

TD 1g — Le notebook, ouvert de trois façons

Guide détaillé, étape par étape

Le TD ouvre un même notebook, `altitudes.ipynb`, de trois façons : dans le navigateur, sans rien installer ; dans l'éditeur de code, VS Code ; dans JupyterLab, lancé depuis Anaconda Navigator. Aucune des trois façons ne demande de créer un fichier ou un environnement. Le notebook reprend le programme du TD 1c, qui calcule la moyenne de trois altitudes, découpé en cellules. La dernière étape fait constater ce que le noyau retient d'une cellule à l'autre. Le TD dure une douzaine de minutes.

Le TD nécessite un navigateur, VS Code configuré au TD 1c, et Anaconda Navigator.

Étape	Objectif	Ce qu'on fait
1	exécuter un notebook sans rien installer	ouvrir le notebook dans le navigateur
2	exécuter un notebook dans un IDE	ouvrir le notebook dans l'éditeur
3	exécuter un notebook dans JupyterLab	ouvrir le notebook depuis Anaconda Navigator
4	relire l'état du noyau d'une cellule à l'autre	relancer une cellule, et lire ce que le noyau retient

Ce que chaque étape fait constater est expliqué à la fin du guide, dans « Ce que le TD fait constater » : faire l'étape d'abord, et noter ce qu'on observe, avant de lire l'explication.

1 · Ouvrir le notebook dans le navigateur

À faire : ouvrir la page jupyter.org/try-jupyter ; y déposer `altitudes.ipynb` ; exécuter ses cellules.

À obtenir : le notebook ouvert dans un onglet du navigateur, et les sorties de ses cellules.

Le dossier du TD

Le dossier `info01\cours1\lg_notebooks\` contient le notebook, la feuille du TD et un `README.md`. Il n'a pas de dossier `depart\` : le notebook est ouvert à son emplacement, sans copie préalable.

```
lg_notebooks\  
├── altitudes.ipynb      le notebook du TD  
├── td_lg_notebooks.pdf  la feuille du TD  
└── README.md
```

`altitudes.ipynb` contient neuf cellules : cinq blocs de texte et quatre cellules de code. Les cellules de code n'ont pas encore de sortie.

JupyterLite

1. Dans le navigateur, ouvrir l'adresse <https://jupyter.org/try-jupyter/lab/>. Le premier chargement prend une dizaine de secondes. La page ne demande ni compte ni installation.
2. Dans l'explorateur de fichiers de Windows, ouvrir `info01\cours1\lg_notebooks\`, puis faire glisser `altitudes.ipynb` dans le panneau de gauche de la page, celui qui liste les fichiers. Le bouton en forme de flèche vers le haut, au-dessus de ce panneau, dépose le fichier de la même façon, en passant par une fenêtre d'ouverture de fichier.
3. Double-cliquer sur `altitudes.ipynb` dans le panneau. Si la page demande de choisir un noyau, prendre celui qui est proposé, Python (Pyodide).
4. Cliquer dans la première cellule de code, puis taper Maj + Entrée : la cellule s'exécute, sa sortie s'affiche dessous, et la cellule suivante est sélectionnée. Continuer jusqu'à la fin du notebook. La première exécution prend quelques secondes.

Si le réseau de la salle est lent, l'enseignant montre cette étape au tableau : passer à l'étape 2.

À noter : ce qu'il a fallu créer ou installer pour exécuter le code, et où le calcul se fait.

2 · Ouvrir le notebook dans l'éditeur

À faire : ouvrir `altitudes.ipynb` dans VS Code ; choisir le noyau base d'Anaconda ; exécuter les cellules.

À obtenir : les quatre cellules de code exécutées, chacune avec son numéro entre crochets.

Ouvrir le fichier

Lancer VS Code, puis menu File, Open Folder (Fichier, Ouvrir le dossier), et choisir Bureau\info01\cours1\lg_notebooks. Dans le panneau Explorer, à gauche, cliquer sur `altitudes.ipynb`.

Le notebook s'ouvre en cellules, avec une barre d'outils en haut. Ouvrir un `.ipynb` nécessite l'extension Jupyter de Microsoft, d'identifiant `ms-toolsai.jupyter`, la troisième du module après l'extension Python (TD 1c) et l'extension C++ (TD 1e). Si elle n'a pas été installée avec les réglages du TD 1c, VS Code la propose à l'ouverture du fichier : accepter. Si VS Code ne la propose pas, l'installer depuis le panneau Extensions (Ctrl + Maj + X), en tapant son identifiant.

Choisir le noyau

1. En haut à droite du notebook, cliquer sur « Select Kernel ».
2. Choisir « Python Environments... », puis la ligne base, dont le chemin est `C:\ProgramData\anaconda3\python.exe`.

Ce que VS Code appelle « noyau » est l'interpréteur Python qui exécute les cellules. Choisir celui d'Anaconda ; le TD 1i explique pourquoi ce choix est nécessaire. Le nom du noyau choisi s'affiche ensuite à la place de « Select Kernel ».

Exécuter les cellules

Cliquer dans la première cellule de code, puis Maj + Entrée, comme dans le navigateur. Exécuter ainsi les quatre cellules, dans l'ordre. Le bouton « Run All » de la barre d'outils les exécute toutes d'un coup.

Un bloc de texte s'affiche mis en forme. Un double-clic dessus montre le Markdown qu'il contient, celui du TD 1f ; Maj + Entrée affiche de nouveau le texte mis en forme.

Vérification : chaque cellule de code a un numéro à gauche, [1] à [4], et la cellule de la somme affiche trois valeurs.

À noter : ce qu'affiche la cellule des données, qui n'appelle pas `print`.

3 · Ouvrir le notebook depuis Anaconda Navigator

À faire : lancer JupyterLab depuis la page d'accueil de Navigator ; ouvrir `altitudes.ipynb` dans son arborescence ; exécuter les cellules.

À obtenir : le notebook ouvert dans un onglet du navigateur, cette fois fourni par un serveur qui s'exécute sur le poste.

1. Menu Démarrer, taper `anaconda navigator`, Entrée, puis attendre la page d'accueil, qui a une fiche par application. En haut, la liste des environnements indique `base (root)`.
2. Sur la fiche JupyterLab, cliquer Launch. Une fenêtre noire s'ouvre, puis un onglet du navigateur, à une adresse qui commence par `localhost:8888/lab`. La fenêtre noire doit rester ouverte.
3. Le panneau de gauche de JupyterLab est une arborescence de fichiers. Y descendre, par double-clics, jusqu'au dossier du TD : `Desktop`, `info01`, `cours1`, `1g_notebooks`.
4. Double-cliquer sur `altitudes.ipynb`, puis exécuter les cellules par Maj + Entrée.

Le fichier est le même qu'à l'étape 2 : s'il a été enregistré dans VS Code avec ses sorties, JupyterLab les affiche à l'ouverture. Le nom du noyau s'affiche en haut à droite du notebook.

À noter : l'adresse de l'onglet, et ce qu'elle indique sur la machine qui fournit le notebook au navigateur.

Garder JupyterLab ouvert pour l'étape 4.

4 · Relancer une cellule, et lire ce que le noyau retient

À faire : relancer la cellule de la somme ; lire `total` ; puis relancer la somme sans remettre `total` à zéro ; redémarrer le noyau et tout exécuter.

À obtenir : les cellules de nouveau exécutées dans l'ordre, numérotées [1] à [4].

L'étape se fait dans JupyterLab, dont les menus sont cités ici ; dans VS Code, les mêmes commandes sont dans la barre d'outils du notebook.

Relancer une cellule

Le notebook se termine par la section « Ce que le noyau retient » : une cellule qui ne contient que `total`, et une consigne.

1. Cliquer dans la cellule de la somme, celle qui commence par `total = 0`, et taper Maj + Entrée.
2. Cliquer dans la dernière cellule, `total`, et taper Maj + Entrée.

À noter : le numéro entre crochets de chaque cellule après ces deux exécutions ; la valeur de `total`.

Relancer la somme sans sa première ligne

La première ligne de la cellule de la somme, `total = 0`, remet la variable à zéro chaque fois que la cellule s'exécute. Le texte du notebook, au-dessus de la cellule `total`, annonce que relancer la somme ajoute une seconde fois les trois altitudes : cette seconde addition ne se produit que si la cellule ne contient pas cette ligne. Pour voir ce que le noyau retient d'une exécution à l'autre, la retirer le temps d'un essai :

1. Dans la cellule de la somme, effacer la ligne `total = 0`.
2. Exécuter cette cellule seule, par Maj + Entrée, puis la dernière cellule, `total`.
3. Remettre la ligne `total = 0` en tête de la cellule, à l'identique.

À noter : les trois valeurs affichées par la cellule de la somme, et la valeur de `total`.

Redémarrer le noyau

Menu Kernel, « Restart Kernel and Run All Cells... », puis confirmer par Restart. Dans VS Code : bouton « Restart » de la barre d'outils, puis « Run All ».

Vérification : les cellules sont numérotées [1] à [4] dans l'ordre de la page, et `total` vaut 387.0.

Pour finir, dans JupyterLab : menu File, Shut Down. La fenêtre noire se ferme ; fermer ensuite l'onglet.

Ce que le TD fait constater

Cette section se lit après avoir fait les étapes.

Étapes 1 à 3 : trois façons d'ouvrir le même fichier

Où l'on ouvre le notebook	Ce qu'il a fallu faire	Ce qu'on constate
Dans le navigateur	ouvrir <code>jupyter.org/try-jupyter</code> , y déposer le fichier	aucun compte, aucune installation, le calcul se fait sur votre ordinateur
Dans l'éditeur	ouvrir le <code>.ipynb</code> , installer l'extension proposée, prendre l'interpréteur d'Anaconda	les cellules s'exécutent par Maj + Entrée
Depuis Anaconda	page d'accueil de Navigator, fiche JupyterLab, Launch	un onglet de navigateur, et le fichier dans l'arborescence

Les trois logiciels ouvrent le même fichier, et l'exécutent de la même façon. Ce qui change de l'un à l'autre est l'endroit où s'exécute le noyau, le programme qui exécute les cellules :

- JupyterLite, la version de JupyterLab ouverte à l'étape 1, n'a pas de serveur. Son noyau est un Python compilé pour le navigateur, Pyodide, qui s'exécute dans l'onglet : le code et les données saisis ne sont pas envoyés sur le réseau. JupyterLite est un service en ligne dont le calcul se fait sur votre machine. Le fichier déposé est enregistré dans le stockage interne du navigateur, et n'apparaît dans aucun dossier du disque.
- JupyterLab, lancé par Navigator, est un serveur qui s'exécute sur le poste : la fenêtre noire est ce serveur, et `localhost:8888` désigne le port 8888 de la machine même. L'onglet du navigateur affiche le notebook, sans exécuter le code.
- VS Code démarre lui-même le noyau, dans l'environnement choisi, sans passer par un serveur.

Aucune des trois façons n'a nécessité de créer un environnement : les postes de la salle ont la distribution Anaconda, qui installe JupyterLab et `ipykernel`, le paquet dont VS Code a besoin pour exécuter un notebook, dans base. Sur un ordinateur installé avec Miniforge, base n'a ni l'un ni l'autre : c'est le premier constat du TD 1i.

Dans le navigateur, une cellule `%pip install` peut installer une bibliothèque écrite entièrement en Python. En revanche, un code qui lance un autre programme du système par le module `subprocess`, comme `ffmpeg` ou `ImageMagick`, ne fonctionne pas. Le notebook du TD ne lance aucun autre programme.

Ce que les cellules affichent

Cellule	Sortie
les données	[128.4, 131.0, 127.6]
la somme	128.4, puis 259.4, puis 387.0
la moyenne	moyenne : 129.0 m
total	387.0

La dernière expression d'une cellule s'affiche sans `print`, comme dans la session interactive du TD 1c : la cellule des données affiche la liste, et la dernière cellule la valeur de `total`. Les trois valeurs de la somme sont celles que le débogueur montrait dans le panneau Variables au TD 1c.

Le bloc de texte de la moyenne contient une formule écrite en LaTeX, entre dollars, que le notebook affiche mise en forme. La formule et le code expriment le même calcul sur la même page, l'une pour être lue, l'autre pour être exécutée.

Étape 4 : ce que le noyau retient

Le numéro entre crochets donne l'ordre d'exécution ; il ne dépend pas de la place de la cellule dans la page. Après la première relance de l'étape 4, la cellule de la somme porte [5] et la dernière [6], alors que la moyenne garde [3]. Un notebook dont les numéros ne se suivent pas a été exécuté dans le désordre.

La dernière cellule affiche `total` sans le recalculer : la variable a été créée par une autre cellule, et le noyau la conserve en mémoire entre deux exécutions. Relancer la cellule de la somme telle qu'elle est écrite redonne 387.0, parce que sa première ligne, `total = 0`, remet la variable à zéro.

Sans cette ligne, la somme part de la valeur de `total` conservée par le noyau, et ajoute une seconde fois les trois altitudes : elle affiche 515.4, 646.4 puis 774.0, et `total` vaut 774.0. La cellule de la moyenne, qui n'a pas été relancée, affiche encore 129.0 m : la page montre alors des résultats qui ne correspondent plus entre eux.

Redémarrer le noyau efface toutes les variables ; le texte des cellules reste, et « Run All » les exécute de nouveau dans l'ordre de la page. Le redémarrage suivi de « Run All » remet ainsi le notebook dans un état cohérent. Sans la ligne `total = 0`, cette réexécution échouerait sur la cellule de la somme, avec une erreur `NameError` : après le redémarrage, `total` n'existe plus.

Le fichier `.ipynb`

Un `.ipynb` est un fichier texte au format JSON. Ouvert dans le Bloc-notes, il montre chaque cellule avec son type et ses lignes :

```
"cell_type": "code",
""
"source": [
  "altitudes = [128.4, 131.0, 127.6]\n",
  "altitudes"
]
```

Les sorties y sont enregistrées avec le code, une fois le notebook exécuté puis enregistré : rouvert, il affiche encore les résultats de la dernière exécution. Le notebook du TD a été écrit dans un autre format de notebook, MyST Markdown, du Markdown dont les cellules de code sont des blocs, puis converti en `.ipynb`.