médicale, CANBIOS Team (Équipe Labelisée LIGUE 2019), Aix-Marseille University, INSERM, IRD, 13009 Marseille, France
³ CERMES3, INSERM, CNRS, EHESS, Université de Paris, 94801 Villejuif, France
⁴ VITROME, Aix-Marseille University, IRD, AP-HM, SSA, 13005 Marseille, France
⁵ Institut COVID-19 Add Memoriam, University of Paris, 75006 Paris, France
⁶ WHO Collaborative Center for Research on Health and Humanitarian Policies and Practices, IRD, Université de Paris, 75006 Paris, France
⁷ Society and Environment, Medical Anthropology, and Public Health, University of Berkeley, Berkeley, CA 94720, USA
⁸ Mediterranean Institute for Advanced Study IMéRA, Institut Paoli Calmettes, Aix-Marseille University, 13004 Marseille, France
⁹ BioSTIC, APHM, Timone, 13005 Marseille, France

* Author to whom correspondence should be addressed.
† Current address: CEPED, 45 Rue des Saints-Pères, 75006 Paris, France.

Academic Editor: Roy McConkey

Int. J. Environ. Res. Public Health **2021**, *18*(5), 2611; <https://doi.org/10.3390/ijerph18052611>

Received: 2 February 2021 / Revised: 2 March 2021 / Accepted: 2 March 2021 / Published: 5 March 2021

(This article belongs to the Section Global Health)

[View Full-Text](#) [Download PDF](#) [Browse Figures](#) [Citation Export](#)

Abstract

The COVID-19 pandemic put clinical research in the media spotlight globally. This article proposes a first measure of familiarity with and attitude toward clinical research in France. Drawing from the “Health Literacy Survey 2019” (HLS19) conducted online between 27 May and 5 June 2020 on a sample of the French adult population ($N = 1003$), we show that a significant proportion of the French population claimed some familiarity with clinical trials (64.8%) and had positive attitudes (72%) toward them. One of the important findings of this study is that positive attitudes toward clinical research exist side by side with a strong distancing from the pharmaceutical industry. While respondents acknowledged that the pharmaceutical industry plays an important role in clinical

Cas : diffuser ses outils à la communauté

AugmentedSocialScientist 2.2.1

`pip install AugmentedSocialScientist`

Released: Oct 30, 2023

A Package to Easily Train Bert-Like Models for Text Classification

Navigation

Project description

Release history

Download files

Project links

Homepage

Download

Project description

The Augmented Social Scientist

New release v2

This package makes it extremely easy to train BERT-like models for text classifications.

For more information on the method and for some use cases from social sciences, see "[The Augmented Social Scientist: Using Sequential Transfer Learning to Annotate Millions of Texts with Human-Level Accuracy](#)" published on *Sociological Methods & Research* by [Salomé Do](#), [Étienne Ollion](#) and [Rubing Shen](#).

To install the package

Plus généralement, les sciences sociales computationnelles

Sociological Methods & Research

Impact Factor: 4.677 / 5-Year Impact Factor: 5.424 JOURNAL HOMEPAGE

Restricted access | Research article | First published online December 4, 2022

The Augmented Social Scientist: Using Sequential Transfer Learning to Annotate Millions of Texts with Human-Level Accuracy

Salomé Do , Étienne Oillion  , and Rubing Shen  [View all authors and affiliations](#)

OnlineFirst | <https://doi.org/10.1177/00491241221134526>

Contents | Get access | Cite article | Share options | Information, rights and permissions | Metrics and citations

Abstract

The last decade witnessed a spectacular rise in the volume of available textual data. With this new abundance came the question of how to analyze it. In the social sciences, scholars mostly resorted to two well-established approaches, human annotation on sampled data on the one hand (either performed by the researcher, or outsourced to microworkers), and quantitative methods on the other. Each approach has its own merits - a potentially very fine-grained analysis for the former, a very scalable one for the latter - but the combination of these two properties has not yielded highly accurate results so far. Leveraging recent advances in sequential transfer learning, we demonstrate via an experiment that an expert can train a precise, efficient automatic classifier in a very limited amount of time. We also show that, under certain conditions, expert-trained models produce better annotations than humans themselves. We demonstrate these points using a classic research question in the sociology of journalism, the rise of a "horse race" coverage of politics. We conclude that recent advances in transfer learning help us augment ourselves when analyzing unstructured data.

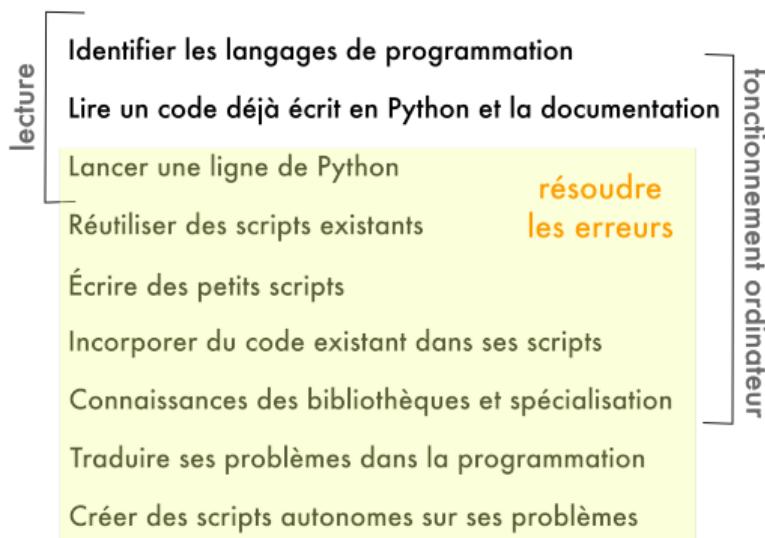
Autres usages

- ▶ Garder une mémoire de ses traitements.
- ▶ Collaboration autour des données : partager son code, faire relire ses résultats intermédiaires
- ▶ Traitement massif de données : parallélisation, déploiement sur des grandes infrastructures, recours aux outils du machine learning
- ▶ Créer une interface utilisateur pour accéder à vos données.
- ▶ Traitement des images.

5. S'y mettre !

Apprendre à programmer : pas une définition univoque

Découvre la programmation



Les obstacles

- ▶ Un outil parmi d'autres : **pas une baguette magique**
- ▶ Courbe d'apprentissage potentiellement longue (mais...)
- ▶ Avoir une idée de quoi en faire : quel imaginaire pratique ?
- ▶ Trouver des ressources locales : importance de la pratique



Programmer ≠ Tout savoir

Apprendre à programmer signifie apprendre à potentiellement pouvoir utiliser de nombreux outils développés par des chercheurs.

Mais chaque domaine a ses savoirs spécifiques : *machine learning*, analyse de réseaux, textométrie, ... Il faut aussi assurer les savoirs théoriques associés à chaque outil.

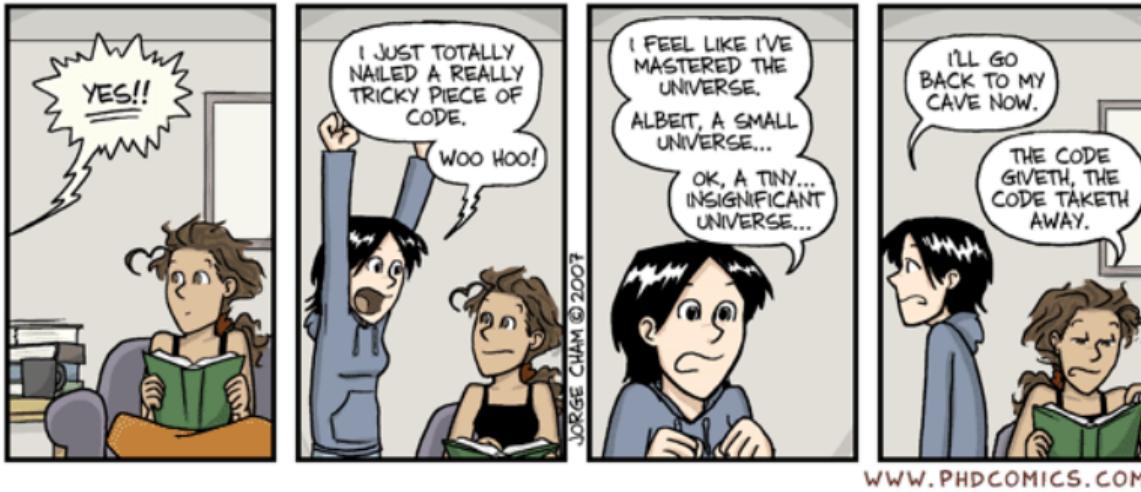
La frontière peut être difficile à tracer.

- ▶ Réutilisation d'outils facilité
- ▶ Mais cela ne replace pas une connaissance experte

Comment progresser ?

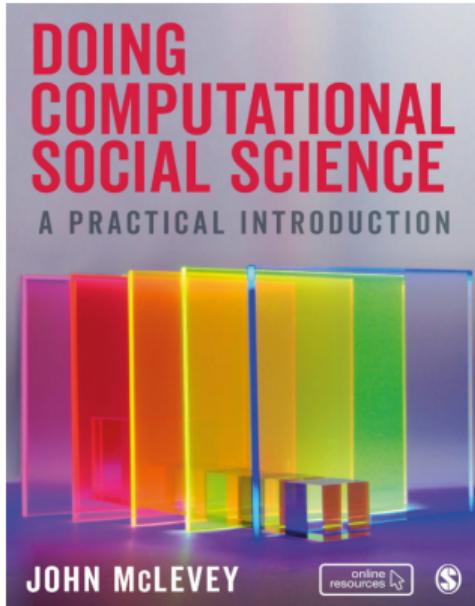
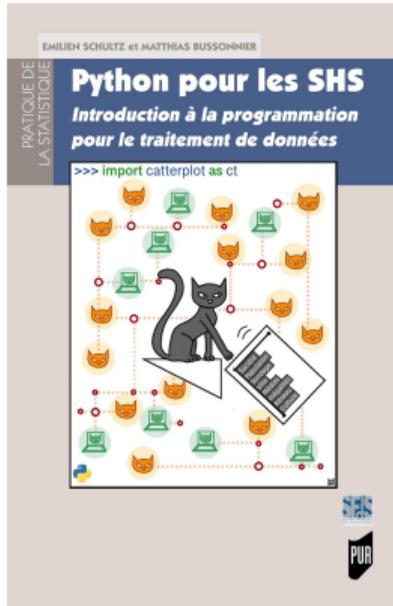
1. Développer l'espace des possibles (**cette formation**)
 - ▶ Avoir une idée des bases du langage / philosophie générale
 - ▶ Des exemples de ce qu'il est possible de faire / de la maturité des outils
2. Identifier un usage pertinent pour soi
 - ▶ Développer sa pratique en autonomie / avec un cours renforcé
 - ▶ Construire de manière itérative sa pratique
3. Améliorer sa pratique
 - ▶ Ajouter les bonnes pratiques de code / partage
 - ▶ Renforcer les aspects "théoriques" et "esthétiques"

Important de valoriser les petites victoires



Ressources pour être autonome

Prolifération de manuels et d'exemples : comment choisir ?

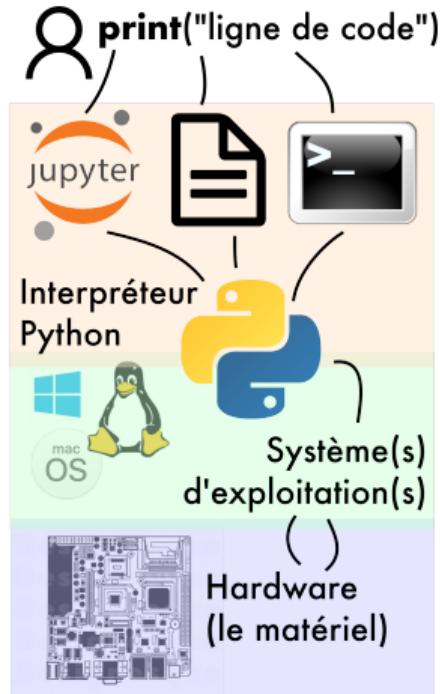


- ▶ Une liste de ressources : <https://github.com/pyshs/ressources-pyshs>
- ▶ Le cours de Melanie Walsh (EN) : <https://melaniewalsh.github.io/Intro-Cultural-Analytics/welcome.html>

Où écrire son code

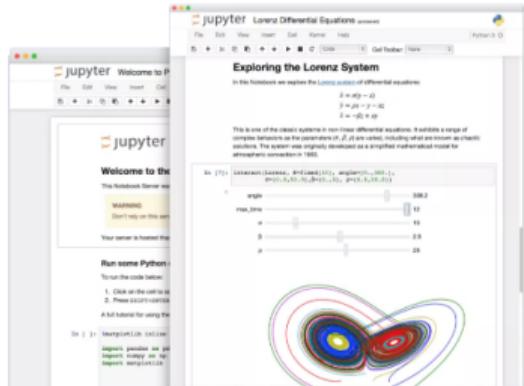
Trois manières d'exécuter un script :

- ▶ Dans un fichier texte (+ Integrated (I)DE)
- ▶ Dans le "logiciel" Python (interactivité)
- ▶ Dans un Notebook (Interactive (I)DE)



Notre choix : le Notebook Jupyter

Une philosophie générale : la programmation lettrée (*literate computing*).



Jupyter Notebook: The Classic Notebook Interface

The Jupyter Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience.

[Try it in your browser](#)

[Install the Notebook](#)



Language of choice

Jupyter supports over 40 programming languages, including Python, R, Julia, and Scala.



Share notebooks

Notebooks can be shared with others using email, Dropbox, GitHub and the [Jupyter Notebook Viewer](#).



Interactive output

Your code can produce rich, interactive output: HTML, images, videos, LaTeX, and custom MIME types.



Big data integration

Leverage big data tools, such as Apache Spark, from Python, R, and Scala. Explore that same data with pandas, scikit-learn, ggplot2, and TensorFlow.

Une philosophie générale de l'usage de la programmation en recherche

D'une perspective ordinateur Read-Eval-Print loop (REPL) vers une perspective de l'utilisateur Write-Eval-Think-Loop (WETL) : "The overarching idea of Jupyter is that humans matter."

Granger, Brian E., and Fernando Perez. 2021. "Jupyter : Thinking and Storytelling with Code and Data." *Computing in Science and Engineering* 23 (2) : 7–14.
<https://doi.org/10.1109/MCSE.2021.3059263>.

Abstract : Project Jupyter is an open-source project for interactive computing widely used in data science, machine learning, and scientific computing. We argue that even though Jupyter helps users perform complex, technical work, Jupyter itself solves problems that are fundamentally human in nature. Namely, Jupyter helps humans to think and tell stories with code and data. We illustrate this by describing three dimensions of Jupyter : 1) interactive computing ; 2) computational narratives ; and 3) the idea that Jupyter is more than software. We illustrate the impact of these dimensions on a community of practice in earth and climate science.

IDE : du Notebook au Lab à autre

- ▶ I pour interactive
- ▶ I pour integrated

The screenshot shows the official Visual Studio Code website. At the top, there's a dark navigation bar with the Visual Studio Code logo, "Visual Studio Code", and links for "Docs", "Updates", "Blog", "API", "Extensions", "FAQ", and "Learn". To the right is a search icon. Below the bar, a banner announces "Version 1.78 is now available! Read about the new features and fixes from April." The main content area has a dark background. On the left, there's a sidebar with links: "Overview", "SETUP", "GET STARTED", "USER GUIDE", "SOURCE CONTROL", "TERMINAL", "LANGUAGES", and "NODEJS /". The main title "Jupyter Notebooks in VS Code" is centered above a detailed description. To the right of the title is a small edit icon. The description explains what Jupyter Notebooks are and how VS Code supports them, listing features like creating notebooks, working with code cells, and using the Variable Explorer.

Jupyter Notebooks in VS Code

[Jupyter](#) (formerly IPython Notebook) is an open-source project that lets you easily combine Markdown text and executable Python source code on one canvas called a **notebook**. Visual Studio Code supports working with Jupyter Notebooks natively, and through [Python code files](#). This topic covers the native support available for Jupyter Notebooks and demonstrates how to:

- Create, open, and save Jupyter Notebooks
- Work with Jupyter code cells
- View, inspect, and filter variables using the Variable Explorer and Data Viewer
- Connect to a remote Jupyter server

A avoir en tête

- ▶ Des avantages
 - ▶ Ludique et interactif
 - ▶ Avoir tous les éléments au même endroit
 - ▶ Partager son script
 - ▶ Très utilisé : "Ten computer codes that transformed science" (Nature, 2021)
<https://www.nature.com/articles/d41586-021-00075-2>
- ▶ Quelques limites
 - ▶ Orde d'exécution des cellules
 - ▶ Vite confus

Si vous voulez des critiques : I don't like notebooks.- Joel Grus

<https://www.youtube.com/watch?v=7jiPeIFXb6U>

Un environnement intégré



Individual Edition is now

ANACONDA DISTRIBUTION

The world's most popular open-source Python distribution platform

Anaconda Distribution

[Download](#)

For MacOS
Python 3.9 • 64-Bit Graphical Installer • 515 MB

Get Additional Installers

A light blue rectangular box containing download links for different operating systems.

An icon of an open book with a green outline.

Open Source

Access the open-source software you need for projects in any field, from data visualization to robotics.

An icon showing three stylized human figures in a circle.

User-friendly

With our intuitive platform, you can easily search and install packages and create, load, and switch between environments.

An icon of a green checkmark inside a circular cloud-like shape.

Trusted

Our securely hosted packages and artifacts are methodically tested and regularly updated.

Exécutons un script déjà écrit !

Script "basique" qui demande une entrée textuelle et compte le nombre de mots de plus d'un certain nombre de lettres