

TD 1d – Un notebook dans JupyterLab

Guide détaillé, étape par étape (version 2, proposition)

Version 2 du cours 1 : proposition de travail pour 2027-2028. Ce guide reprend celui du TD 1g de 2026, réduit à JupyterLab lancé depuis un terminal, et ajoute l'écriture d'une cellule de texte en Markdown.

Le TD ouvre un notebook, `altitudes.ipynb`, dans JupyterLab, lancé depuis un terminal. Le notebook reprend le programme du TD 1c, qui calcule la moyenne de trois altitudes, découpé en cellules. La troisième étape fait constater ce que le noyau retient d'une cellule à l'autre ; la dernière ajoute au notebook une cellule de texte, écrite en Markdown. Le TD dure une douzaine de minutes.

Le TD nécessite un terminal où conda est actif, et un navigateur. Il lance JupyterLab depuis l'invite de commandes d'Anaconda, puis depuis Git Bash, où les commandes d'Anaconda sont disponibles depuis le début du TD 1c. Anaconda Navigator sert de dernier secours.

Étape	Objectif	Ce qu'on fait
1	ouvrir un notebook dans JupyterLab	lancer JupyterLab et ouvrir le notebook
2	exécuter un notebook cellule par cellule	exécuter les cellules
3	relire l'état du noyau d'une cellule à l'autre	relancer une cellule, et lire ce que le noyau retient
4	documenter un notebook en Markdown	ajouter une cellule de texte
5	lancer JupyterLab depuis un terminal	relancer JupyterLab depuis Git Bash, si le temps le permet

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 · Lancer JupyterLab et ouvrir le notebook

À faire : ouvrir l'invite de commandes d'Anaconda ; aller dans le dossier `cours1\1d_notebook\` ; lancer JupyterLab par `jupyter lab` ; ouvrir `altitudes.ipynb`.

À obtenir : le notebook ouvert dans un onglet du navigateur, à une adresse qui commence par `localhost:8888/lab`.

Le dossier du TD

Le dossier `info01\cours1\1d_notebook\` 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.

```
1d_notebook\  
├── altitudes.ipynb      le notebook du TD  
├── td_1d_notebook.pdf   la feuille du TD  
└── README.md
```

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

Lancer JupyterLab depuis l'invite de commandes d'Anaconda

1. Menu Démarrer, taper `anaconda prompt`, puis choisir « Anaconda Prompt ». Une fenêtre noire s'ouvre. Son invite commence par `(base)`, suivi du dossier personnel : `(base) C:\Users\eleve>`. `conda y` est toujours actif, sans réglage.
2. Taper la commande suivante, puis Entrée :

```
cd Desktop\info01\cours1\1d_notebook
```

L'invite se termine alors par `1d_notebook>`. Dans l'invite de commandes, les chemins s'écrivent avec des `\`.

3. Taper `jupyter lab`, puis Entrée. L'invite affiche des lignes de journal, puis le navigateur ouvre un onglet, à une adresse qui commence par `localhost:8888/lab`.
4. La fenêtre de l'invite reste occupée tant que JupyterLab tourne : elle doit rester ouverte.
5. Le panneau de gauche de JupyterLab est une arborescence de fichiers. Elle part du dossier où `jupyter lab` a été lancé, `1d_notebook`. Double-cliquer sur `altitudes.ipynb`.

Si le navigateur ne s'ouvre pas, l'invite affiche une adresse qui commence par `http://localhost:8888/lab?token=` : la recopier dans la barre d'adresse du navigateur.

Le module donne deux façons de lancer JupyterLab : l'invite de commandes d'Anaconda, qui n'a besoin d'aucun réglage, et Git Bash, le terminal du module, qui demande le réglage du TD 1c. L'étape 5 relance JupyterLab depuis Git Bash. Si un jour le réglage de Git Bash manque sur un poste, l'invite de commandes d'Anaconda reste disponible.

En dernier secours : Anaconda Navigator

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)`. Navigator peut mettre plusieurs minutes à s'ouvrir.
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 part cette fois du dossier personnel. Y descendre, par double-clics, jusqu'au dossier du TD : `Desktop, info01, cours1, 1d_notebook`. Double-cliquer sur `altitudes.ipynb`.

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.

2 · Exécuter les cellules

À faire : exécuter les cellules une à une, de haut en bas, par Maj + Entrée.

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

Avant toute exécution, les cellules de code ont des crochets vides, [], à leur gauche.

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.

Un bloc de texte s'affiche mis en forme. Un double-clic dessus montre le Markdown qu'il contient, dont la syntaxe est celle de la partie 3 du cours ; Maj + Entrée affiche de nouveau le texte mis en forme.

Vérification : chaque cellule de code a un numéro à gauche, [1] à [5], la cellule de la boucle affiche trois valeurs, et la moyenne vaut 129.0 m.

À noter : ce qu'affiche la cellule des données, qui n'appelle pas print, et ce qu'affiche la cellule total = 0.

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

À faire : exécuter une seconde fois la cellule de la boucle, puis celle de la moyenne ; lire `total` ; redémarrer le noyau et tout exécuter.

À obtenir : les cellules de nouveau exécutées dans l'ordre, numérotées [1] à [5].

Relancer la boucle

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 boucle, celle qui commence par `for`, et taper Maj + Entrée. Ne pas exécuter la cellule `total = 0`, qui la précède.
2. Cliquer dans la cellule de la moyenne, et taper Maj + Entrée.
3. Cliquer dans la dernière cellule, `total`, et taper Maj + Entrée.

À noter : les trois valeurs affichées par la boucle, la moyenne, la valeur de `total`, et le numéro entre crochets de chaque cellule après ces trois exécutions.

Redémarrer le noyau

Menu Kernel, « Restart Kernel and Run All Cells... », puis confirmer par Restart.

Vérification : les cellules sont numérotées [1] à [5] dans l'ordre de la page, `total` vaut 387.0, et la moyenne 129.0 m.

4 · Ajouter une cellule de texte

À faire : ajouter une cellule en tête du notebook ; en faire une cellule de texte ; y écrire, en Markdown, un titre, une phrase, une liste et un lien ; l'afficher mise en forme ; enregistrer.

À obtenir : une première cellule sans numéro entre crochets, qui affiche un titre, une phrase, une liste à puces et un lien.

1. Cliquer sur la première cellule du notebook, puis taper Échap. La cellule passe en mode commande, où une touche agit sur la cellule entière. Taper A : une cellule vide apparaît au-dessus.
2. Taper M. La liste déroulante de la barre d'outils, en haut du notebook, indique alors « Markdown » ; choisir « Markdown » dans cette liste fait le même changement.
3. Cliquer dans la nouvelle cellule, et y écrire, en Markdown :
 - un titre, la ligne commençant par # suivi d'un espace ;
 - une ligne vide, puis une phrase qui dit ce que fait le programme ;
 - une ligne vide, puis la liste des trois altitudes, une ligne par altitude, chacune commençant par - suivi d'un espace ;
 - une ligne vide, puis un lien vers le site de Jupyter, écrit [texte] (adresse), par exemple [Jupyter] (<https://jupyter.org>).
4. Taper Maj + Entrée : la cellule s'affiche mise en forme. Un double-clic sur la cellule revient au texte tapé.
5. Enregistrer le notebook, Ctrl + S.

La syntaxe est celle de la diapositive « La syntaxe minimale de Markdown ».

À noter : ce qui distingue, à gauche, la cellule de texte d'une cellule de code ; l'aspect du texte pendant qu'on l'écrit, puis après Maj + Entrée.

Pour arrêter JupyterLab : dans JupyterLab, menu File, Shut Down, puis fermer l'onglet. L'invite de commandes d'Anaconda attend de nouveau une commande. Ctrl + C dans l'invite arrête aussi JupyterLab ; s'il demande une confirmation, taper y puis Entrée. Lancé depuis Navigator, JupyterLab se ferme avec sa fenêtre noire.

5 · Relancer JupyterLab depuis Git Bash, si le temps le permet

À faire : ouvrir Git Bash dans le dossier `cours1/1d_notebook/` ; lancer JupyterLab par `jupyter lab` ; rouvrir `altitudes.ipynb`.

À obtenir : le notebook, avec la cellule de texte de l'étape 4.

1. Dans l'explorateur de fichiers, ouvrir Bureau\info01\cours1\1d_notebook, puis clic droit sur un endroit vide du dossier, « Afficher d'autres options », « Open Git Bash here ». À défaut, ouvrir Git Bash depuis le menu Démarrer, puis taper `cd ~/Desktop/info01/cours1/1d_notebook`.
2. Taper la commande suivante, puis Entrée :

```
jupyter lab
```

Git Bash affiche des lignes de journal, puis le navigateur ouvre un onglet, à une adresse qui commence par `localhost:8888/lab`.

3. Git Bash reste occupé tant que JupyterLab tourne : sa fenêtre doit rester ouverte.
4. Le panneau de gauche de JupyterLab est une arborescence de fichiers. Elle part du dossier où `jupyter lab` a été lancé, `1d_notebook`. Double-cliquer sur `altitudes.ipynb`.

Si le navigateur ne s'ouvre pas, Git Bash affiche une adresse qui commence par `http://localhost:8888/lab?token=` : la recopier dans la barre d'adresse du navigateur.

Si Git Bash répond `jupyter: command not found`, le début du TD 1c, qui rend les commandes d'Anaconda disponibles dans Git Bash, n'a pas été fait sur ce poste. L'invite de commandes d'Anaconda de l'étape 1 reste disponible.

Vérification : le notebook s'ouvre avec la cellule de texte de l'étape 4, enregistrée dans le fichier. Arrêter ensuite JupyterLab, par File, Shut Down, ou `Ctrl + C` dans Git Bash.

Ce que le TD fait constater

Cette section se lit après avoir fait les étapes.

Étape 1 : JupyterLab, un serveur sur le poste

JupyterLab, lancé par `jupyter lab` ou par Navigator, est un serveur qui s'exécute sur le poste : le programme qui occupe Git Bash, ou la fenêtre noire de Navigator, 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. Le noyau, le programme qui exécute les cellules, est démarré par ce serveur, sur le même poste. Le cours 5 revient sur le client et le serveur.

Le TD n'a pas nécessité de créer un environnement : les postes de la salle ont la distribution Anaconda, qui installe JupyterLab dans base.

Ce que les cellules affichent

Cellule	Sortie
les données	[128.4, 131.0, 127.6]
<code>total = 0</code>	rien
la boucle	128.4, puis 259.4, puis 387.0
la moyenne	moyenne : 129.0 m
<code>total</code>	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`. La cellule `total = 0` n'affiche rien : une affectation ne produit pas de valeur à afficher. Les trois valeurs de la boucle sont celles que prend `total` à chaque tour.

Étape 3 : 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 les trois exécutions de l'étape 3, la boucle porte [6], la moyenne [7] et la dernière cellule [8], alors que la cellule `total = 0` garde [2]. 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.

La cellule `total = 0` n'a pas été relancée. La boucle part donc 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 moyenne, relancée ensuite, vaut 258.0 m : la page montre alors une moyenne qui ne correspond plus aux trois altitudes affichées.

Redémarrer le noyau efface toutes les variables ; le texte des cellules reste, et « Restart Kernel and Run All Cells » les exécute de nouveau dans l'ordre de la page. Le redémarrage remet ainsi le notebook dans un état cohérent.

Étape 4 : la cellule de texte

Une cellule de texte possible, ligne par ligne :

Ce qu'on tape	Ce qui s'affiche
<code># Moyenne des altitudes</code>	un titre, en gros caractères
une ligne vide, puis <code>Calcule la moyenne de trois altitudes, en mètres.</code>	un paragraphe
une ligne vide, puis <code>- 128.4, - 131.0 et - 127.6, un par ligne</code>	une liste de trois puces
une ligne vide, puis <code>[Jupyter](https://jupyter.org)</code>	le mot Jupyter, souligné, qui ouvre jupyter.org au clic

Le # et le - sont suivis d'un espace. Tant que la cellule est en cours d'écriture, le texte reste brut ; Maj + Entrée l'affiche mis en forme.

Une cellule de texte n'a pas de numéro entre crochets : elle ne passe pas par le noyau, et Maj + Entrée ne fait que la mettre en forme. En mode commande, A insère une cellule au-dessus de la cellule sélectionnée, B au-dessous.

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"  
]
```

La cellule de texte ajoutée à l'étape 4 y a le type "markdown". 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.