Notebook

Document computationnel

04 décembre 2023

Table des matières

1. Késaco ?

1.1. Définition

Un notebook est une **interface de programmation** qui permet de combiner des sections en **langage naturel** et des sections en **langage informatique**.

Il s'agit de l'outil idéal pour appliquer le paradigme de la **programmation lettrée**.

Si vous voulez voir ou revoir le module sur la programmation lettrée, cliquez ici.

1.2. De nombreuses dénominations

En raison de la diversité des outils logiciels disponibles dans ce domaine, il existe de **nombreuses dénominations pour les notebooks**.

- **Article exécutable**
- **Document computationnel**
- Document électronique interactif
- Cahier de programmation
- Cahier électronique d’analyse
- Calepin électronique
- Carnet de code
- Bloc-code
- Manifeste algorithmique
- …

Cela reflète la diversité de leurs caractéristiques, de leurs fonctionnalités et des approches différentes de conception.

C'est pour cela que les **noms propres des différents dispositifs sont souvent utilisés** (carnet Jupyter, carnet Observable, Rmarkdown, Quarto…).

2. Une histoire de reproductibilité

2.1. À la recherche de la reproductibilité

Dans les années 70, plusieurs scientifiques s'inquiètent de l'impact des évolutions technologiques et des capacités de calcul sur la transparence et la reproductibilité des travaux scientifiques. De ce constat émerge le concept de **recherche reproductible**.

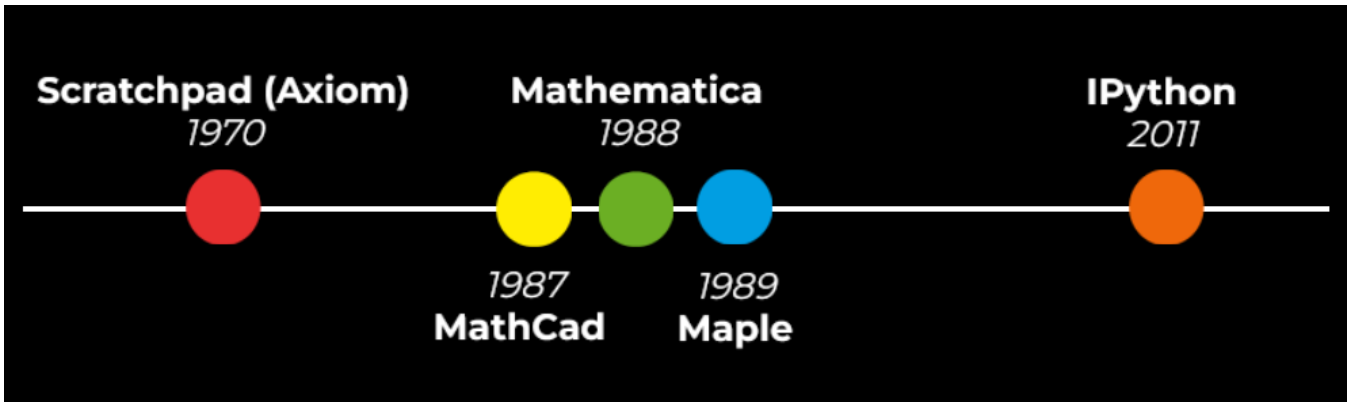
+info – La recherche reproductible est une démarche qui consiste à fournir l'ensemble des informations (texte, données, code de programmation accompagnés d'une description algorithmique) utilisées et appliquées pour obtenir des résultats présentés dans le cadre d'un travail scientifique.

À l'image de **Jon F. Clearbout**, plusieurs chercheurs vont participer au développement d'interfaces de programmation permettant d'assembler du texte mis en forme et du code exécutable, pour une meilleure intelligibilité des scripts et la reproductibilité des analyses présentées.

+info – Jon F. Claerbout
Géophysicien et sismologue américain né en 1938. Professeur émérite du département de Géophysique de l'Université de Stanford.
Il est l'un des premiers chercheurs à exprimer le fait que les méthodes de calcul menacent la reproductibilité de la recherche si l'on ne donne pas un accès libre aux données et aux logiciels qui les sous-tendent. Il semble être à l'origine de la première apparition de l'expression "recherche reproductible" dans une publication scientifique (Claerbout & Karrenbach, 1992).
En 1976, il publie un ouvrage devenu une référence en traitement de signal : « Fundamentals of Geophysical Data Processing ». Cette 1ère version - non-reproductible – l'amènera à s'intéresser à la reproductibilité et aux notebooks.
Voir sa fiche sur Wikipédia

Les **notebooks** ou **documents computationnels** sont nés !

2.2. Les premiers notebooks



Frise chronologique présentant quelques-uns des premiers notebooks :

- 1970 : Scratchpad (Axiom)
- 1987 : MathCad
- 1988 : Mathematica
- 1989 : Maple
- 2011 : IPython

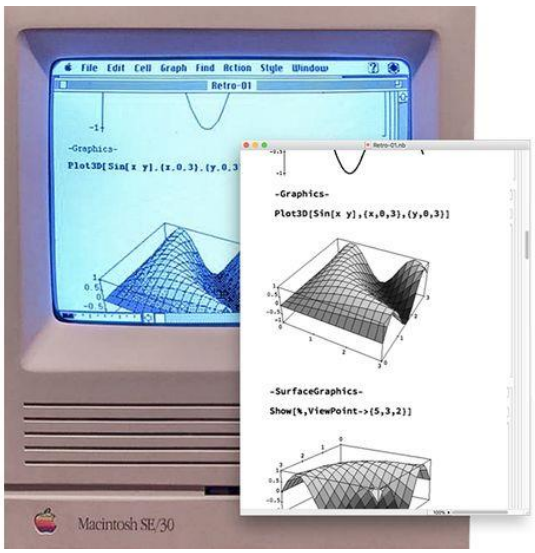


Illustration de Mathematica 1.0

Source : https://blog.wolfram.com/2018/06/21/weve-come-a-long-way-in-30-years-but-you-havent-seen-anything-yet/

Au fil du temps, plusieurs systèmes de notebooks ont été développés.
Il en existe aujourd'hui plusieurs dizaines, qui offrent différentes fonctionnalités et sont adaptés à divers langages de programmation.

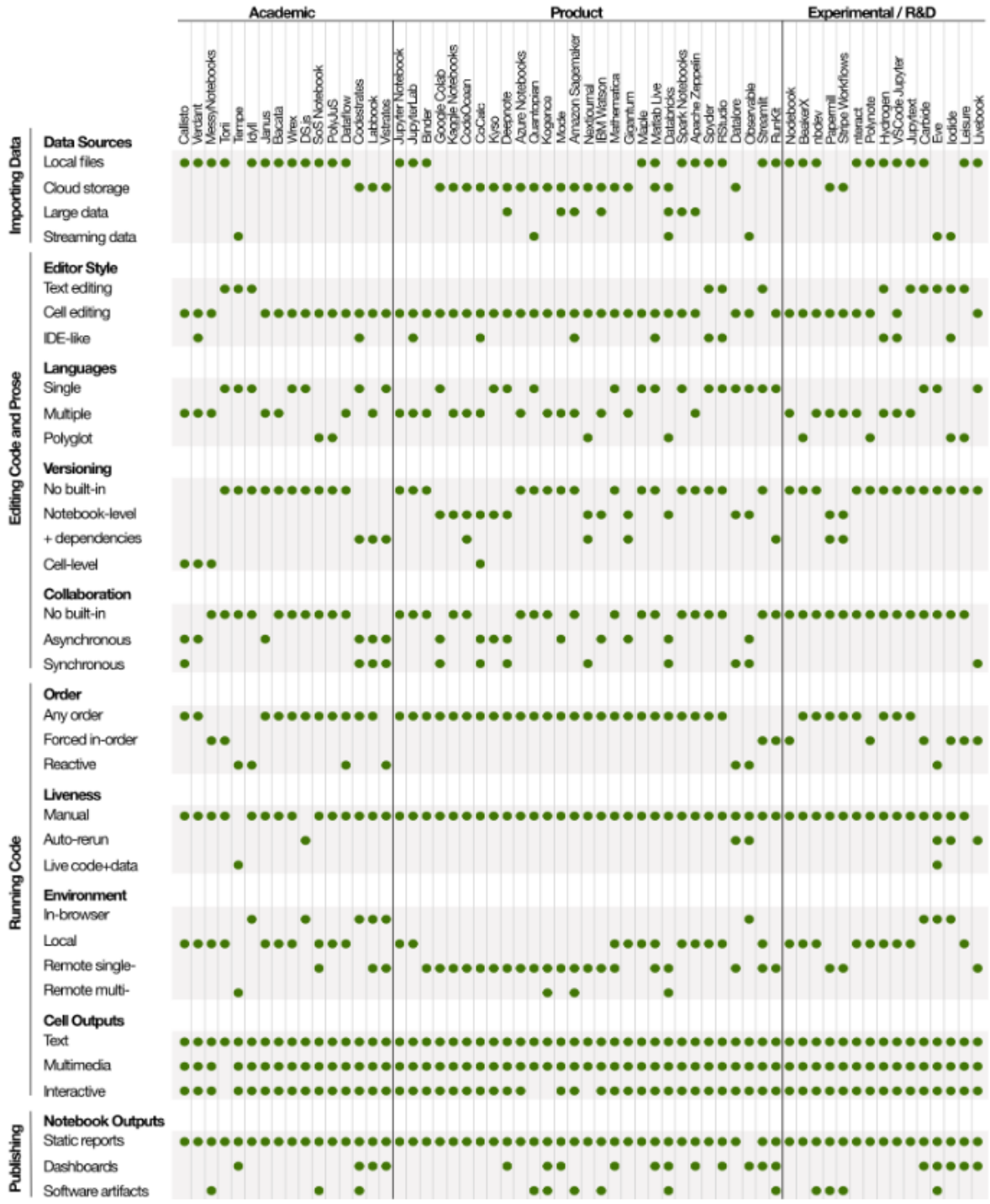


TABLE I

HOW THE 60 NOTEBOOK SYSTEMS IN OUR STUDY (COLUMNS) FIT INTO THE DESIGN SPACE DIMENSIONS (ROWS) WE ILLUSTRATED IN FIGURE 1.

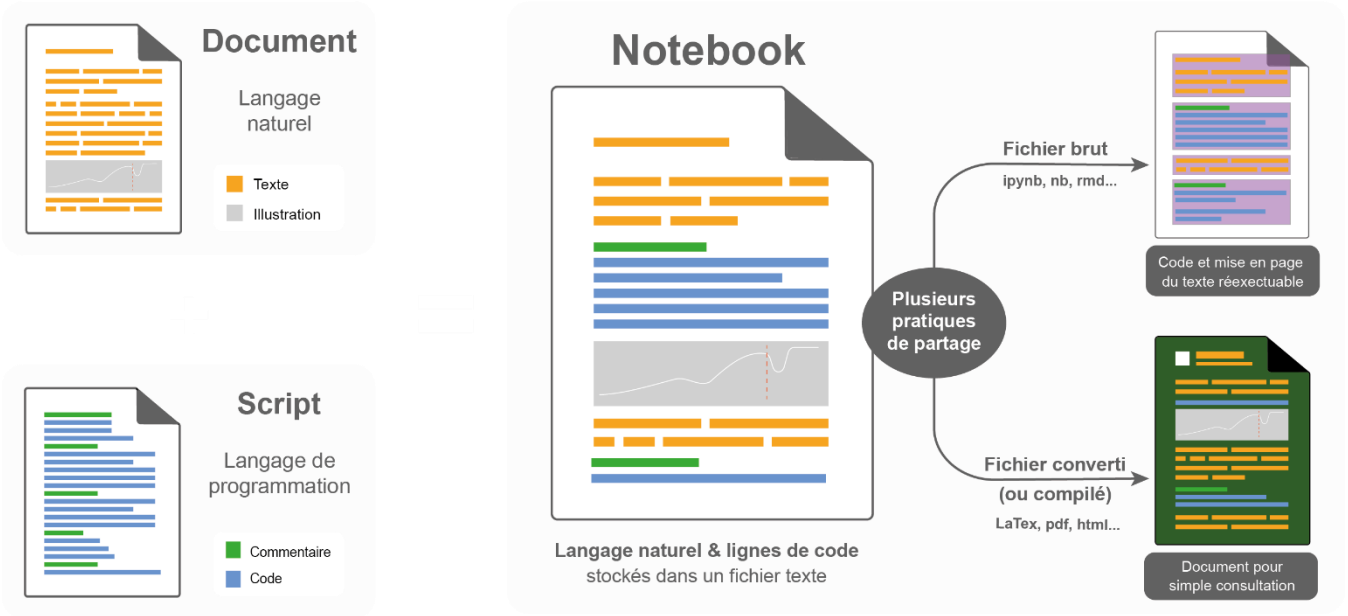
Source : S. Lau, I. Drosos, J. M. Markel and P. J. Guo, “The Design Space of Computational Notebooks: An Analysis of 60 Systems in Academia and Industry,” 2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), 2020

Tableau à la page 5.

https://pg.ucsd.edu/publications/computational-notebooks-design-space_VLHCC-2020.pdf

3. Anatomie d'un *notebook*

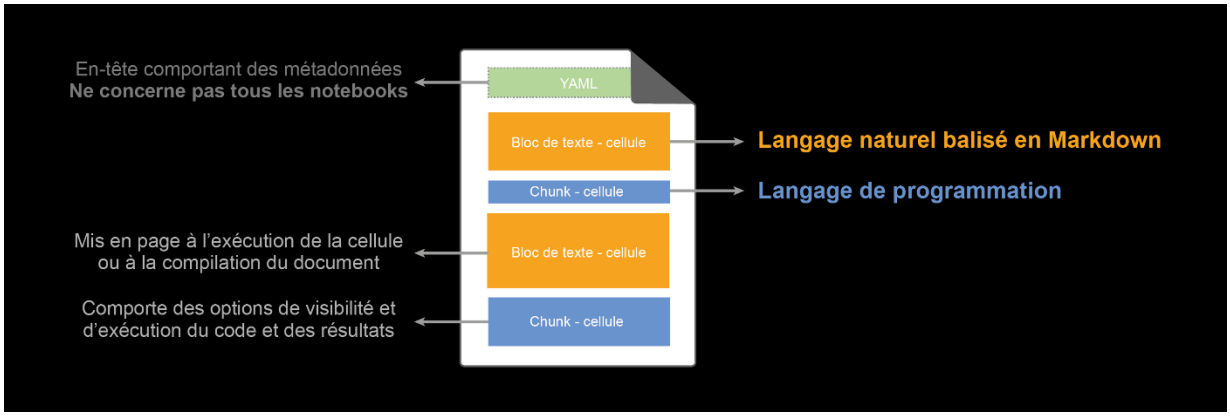
3.1. Du texte & du code dans un seul fichier texte



Selon le cas d'usage, un notebook se partage de différentes manières :

- Partage du fichier source : un simple fichier texte, dans lequel se trouve tout le code source (langage naturel et langage de programmation) prêt à être compilé en document mis en page.
- Partage du document généré : le texte est mis en forme, le code a été exécuté (résultats visibles) et l'ensemble du document est mis en page. Les formats possibles sont multiples : PDF, HTML, Word…
- Partage du code source + document généré

3.2. Les 3 composants d'un notebook



+info - En-Tête structurée en langage YAML

L'en-tête YAML (Yet Another Markup Language), contient des variables prises en compte à l'exécution du fichier, comme le titre, l'auteur, la date... Des variables permettant de configurer la mise en page (template, TOC*, numérotation, CSS, Biblio…) et le format de sortie du notebook peuvent également être renseignées dans l'en-tête du fichier. Dans le reste du document, que l'on appelle le corps, deux types de contenu peuvent être entremêlés à la convenance de l'auteur :

- Du texte en langage naturel, balisé avec le langage Markdown pour le mettre en forme
- Des blocs de code, appelés cellules ou chunks, qui contiennent un langage de programmation qui pourra être interprété à la compilation du document.

Cliquer ici pour voir la définition de YAML sur Wikipédia

*TOC = Table of content

4. Le langage Markdown

4.1. Le Markdown

Créé en 2004 par John Gruber, avec l'aide d'Aaron Swartz, le Markdown est le langage de balisage le plus léger au monde. **Il offre une syntaxe facile à lire et à écrire.**
« Un document balisé par Markdown peut être lu en l'état sans donner l'impression d'avoir été balisé ou formaté par des instructions particulières. » John Gruber, 2004

Un document balisé en Markdown peut être converti dans de nombreux formats : pdf, html, word, ePub, odt…

4.2. Exemple de balisage en Markdown

Le markdown **permet de formater du texte pour le mettre en forme**, comme le langage html le permet pour une page web.

Exemple de balises Markdown (en rouge) :

```
# Titre de niveau 1
## Titre de niveau 2

Texte en italique : *italique*
Texte en gras  : **gras**
Texte en gras-italique : ***gras-italique***

Lien cliquable : [cliquez ici](https://view.genial.ly
/63fde1e28e6ae700122e6a52)
```

Résultat graphique :

Pour voir l'exemple complet, cliquer ici.

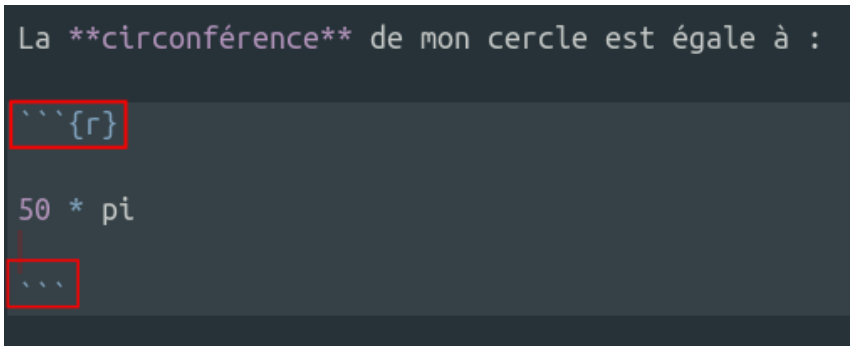
5. Les blocs de code

5.1. Insérer du code dans un notebook

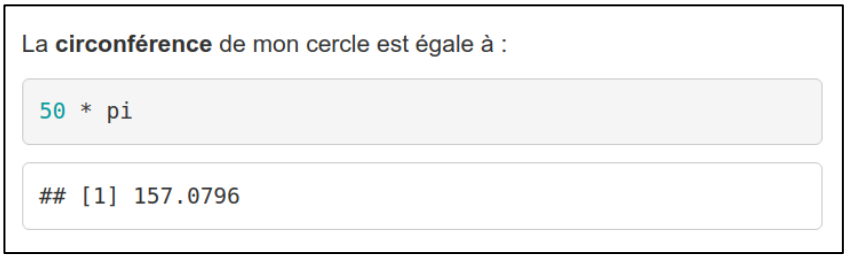
La manière d'insérer du code dans un document computationnel est propre à chaque système, mais le procédé est toujours le même. **Les blocs de code (ou cellule, chunk...) sont délimités par un ou plusieurs caractères et le langage utilisé est indiqué en début de bloc.**

Exemple avec le système Rmarkdown, où les blocs de code (*chunks*) sont délimités avec 3 guillemets inversés. Le langage à interpréter est ensuite indiqué entre accolade en début de *chunk*.

Code source :



Rendu graphique après compilation :



5.2. Paramétrer le comportement d'un bloc de code

Le comportement d'un bloc de code lors de son exécution est paramétrable.
Il est par exemple possible de ne pas faire exécuter le bloc de code, de ne pas afficher le code et/ou le résultat dans le document compilé.

Dans cet exemple (Rmarkdown), des variables comportementales ont été ajoutées entre les accolades en début de *chunk* :

- **eval = TRUE** indique que **le code sera exécuté** à la compilation du document
- **echo = FALSE** indique que **le code source ne sera pas affiché**.

Code source :

````
La **circonférence** de mon cercle est égale à :

```{r, eval = TRUE, echo = FALSE}

50 * pi

```
````

Rendu graphique après compilation :

La **circonférence** de mon cercle est égale à :

```
## [1] 157.0796
```

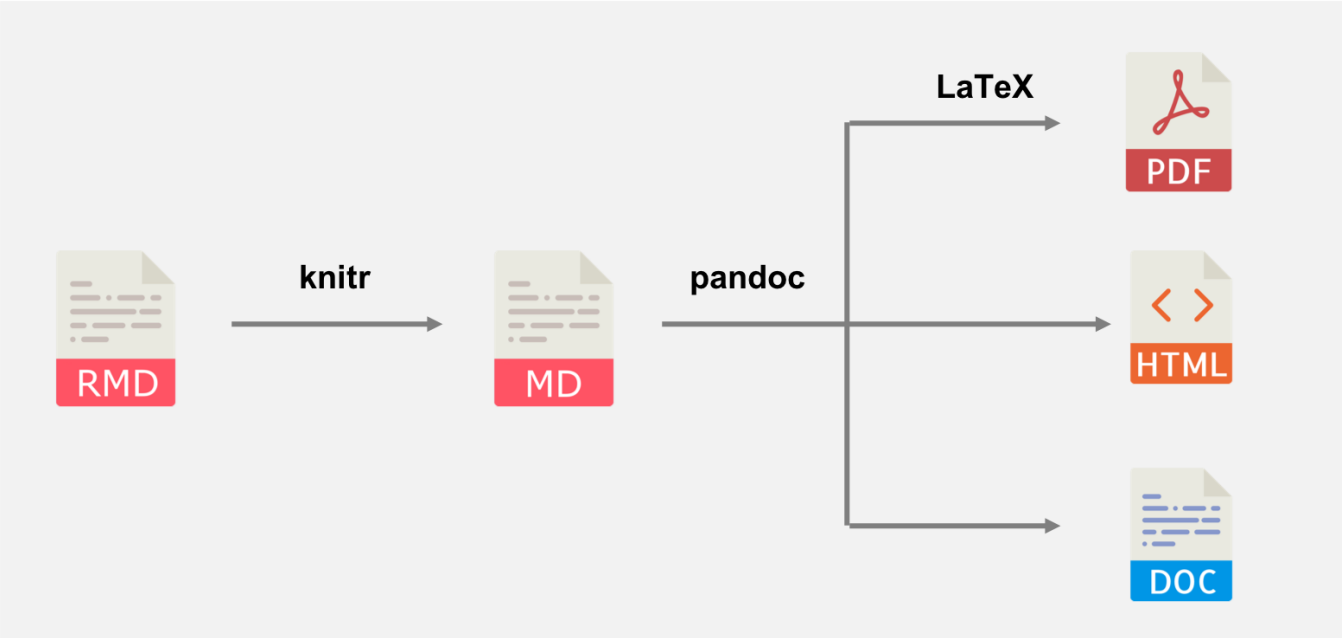
6. Exécution d'un *notebook* ?

6.1. Exécution (compilation) d'un *notebook* ?

Lorsque l'on exécute un notebook, le document est compilé en deux étapes dans le format désiré :

1. **Le code est exécuté** et les résultats générés. L'ensemble du document, incluant le code de programmation et les résultats produits sont balisés en langage markdown.
2. **Le document est mis en forme** dans le format souhaité (HTML, doc, PDF…) via le logiciel **Pandoc**.

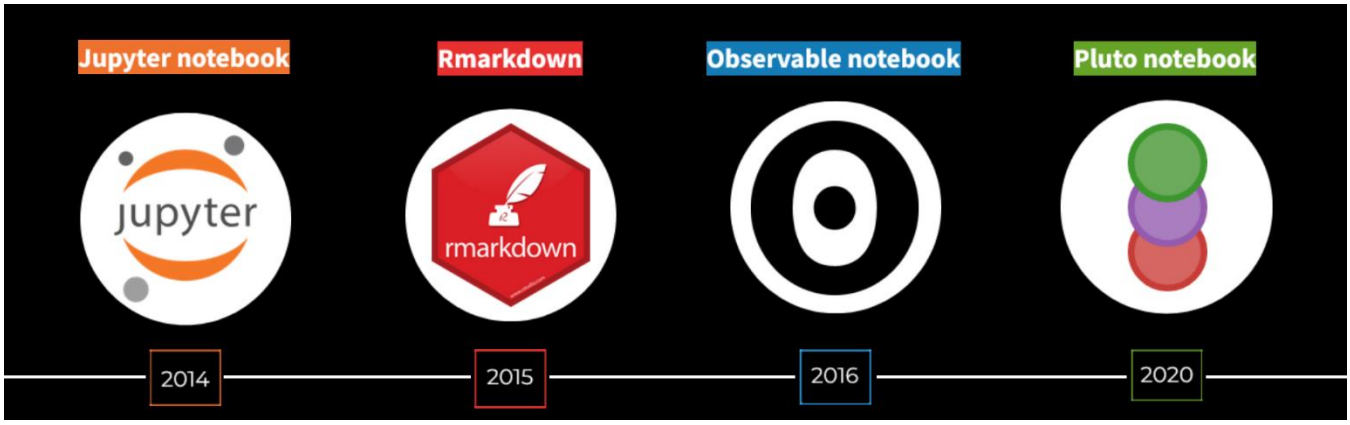
Focus Pandoc :
Pandoc est un logiciel libre de conversion de documents numériques.
Il permet notamment de convertir des documents en Markdown vers des formats **HTML, Microsoft Word, PDF**.
Voir le site de Pandoc.



Processus de compilation d'un notebook Rmarkdown (langage R) à son exécution (appelée "knit")

7. Les *notebooks* dans la recherche scientifique

7.1. Les systèmes de notebook récents



Carnet Jupyter (2014) :

Ce système de notebook est le résultat d'une extension du projet Ipython. L'apparition des carnets Jupyter est une avancée importante pour le partage et le développement interactif dans l'univers des notebooks. Il s'agit d'une technologie Open Source qui permet l'interprétation de trois langages dans un seul notebook : Julia, Python, et R. Les carnets Jupyter peuvent être utilisés via l'interface de développement intégré JupyterLab qui en facilite l'utilisation ou via JupyterHub, un serveur multi-utilisateurs qui permet d'héberger et de gérer des carnets Jupyter à plusieurs.

Rmarkdown (2015) :

Ce système de notebook Open Source est développé par l'entreprise Rstudio. Le Rmarkdown est explicitement orienté programmation lettrée, pour le domaine scientifique.

De nombreuses extensions permettent la production de différents types de documents (rapport, article scientifique, livre, tutoriel, diaporama, blog, site web…). De nombreux modèles de mise en page sont également mis à disposition.

Si nativement le Rmarkdown n'interprète que le langage R, il existe des solutions pour intégrer de nombreux langages dans un document Rmarkdown.

Pour voir le site : https://rmarkdown.rstudio.com/

Carnet Observable (2016) :

Ce système de notebook est developpé par Mike Bostock, également à l'origine de la bibliothèque graphique JavaScript de référence D3.js.

Ce notebook est spécialisé dans les représentations graphiques dynamiques en Javascript. Il s'agit d'ailleurs du seul langage interprété par les carnets Observable.

Son point fort est la réactivité et l'extrême souplesse dans l'organisation des cellules de code, où l'ordre d'apparition n'a aucune importance.

Pour voir le site : https://observablehq.com/

Carnet Pluto (2020) :

Julia est un langage récent (version 1.0 en 2018) qui a pour objectif sous-jacent de combiner le meilleur de différents langages. Ainsi, Julia possède une extension (pluto.js, Open Source) permettant la création de notebooks.

Le système de notebook Pluto est très semblable aux notebooks Jupyter. Si aujourd'hui beaucoup moins d'extensions sont disponibles comparé aux notebooks Jupyter, ce système offre une meilleure réactivité. Néanmoins, son niveau d'utilisation reflète la taille de la communauté des utilisateurs du langage Julia : en forte croissance mais à ce jour toujours moins utilisé que Python ou R.

Pour voir le site : https://help.juliahub.com/juliahub/stable/tutorials/pluto_notebooks/

7.2. La dernière génération : Quarto (2022)

Développé par l'entreprise **Posit** (anciennement Rstudio), Quarto est un **système de publication scientifique et technique open-source**. Il unifie et étend l'écosystème R Markdown existant. Il tente également de rendre la pratique des documents computationnels plus accessible.

Il s'agit sans doute du système de notebook le plus abouti, résultant des 50 années de recherche et développement des notebooks.

Focus : Concrètement…

Quarto permet de générer très facilement des documents computationnels contenant toutes les caractéristiques d'un document scientifique (équivalent LaTeX, bibliographie, TOC*, TOF*, note de bas de page…).

TOC* = Table of contents

TOF* = Table of figures

Focus 4 langages :

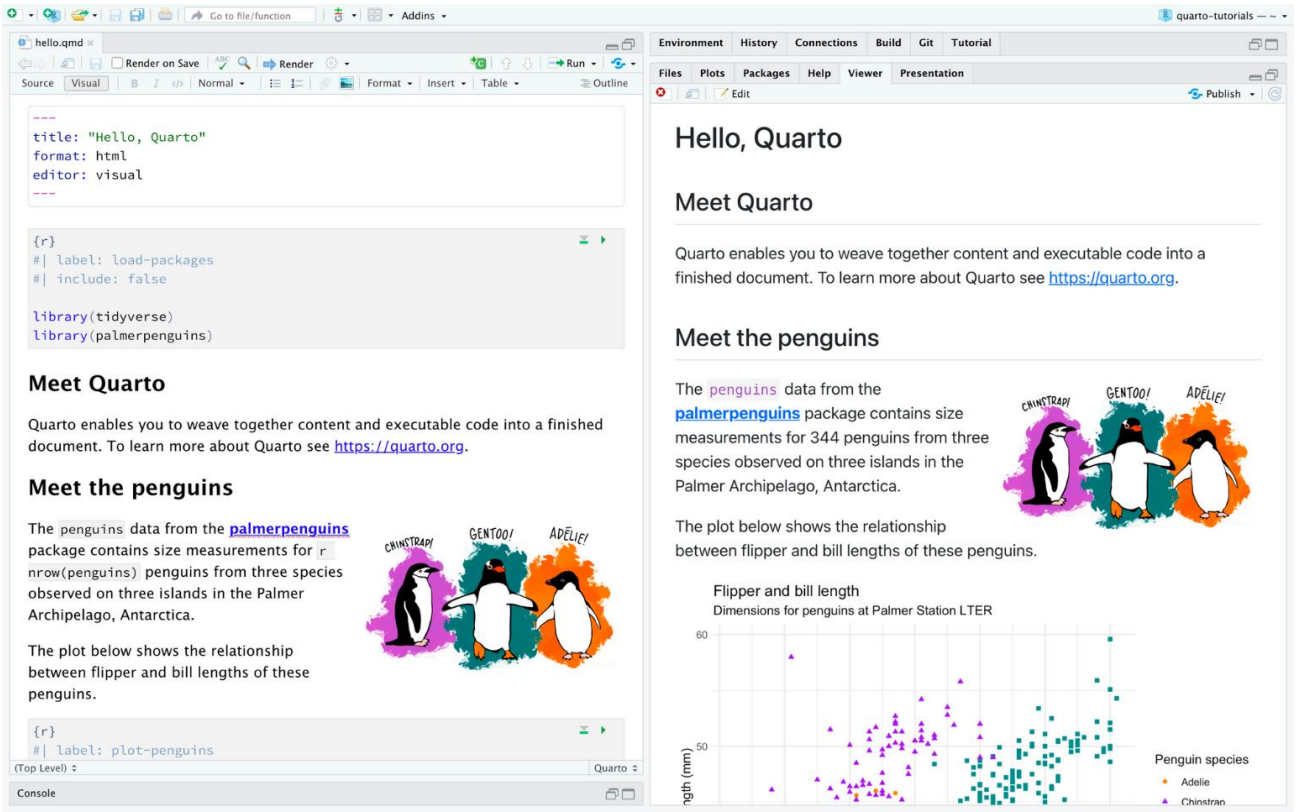
Quarto interprète nativement 4 langages :

- R,
- Python,
- Julia,
- Observable (Javascript).

Focus 4 types de publication :

Quatre types de publication sont possibles :

- article,
- livre,
- diaporama
- et site web.



Pour voir le site : https://quarto.org/

7.3. Exemples de *notebooks* Quarto

Voici un exemple pour chaque type de publication possible avec Quarto.

Exemple 1 - Article scientifique :

Inference of Multiscale Gaussian Graphical Model

Edmond Sanou, Christophe Ambroise, and Geneviève Robin

Computo, 2023

Cliquez ici pour voir la sortie HTML.

Cliquez ici pour voir la sortie PDF.

Cliquez ici pour voir le code source.

Exemple 2 – Livre et Manuel

Géomatique avec R

Timothée Giraud et Hugues Pecout

MASTER G2M, Université Paris 8 Vincennes – Saint-Denis, 2023

Cliquez ici pour voir la sortie HTML.

Cliquez ici pour voir le code source.

Exemple 3 – Diaporama

Document computationnel avec R

Hugues Pecout

Séance ElementR, 2023.

Cliquez ici pour voir la sortie HTML.

Cliquez ici pour voir le code source.

Exemple 4 – Site web

Airbnb en Île de France : Géovisualisation multi-échelles des locations Airbnb en région parisienne (2016-2022)

Louis Laurian, Ronan Ysebaert, Marianne Guérois et Malika Madelin

UAR RIATE, 2023.

Cliquez ici pour voir la sortie HTML.

Cliquez ici pour voir le code source.

7.4. Un format de publication éclectique

Le format notebook peut être utilisé pour des objectifs très différents.

Exemples :

- **Mettre à disposition du code** explicité et réplicable

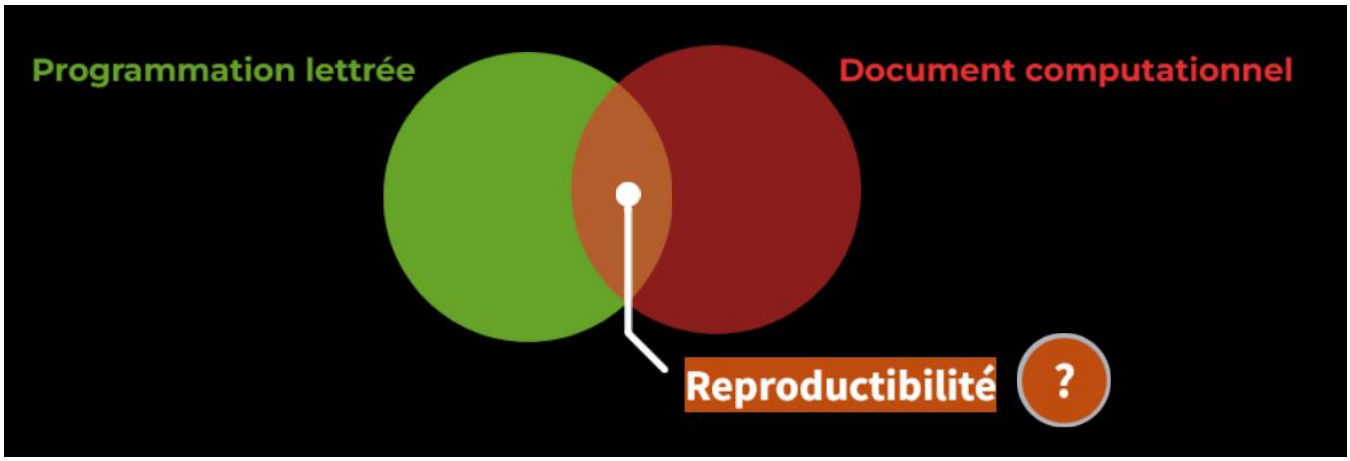
- **Produire un rapport statistique** qui peut être facilement mis à jour
- **Rédiger un *data paper***
- **Rédiger un article scientifique** qui incorpore les traitements présentés
- **Rédiger un livre**, en lien ou non avec un langage de programmation
- **Produire un support de communication**
- **Produire un site internet statique**
- …

L'hétérogénéité des possibilités en fait un format de publication puissant.

8. Notebook & reproductibilité

8.1. Progr. lettrée + Notebook = reproductibilité ?

Appliquer le paradigme de la programmation lettrée dans un notebook assure-t-il la reproductibilité de son travail ?



La réponse est non !

Appliquer le paradigme de la programmation lettrée dans un notebook facilite la compréhension et la transmission d'une méthodologie, mais **cela n'assure en aucun cas la reproductibilité des analyses ou traitements présentés.**

D'autres aspects doivent être pris en compte :

- **Comment partager son code source ?**
- **Comment inviter à sa réutilisation ?**
- **Comment assurer sa réplicabilité par un tiers ?**

8.2. Rendre un notebook reproductible

- **Mettre à disposition le code et les données** nécessaires à la **reproductibilité**
- **Fournir toutes les informations** (ou directement un environnement virtuel) **concernant la session et les versions logiciels utilisées** (version du système d'exploitation, du langage et des extensions utilisés...) nécessaires à la **réplicabilité**
- **Assigner une licence ouverte à votre travail** pour inviter à sa réutilisation !

Focus : Reproductibilité et réplicabilité ?

La réplicabilité est la capacité, par une équipe différente, de reproduire une expérience en ré-utilisant le même dispositif expérimental décrit (y compris les codes logiciels).

La reproductibilité est la capacité, par une équipe différente, de reproduire une expérience, sans se fier au dispositif expérimental et aux codes logiciels développés par l'équipe d'origine.

Source : https://science-ouverte.inrae.fr/fr/les-donnees-et-le-numerique-scientifiques/produire-des-donnees-fair/quest-ce-quune-recherche

9. Forges & Notebooks

9.1. Partager un notebook via une forge

Une forge, associée à un logiciel de gestion de versions décentralisé (ex : GitLab + Git), **permet de travailler et collaborer autour d'un projet logiciel**.

+info - Une forge en informatique...
... comporte généralement :
- un gestionnaire de code source,
- un visualiseur de code source,
- une gestion des droits d'accès,
- un gestionnaire de tickets,
- un espace de rédaction (wiki…)
- et des fonctionnalités de gestion de projet.

Source : Voir la page Wikipédia sur la Forge (informatique)

Une forge est un outil parfait pour **partager le code source d'un notebook** et même pour le **déployer (mettre en ligne)**, dans un contexte de **déploiement continu**.

+info - Déploiement continu
Le déploiement continu est une stratégie de développement logiciel dans laquelle les modifications apportées au code d'une application sont publiées automatiquement dans l'environnement de production. Cette automatisation repose sur une série de tests prédéfinis. Lorsque les tests des nouvelles mises à jour aboutissent, le système envoie les mises à jour directement aux utilisateurs du logiciel.
Il s'agit d'une technique très puissante pour le développement de documents computationnels, même réalisée de manière individuelle.
Cela permet la prise en compte automatique de la modification du code source d'un notebook. Cette mise à jour se répercute automatiquement sur le notebook mis en page (HTML) et mis en ligne. **Votre document est ainsi partagé de manière pérenne, tout en pouvant subir aisément des mises à jour régulières**.
Source : https://www.ibm.com/fr-fr/topics/continuous-integration

9.2. Notebook en déploiement continu ?

Voici l'exemple d'un notebook Quarto archivé et déployé par l'instance GitLab (forge open source) de l'IR* Huma-Num.

Le code source et ses ressources associées sont archivés et mis à disposition sur un dépôt public :
https://gitlab.huma-num.fr/elementr/session_notebook/programmation_lettree

Le notebook compilé est déployé en ligne grâce à GitLab :
https://elementr.gitpages.huma-num.fr/session_notebook/programmation_lettree/#/title-slide

Une modification du code source sur le dépôt de forge engendre une mise à jour automatique du document déployé sur le web !

10. Conclusion

10.1. Un format de publication légitime ?

Les documents computationnels offrent des fonctionnalités très puissantes dans le cadre de la recherche scientifique et particulièrement dans un contexte de science ouverte.
Le coût d'entrée technique et méthodologique de la pratique du notebook, voire des forges logicielles, n'est pas un acquis facile pour de nombreuses disciplines.
Toutefois, les évolutions récentes des systèmes de notebook (croissance des fonctionnalités et accès facilité) permettent à ce format de publication de devenir de plus en plus légitime.

Plusieurs initiatives communautaires vont dans ce sens...

Revues d'articles scientifiques :

- Computo : revue de statistique (format Quarto) portée par la Société Française de Statistique (SFdS). https://computo.sfds.asso.fr/
- Rzine : collection de publications méthodologiques sur la pratique de R en SHS (format Rmarkdown) portée par le collège Internationnal des sciences territoriales (FR CIST). https://rzine.fr/

Groupes de travail :

- GT Notebook : collectif interdisciplinaire et francophone dont l'objectif est de présenter les enjeux et problématiques du Notebook, pris comme un objet complexe au sein du cycle de vie des données de la recherche. https://gt-notebook.gitpages.huma-num.fr/site_quarto/
- Notebooks Now! : initiative communautaire internationnale et financée pour "Élever les notebooks au rang d'éléments primaires du dossier scientifique". https://data.agu.org/notebooks-now/

11. Références bibliographiques

Kery, Mary Beth, Radensky, Marissa, Arya, Mahima, John, Bonnie E. et Myers, Brad A. « The Story in the Notebook: Exploratory Data Science using a Literate Programming Tool ». Dans : Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems. ACM, 2018, p. 111. https://dl.acm.org/doi/10.1145/3173574.3173748