

**Quels usages pour les Jupyter Notebooks en Sciences Sociales ?**

Inconvénients, avantages et bonnes pratiques pour les Jupyter Notebooks.

Petit panorama d’un outil en pleine expansion à destination des acteurs de l’Enseignement Supérieur et de la Recherche en Sciences Humaines et sociales.

Par POPINEAU Maxime, en mai 2023.

Introduction :

En 2014, le chercheur en physique Fernando Pérez, annonce un projet issu de son environnement interactif iPython facilitant la programmation pour les scientifiques : le projet Jupyter. Ce dernier se développe autour d’un écosystème de logiciels interactifs pour la programmation comme : le Jupyter Hub, le JupyterLab ou le Jupyter Notebook.

Un Jupyter Notebook est un document manipulable dans le navigateur internet composé de plusieurs cellules, combinant à la fois du code, du texte et des visualisations graphiques, le code pouvant être modifié et exécuté interactivement. Venu de la communauté Python, intégrant R[[1]](#footnote-1) ou Julia[[2]](#footnote-2), les Jupyter Notebooks supportent plus de 40 langages de programmation. Les Jupyter Notebooks et notamment le langage informatique Python[[3]](#footnote-3) commencent à être très utilisés dans de nombreux domaines scientifiques mais encore peu en SHS.

Ce support est en mesure de faciliter la pratique computationnelle des chercheurs en sciences humaines et sociales ; en favorisant un standard permettant d’homogénéiser les réflexions intégrant du code et de partager facilement leur recherche. Les Jupyter Notebooks peuvent avoir de nombreux avantages comme le fait de faciliter une approche interdisciplinaire car il est facile de réutiliser un Notebook pour le modifier selon ses besoins. Un des autres avantages des Notebooks est leur aspect versatile : ils peuvent être utiliser dans le cadre de la recherche ou bien de l’enseignement.

Bien qu’encore peu utilisés dans le domaine des sciences humaines et sociales, les Notebooks ont gagné en popularité dans le contexte de l’éducation. En France, c’est actuellement l’un des outils les plus utilisés par les enseignants en informatique en raison de leur facilité pédagogique[[4]](#footnote-4).

Cette synthèse propose de résumer la littérature sur les inconvénients et les avantages de cet outil dans le cadre de la pratique de recherche en SHS impliquant à divers degrés du traitement de données avec un langage de programmation.

Pour donner quelques exemples, les Jupyter Notebooks peuvent être utilisé dans beaucoup de disciplines des sciences sociales et humaines et pour des usages variés

* En linguistique : les Notebooks sont utiles pour faire de la linguistique computationnelle. Ce notebook[[5]](#footnote-5) propose une analyse sur la traduction d’un texte en différentes langues. L’analyse se porte sur le nombre de mots par traduction et leur divergence au niveau statistique.
* En géographie : les Notebooks peuvent être utiles pour améliorer le travail collaboratif en donnant plus de souplesse au traitement de données géolocalisées par exemple en combinant GeoPandas et CartoPy dans une interface interactive reproductible.
* En sociologie : les Notebooks sont utilisés pour explorer les données statistiques largement disponibles auprès des institutions ou des organismes de recherche participant à la science ouverte. Un notebook[[6]](#footnote-6) a été fait sur la mobilité professionnelle en partant des données de l’INSEE par exemple.

Face à la diversité des usages se pose la question de la stabilisation des bonnes pratiques et de faciliter l’adoption de cet outil pour les nouveaux arrivants. Nous allons tenter de répondre à cette question en faisant un tableau synthèse issu de la littérature scientifique qui commence à exister sur les avantages et inconvénients des Jupyter Notebooks.

**Avantages :**

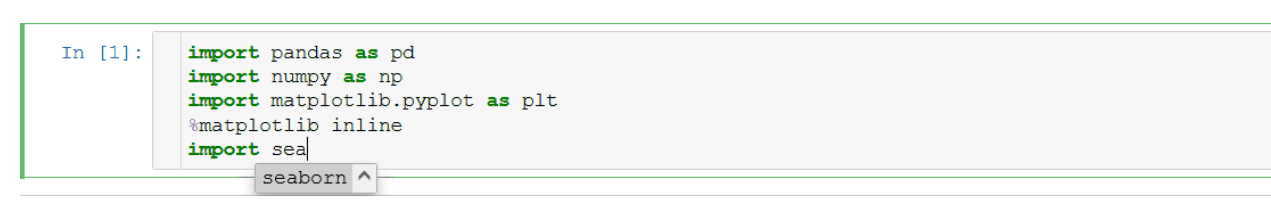
|  |  |
| --- | --- |
| Création d'un standard | * Utilisation dans toutes les branches des sciences sociales et dans des domaines différents. * 100% open source. * 40 langages de programmation * Utilisation possible sur tous types de navigateurs et tous OS (Windows, Linux, Mac). * Homogénéise les données créées * Rend les données interopérables entre les chercheurs. |
| Rend l'enseignement + interactif et facile | * Combination du code, du texte et des visualisations graphiques dans un document. * Favorise l’engagement des étudiants car ils peuvent interagir avec le Notebook en faisant des exercices. * Favorise la narration. * Mettre en place des cellules pour clarifier les différentes étapes. * Possiiblité pour les utilisateurs de changer les résultats grâces aux ipy widgets[[7]](#footnote-7) . |
| Travail réutilisable | * Faciliter d’accès à son travail et pouvoir le partager. * Conversion dans d'autres formats. * Très facile d’utiliser l’outil Github[[8]](#footnote-8) pour présenter ses recherches. * Permet à d’autres chercheurs de modifier le code pour obtenir d’autres résultats. * Permet aux étudiants d'envoyer leurs travaux en programmation à leur professeur. * Utile pour la publication d'articles. |
| Facilite les échanges entre chercheurs en SHS | * Permet à la fois de partager du texte et code dans le même document. Utile pour la publication d’articles. * Pas de but fixé par le logiciel : l'utilisateur réfléchit aux résultats qu'il veut obtenir. * Permet d’expliquer clairement son domaine de recherche à d’autres chercheurs lors d’une conférence. * Permet le travail collaboratif entre différents chercheurs à partir d'un même outil. |
| Mise en place d’un écosystème d’extensions et d’évènements. | * Traduction possible dans d’autres formats : livres, slides. * Partage simple d’un Notebook via Nbviewer[[9]](#footnote-9) et Binder[[10]](#footnote-10) permettent de visualiser le Notebook. * Communauté d’utilisateurs comme dans la recherche qui construit des extensions, des applications et qui aident les nouveaux utilisateurs. * Mis en place d’évènements communs (Jupyter Community Workshops[[11]](#footnote-11), JupyterDays[[12]](#footnote-12) et JupyterCon[[13]](#footnote-13)). |

**Inconvénients :**

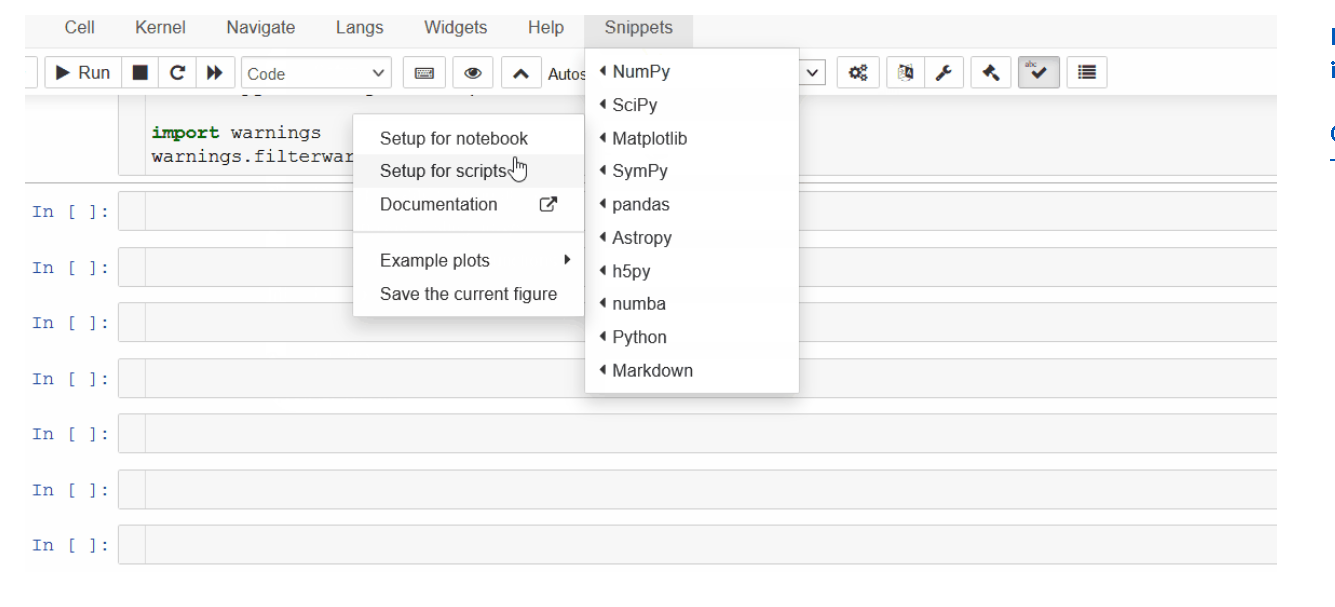
|  |  |
| --- | --- |
| Nécessite des connaissances de base. | * Nécessite de connaître un langage informatique (Python ou autre). * Une philosophie qui n’est pas nécessairement intuitive dans l’exécution du code. * Moins de support que les IDE[[14]](#footnote-14). Par exemple : le Notebook ne dit pas s’il y a une variable non-utilisée. * Face à cet outil très customisable : difficulté de savoir quels plugins ou extensions à exploiter. |
| Problème de reproductibilité | * Les Notebooks sont particulièrement compliqués à reproduire pour les débutants car ils nécessitent de réécrire le code (sur 1 million de Notebooks sur Github seulement 4% sont utilisables en état). * Impossibilité de reproduire le Notebook dans certains cas : le code peut être obsolète. Il faut aussi retrouver la bonne version des librairies utilisées. * Le code peut être exécuté de manière non linéaire, provoquant des « hidden states ». * Lien entre les données et le Notebook pas évident : l’accès aux fichiers est une des causes communes au fait que les Notebooks ne soient pas reproductibles. Il faut changer le path du fichier directement dans le code du Notebook. |
| Outil qui conduit à prendre de mauvaises habitudes pour écrire du code | * Compliqué de passer de Visual Studio Code à Jupyter Notebook pour un débutant. * Faible pratique des tests[[15]](#footnote-15) : il est compliqué de faire des tests pour un Jupyter Notebook. * Favorise des pratiques discutables dans le nom des fichiers et dans la version et la modularisation du code. * Pas de règles homogènes sur l’écriture des notebooks. Par exemple : moins de commentaires à la fin des Notebooks : la plupart des Notebooks ont une cellule d’introduction mais presque aucun a de cellules de conclusions. |
| Une diversité de solutions concurrentes de carnets interactifs ne facilitant pas le choix. | * Les différents IDE (PyCharm, Spyder, Visual Studio Code) permettent de contrôler le code avec des linters[[16]](#footnote-16),de faire du contrôle de version ainsi que faire du debugging[[17]](#footnote-17). * Les outils « cloud based »[[18]](#footnote-18) comme Google Collab et Azure Notebook permettent de collaborer avec d’autres utilisateurs et de contrôler la version[[19]](#footnote-19). * Diversité de formats similaires et difficulté pour choisir : des Notebooks interactifs comme Apache Zeppelln[[20]](#footnote-20) ou BeakerX[[21]](#footnote-21). * Deepnote[[22]](#footnote-22) permet de collaborer dans le même notebook alors que dans les Jupyter Notebooks sont stockés en local dans un ordinateur. Deepnote prend en charge le management des versions et permet de mettre en place des permissions différentes en fonction des utilisateurs du Notebook. |

La plupart des problèmes adressés peuvent être résolu grâce à des extensions, des logiciels à utiliser en complément de Jupyter Notebook :

* Une des critiques adressées est le fait qu’il n’y ait pas d’auto complétion du code ce qui peut poser des problèmes à des utilisateurs débutants : l’extension Hinterland permet d’activer l’auto complétion du code dans les Jupyter Notebooks.



* Une autre critique est le fait que les Jupyter Notebooks ne sont pas faciles à utiliser pour des utilisateurs débutants. Le « Snippets » menu permet d’obtenir du code prêt à être utiliser avec différentes librairies et pour différents usages. Cela permet à des utilisateurs débutants de ne pas avoir à regarder la documentation pour avoir la syntaxe du code.



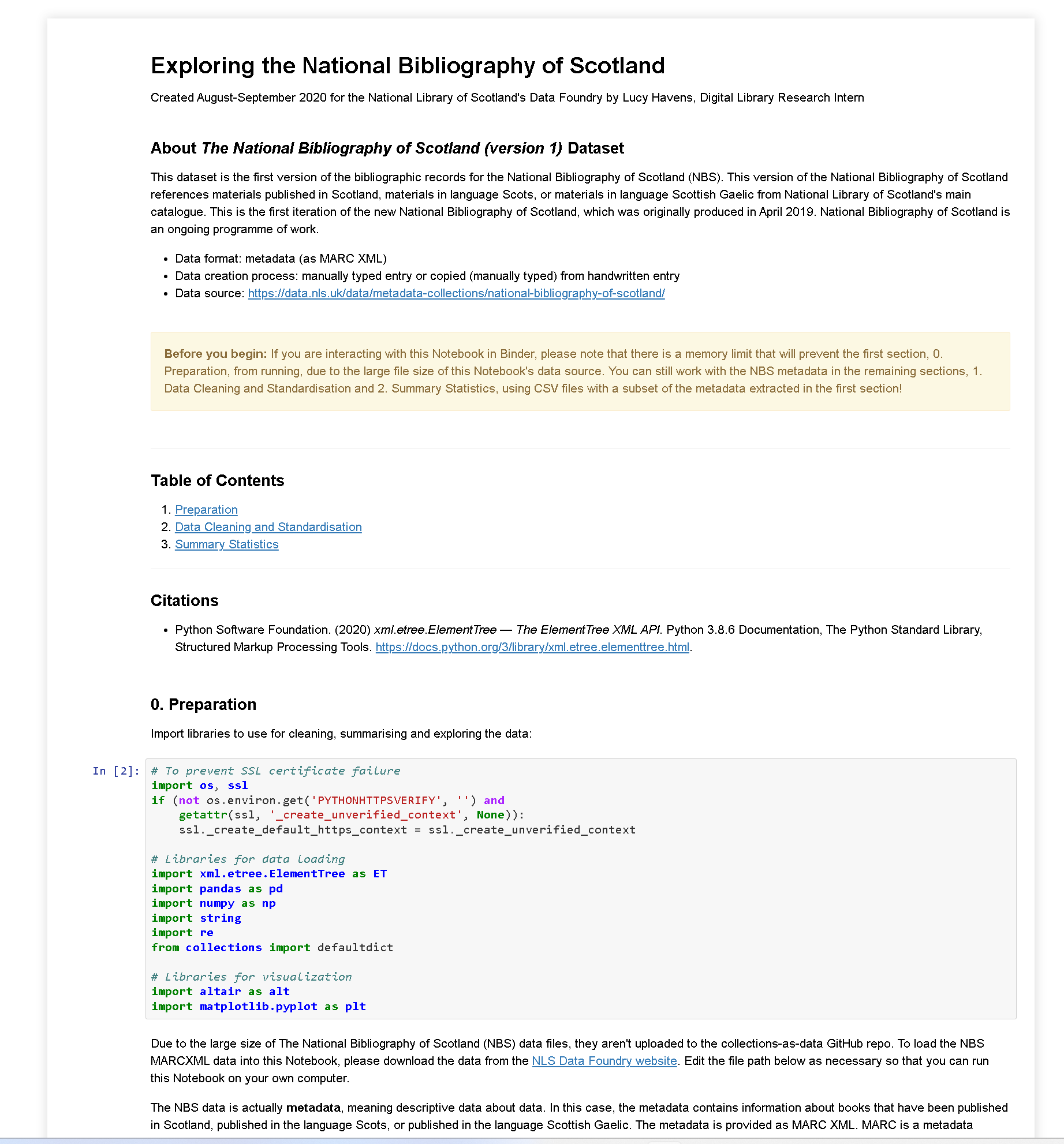
Prérequis : Malgré le fait qu’il y ait beaucoup d’extensions qui facilitent l’usage des Jupyter Notebooks, il existe des connaissances minimales à avoir pour les utiliser :

* Avoir des connaissances en codage Python sur un IDE (comme Visual Studio Code) préférablement avant de passer à Jupyter Notebook.
* Savoir qu’il faut changer les chemins des fichiers pour pouvoir réutiliser le code.
* Installer Anaconda sur son ordinateur pour pouvoir utiliser Jupyter Notebook et d’autres outils.

La littérature permet d’identifier quelques bonnes pratiques à avoir pour construire et partager un notebook[[23]](#footnote-23) :

* L’importance est de raconter à l’audience une histoire et cela nécessite une explication des différentes étapes pour essayer de résoudre la question de recherche.
* Il est important de documenter tout le processus et pas seulement les résultats.
* Utiliser les cellules pour clarifier les différentes étapes (ne pas utiliser des longues cellules avec plus de 100 lignes).
* Utiliser la modularité : il est important d’éviter de dupliquer le code. Dans les Notebooks, il est facile de copier/coller du code et de changer quelques lignes. Cela rend le Notebook très compliqué à lire et impossible à maintenir. Il est préférable de mettre le code dans une fonction et d’appeler cette fonction dans différentes cellules.
* Faire un dépôt Github avec un fichier READme et un fichier requirement avec les différentes versions des libraires.
* Mettre en place une licence open source pour que le travail soit réutilisable.
* Bien organiser l’architecture du dossier du projet, notamment les données et les documents intermédiaires.
* Utilisation du contrôle de version : dû à l’aspect interactif du Notebook il est facile de changer accidentellement du code.
* Faire un « pipeline » : créer un modèle : le but est de faire un Notebook qui soit réutilisable en changeant les données et les paramètres.
* Faire que le Notebook soit le plus lisible possible avec une présentation claire :

**Exemple d’un Notebook de qualité ci-dessous :**



Ce Notebook est de très bonne qualité. Sont présents le titre, la date de création et les auteurs. Il contient une présentation des données utilisées ainsi qu’une table des matières. L’article qui discute de ce code est référencé. La démarche est très claire : un titre est présent, le code a des commentaires et les cellules au format texte permettent de suivre la réflexion. Le Notebook présenté est disponible au format Web ce qui est utile pour l’utilisateur n’ayant pas installé Anaconda sur son ordinateur.

* Exporter le Notebook en fichier PDF et mettre ce fichier PDF dans le Github. Cela permet à des personnes n’ayant pas installé Anaconda de voir votre travail.

Conclusion :

La programmation en science, notamment avec Python, et les nouveaux outils comme Jupyter Notebook sont importants pour les sciences humaines et sociales. Une partie des critiques adressées aux Notebooks ne sont pas dus au « médium » mais plutôt à la manière dont les personnes s’en servent. La plupart des problèmes adressés dans cet article ont des solutions : les Jupyter Notebooks sont très customisables selon les besoins de ses utilisateurs. Les Jupyter Notebooks peuvent être une bonne solution dans les situations suivantes :

* Dans le cadre d’une conférence pour montrer et expliquer les résultats de recherche produits. Le Notebook facilite l’interaction : il permet de faire une démonstration très facilement, ce qui en dynamise la présentation.
* Dans le cadre d’un travail collaboratif de recherche entre des chercheurs connaissant l’outil et Python. Il est plus facile de travailler sur un outil en commun plutôt qu’avec des fichiers utilisant des logiciels différents.
* Dans le cadre d’un enseignement sur l’informatique. Les Notebooks permettent de mettre en place des exercices qui peuvent être ensuite fait par les étudiants. Les Notebooks sont très facilement modifiables : les étudiants peuvent comprendre la modification dans le code et voir comment les résultats évoluent.

Les Jupyter Notebooks ne mettent pas la machine au cœur du processus de recherche mais bien l’humain : l’environnement favorise l’interaction et l’exploration. L’interactivité des Notebooks permet aux utilisateurs d’exécuter, de modifier, de documenter, d’explorer et de collaborer de manière efficace dans un même environnement de développement intégré ce qui en fait un très bon outil pour les SHS.

La communauté des Jupyter Notebook est ancrée largement dans la communauté scientifique : le projet Jupyter a été créé par des chercheurs et pour des chercheurs. Ce faisant, des valeurs similaires sont partagées autour de l’open source, de l’aide des nouveaux arrivants, de l’exploration des nouvelles solutions et de la reproductibilité. Les Notebooks ne sont pas qu’un outil mais aussi et principalement une communauté dynamique qui crée de nouvelles solutions en permanence adaptées à la recherche.

Avec le développement des nouvelles technologies dans le champ de la recherche, il me semble que l’évolution de la recherche et de l’enseignement en SHS se fera avec la programmation et les Notebooks interactifs notamment le projet Jupyter.

Nous n’attendons plus que vous !

**Liste de ressources utiles pour commencer :**

https://www.python.org/downloads/

<https://code.visualstudio.com/>

La première étape est de choisir un langage informatique : Python est le choix de prédilection pour tout débutant en programmation. Le téléchargement de Python est disponible sur le site officiel. Le deuxième lien est pour télécharger Visual Studio Code qui permet d’avoir un éditeur de texte qui propose des fonctionnalités très utiles pour les débutants mais aussi pour les utilisateurs avancés.

https://www.freecodecamp.org/

Le site FreeCodeCamp met en ligne des ressources qui permettent d’apprendre à coder gratuitement. Leur chaîne youtube propose aussi des vidéos notamment une sur python de très bonne qualité disponible à cette adresse :

<https://www.youtube.com/watch?v=rfscVS0vtbw>

https://programminghistorian.org/fr/

Le site Programming Historian propose des leçons en ligne pour aider les chercheurs en sciences humaines et sociales à acquérir des compétences en programmation. Les leçons sont disponibles en anglais, en espagnol mais aussi en français.

<https://jupyter.org/>

Après avoir appris à programmer, nous pouvons utiliser les Jupyter Notebooks ainsi que les nombreuses extensions pour pouvoir partager notre code et aussi pour pouvoir modifier d’autres travaux.

<https://stackoverflow.com/>

Le site Stack Overflow est un forum dédié à la communauté des développeurs qui permet aux utilisateurs de poser des questions et d’obtenir des réponses à des problèmes liés à la programmation.

1. R est un langage de programmation (comme Python) qui est utilisé dans le cadre des statistiques et de la science des données. [↑](#footnote-ref-1)
2. Julia est un langage de programmation très performant qui est utilisé pour le calcul scientifique. [↑](#footnote-ref-2)
3. Python est le langage de programmation le plus utilisé et le plus simple. Python est utilisé dans l’enseignement pour permettre une initiation aux concepts de base de la programmation mais aussi dans le cadre de la recherche pour obtenir des résultats. [↑](#footnote-ref-3)
4. Christophe Casseau, Jean-Rémy Falleri, Xavier Blanc, Thomas Degueule. Immediate Feedback for Students to Solve Notebook Reproducibility Problems in the Classroom. 2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), Oct 2021, Saint Louis, Missouri, United States. ⟨10.1109/VL/HCC51201.2021.9576363⟩. ⟨hal-03378094⟩ [↑](#footnote-ref-4)
5. https://github.com/quinnanya/litlab\_translations/blob/master/litlab\_translations\_2019-04-15\_jupyter\_notebook.ipynb [↑](#footnote-ref-5)
6. https://gitlab.huma-num.fr/io/mobilites-professionnelles-final/-/blob/main/notebooks/Mobilit%C3%A9s%20professionnelles.ipynb [↑](#footnote-ref-6)
7. Les Ipywidgets permettent l’interactivité du Notebook. L’utilisateur peut avoir une barre par exemple pour changer les valeurs et ensuite obtenir les résultats qui sont modifiés et retournés par l’algorithme. Les Ipywidgets sont très utilisés dans le cadre de l’enseignement et ils permettent aussi à des personnes qui ne savent pas programmer d’interagir avec le Notebook. [↑](#footnote-ref-7)
8. Github est une plateforme de collaboration qui permet d’héberger au format open-source des projets informatiques comportant du code. Les utilisateurs ont accès à des millions et des millions de fichiers python ou des Jupyter Notebooks sur des sujets très variés et dans des disciplines différentes. Ce service propose aussi des fonctionnalités recherchées par les programmeurs : l’intégration continue, la gestion de versions … [↑](#footnote-ref-8)
9. Nbviewer est disponible à cette adresse : <https://nbviewer.org/>. Cette solution est utilisée pour partager de manière facile les Jupyter Notebooks sans passer par l’installation de logiciels tiers. Le notebook est disponible sous forme de page HTML statique. Il est possible d’utiliser le Notebook au format HTML dans un site web en copiant/collant l’URL dans le code. Un tutoriel est disponible pour apprendre à utiliser nbviewer ici : https://www.tutorialspoint.com/jupyter/sharing\_jupyter\_notebook\_using\_github\_and\_nbviewer.htm [↑](#footnote-ref-9)
10. Binder est une extension permettant de réexécuter le code et de supprimer ou d’ajouter des cellules. Cette extension permet de rendre le Notebook interactif au contraire de nbviewer qui donne une version du Notebook statique. [↑](#footnote-ref-10)
11. La communauté Jupyter propose beaucoup d’ateliers sur des thématiques précises par exemple : un atelier Jupyter s’est tenu en décembre 2022 à Paris sur le sujet du projet JupyterLite qui permet d’avoir un Jupyter Notebook qui tourne dans le navigateur web sans avoir à faire d’autres installations. [↑](#footnote-ref-11)
12. Les JupyterDas sont comme le nom l’indique une conférence d’un jour sur le sujet des Jupyter Notebooks. [↑](#footnote-ref-12)
13. La JupyterCon est la conférence qui regroupe tous les utilisateurs des Jupyter Notebooks. L’évènement dure pendant plusieurs jours et comporte des conférences, des tables-rondes et aussi des cours. La JupyterCon permet de renforcer la communauté et d’avoir des interactions en personne pour des utilisateurs qui collabore à distance durant le reste de l’année. [↑](#footnote-ref-13)
14. Un IDE (environnement de développement intégré) est un ensemble d’outils qui permet d’augmenter la productivité des programmeurs qui créent des logiciels. Un IDE comporte un éditeur de texte permettant de faire de la programmation. Les exemples d’IDE sont nombreux : Visual Studio Code, CodeLite, NetBeans, PytCharm. [↑](#footnote-ref-14)
15. Un test est un morceau du code qui permet de vérifier que le code fonctionne comme attendu et qu’il ne produit pas de bugs. [↑](#footnote-ref-15)
16. Un Linter est un outil d’analyse de code qui permet de détecter les erreurs et les problèmes de syntaxe. [↑](#footnote-ref-16)
17. Le « debugging » est une pratique qui consiste à trouver et à corriger les erreurs dans le code. Les programmeurs étudient le code pour déterminer la raison des erreurs. Le but est d’exécuter le code et de le vérifier étapes par étapes pour résoudre le problème. Les tests aident pour le « debugging ». [↑](#footnote-ref-17)
18. « Cloud based » signifie que les données ne sont pas sur un serveur local ou sur un ordinateur personnel mais que les données sont stockées sur des serveurs informatiques à distance puis ensuite hébergés sur internet. [↑](#footnote-ref-18)
19. Le « versioning » permet d’avoir la version des librairies utilisées ce qui est utile pour pouvoir reproduire le Notebook. Certaines fonctionnalités ne sont que disponibles qu’avec une version spécifique de la librairie. [↑](#footnote-ref-19)
20. Apache Zepellin est un projet similaire aux Jupyter Notebooks. Apache Zepellin est un projet permettant d’analyser et de mettre en forme, de manière visuelle et interactive, de gros volumes de données traités via le framework de calcul distribué Spark. (source : <https://fr.wikipedia.org/wiki/Apache_Zeppelin>) [↑](#footnote-ref-20)
21. BeakerX est une extension open source du projet Jupyter Notebook visant à améliorer la visualisation graphique et l’interactivité des Notebooks. [↑](#footnote-ref-21)
22. DeepNote est un Notebook collaboratif qui cherche donc à rendre la collaboration entre plusieurs utilisateurs plus facile. [↑](#footnote-ref-22)
23. Rule, A., Birmingham, A., Zuniga, C., Altintas, I., Huang, S.-C., Knight, R., … Rose, P. W. (2019). Ten simple rules for writing and sharing computational analyses in Jupyter Notebooks. PLOS Computational Biology, 15(7), e1007007. doi:10.1371/journal.pcbi.1007007 [↑](#footnote-ref-23)