



Outils d'optimisation et d'algèbre linéaire numérique

Contacts : Paul Raynaud

1 Julia

1.1 Installation

Afin d'installer JULIA sur votre plateforme (Windows, MacOS ou Linux) il suffit de suivre les instructions sur la section des téléchargements de julialang.org pour récupérer la dernière version stable du langage. La version de Julia installée sur votre ordinateur doit être au minimum la v1.6.

1.2 REPL

Lorsqu'on lance JULIA, on obtient l'invite de commande REPL (read-eval-print loop). On peut basculer par la suite sur des menus différents en tapant un caractère spécial après le `>` :

- `?` pour obtenir de l'aide sur le fonctionnement d'une fonction ;
- `|` pour accéder au gestionnaire d'installation des modules ;
- `;` pour utiliser le terminal de notre système d'exploitation.

1.3 Paquets

JULIA possède de très nombreuses bibliothèques de programmes. Parmi ces bibliothèques, certaines sont écrites en pur langage Julia, tandis que d'autres réalisent une interface entre Julia et une bibliothèque écrite dans un autre langage comme du C, du Fortran ou encore du Python! L'installation de ces paquets se fait à l'aide du gestionnaire de paquets appelé Pkg. Les principales commandes de gestion de package sont :

- **add package** installe le paquet et ses dépendances ;
- **rm package** supprime le paquet ;
- **update package** met à jour le paquet s'il possède une version plus récente ;
- **status** liste les paquets installés et leur version ;
- **update** met à jour l'ensemble des paquets installés.

1.4 Fichiers de programmes Julia

Les fichiers de programmes en JULIA sont des fichiers avec l'extension `.jl`. On peut les exécuter dans Julia de la manière suivante :

```
julia> include("code.jl")
```

2 VS Code

2.1 Installation

Pour installer VISUAL STUDIO CODE, il faut se rendre sur le site :

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

Il suffit ensuite de sélectionner votre système d'exploitation et de suivre la procédure d'installation. Visual Studio Code est un éditeur de code, c'est à dire qu'on l'utilise pour écrire notre code. Un éditeur de code intègre des outils facilitant la rédaction de code (coloration syntaxique, auto-complétion, documentation de fonctions lorsque l'on passe le curseur sur une fonction, etc...). Un des nombreux avantages de VS Code est la possibilité d'installer des *extensions* permettant d'améliorer l'environnement de programmation de l'utilisateur.

2.2 Utiliser Julia dans VS Code

La documentation est disponible à l'adresse :

<https://www.julia-vscode.org/docs/stable/>

Démarrer VS Code. Dans la liste tout à gauche un icône avec 4 cubes désignent les *extensions*. Dans l'outil de recherche, chercher JULIA et l'installer.

2.2.1 Julia terminal dans VS Code

Outre l'auto-complétion et la coloration du code, cette extension offre aussi la possibilité d'avoir un terminal avec Julia au sein de VS Code.

```
Ctrl + Shift + P  
>julia Start REPL
```

ou bien le raccourci

```
Alt + j + o
```

Un terminal avec Julia s'ouvre en bas de page et vous pouvez taper votre première commande.

2.2.2 Créer et exécuter un fichier Julia

Créer un nouveau fichier, *File - New File*.
Ecrire une commande Julia par exemple

```
println("Hello world")
```

sauvegarder le fichier avec l'extension .jl qui définit les fichiers Julia.

Vous pouvez exécuter ce fichier en cliquant sur le triangle en haut à droite de l'écran (ou alors Ctrl + Shift + P puis >julia execute code).

2.2.3 Pour aller plus loin

Quelques vidéos pour en savoir plus ou expliquer différemment :

- <https://code.visualstudio.com/docs/introvideos/basics>
- <https://code.visualstudio.com/docs/introvideos/extend>
- <https://www.youtube.com/watch?v=IdhnP00Y1Ks>

2.3 [Pour aller plus loin] Markdown et VS Code

Il est possible de visualiser les codes MARKDOWN dans VS Code, pour cela il faut l'extension :

<https://marketplace.visualstudio.com/items?itemName=shd101wyy.markdown-preview-enhanced>

2.4 [Pour aller plus loin] Jupyter et VS Code

Il est possible de visualiser les codes JUPYTER dans VS Code, pour cela il faut l'extension :

<https://devblogs.microsoft.com/python/introducing-the-jupyter-extension-for-vs-code/>

3 Jupyter

JUPYTER permet de combiner dans des documents du texte, des formules mathématiques, des graphiques, des images ou encore des vidéos avec du code informatique. On appelle ces documents des notebooks. Un fichier notebook est reconnaissable par son extension **.ipynb**. JUPYTER est utilisable avec plusieurs langages de programmation (JULIA, PYTHON et R). On peut basculer de l'un à l'autre en changeant de noyau. Pour lancer JUPYTER depuis JULIA, on a besoin du paquet *IJulia*.

```
julia> using IJulia
julia> notebook(detached=true)
```

4 Markdown

MARKDOWN est un langage ayant une syntaxe facile à lire et écrire. Il est utilisé dans les carnets JUPYTER et PLUTO ainsi que sur GITHUB (voir plus loin). On reconnaît les fichiers écrits en MARKDOWN grâce à l'extension `.md` ou `.markdown`. Les commandes sont récapitulées dans la *cheat sheet* suivante

- <https://www.markdownguide.org/cheat-sheet/>
- <https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet>

Vous pouvez tester le code suivant en ligne sur dillinger.io pour familiariser avec la syntaxe.

```
# Hello Poly !
## Joyeuse début de session
### bon courage en Markdown
#### et en Julia
```

On peut mettre du texte en forme en l'encadrant par des caractères spéciaux.

Texte en **italique**, en ****gras**** ou les `*__deux__*`.

~~Si on s'est trompé on peut tout rayé.~~

Ma liste de bonne résolution:

1. Faire une listes
2. Bien organisé
 - avec des sous-catégories
 - et encore des sous-catégories
 - il faut faire attention aux espaces
3. Ou d'autres listes avec des + ou * au lieu des -.

Ma liste de bonnes résolutions dans un tableau:

Tables	Are	Cool	
-----	:-----:	-----:	
col 3 is	right-aligned	\$1600	
col 2 is	centered	\$12	
zebra stripes	are neat	\$1	

On peut avoir la coloration syntaxique d'un langage

```
““julia
```

```
println("Hello Poly!")
```

```
““
```

, écrire des mathématiques $\Delta u(x) = 0$, $u_0(x) = 0$,
ajouter un lien vers une [vidéo](lien_de_la_video) ou encore
insérer des images [![label_image](lien_de_mon_image)]

5 Git

Git est un logiciel de révisions de code. Pour une liste exhaustive de commandes, vous pouvez regarder la *cheat sheet*.

6 GitHub

GitHub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. Il nous permet d'héberger en ligne nos codes et favorisent ainsi l'exposition des codes ainsi que le travail collaboratif. Pour créer un compte il suffit de suivre la procédure sur github.com.

Un exemple de dépôt GITHUB complet :

<https://github.com/JuliaSmoothOptimizers/Krylov.jl>

<https://github.com/JuliaSmoothOptimizers/NLPModelsJuMP.jl>

Atlassian regroupe une documentation complète de l'outil git

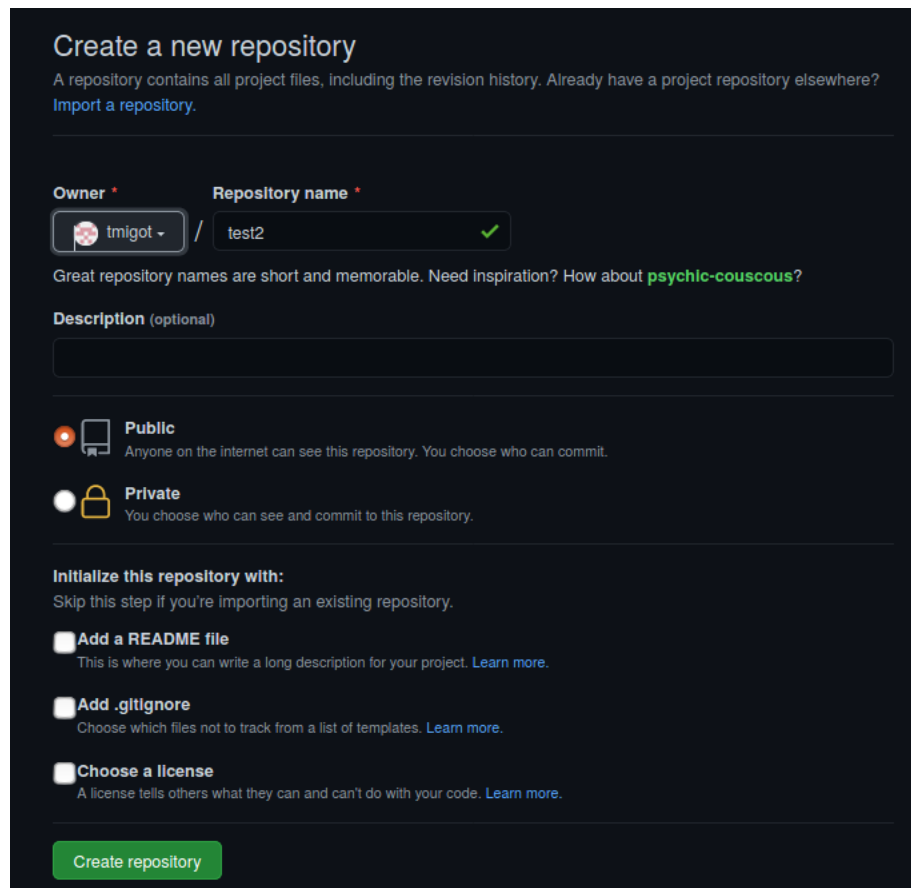
<https://www.atlassian.com/git>

La suite de cette section couvre les basiques de git/Github.

6.1 Créer un dépôt

Objectif 1 : Créer un premier dépôt qui s'appelle : **Bonjour le monde** soit en passant par <https://github.com/new> (pas très pratique de garder cette adresse) ou plutôt en cliquant sur *Your repository*

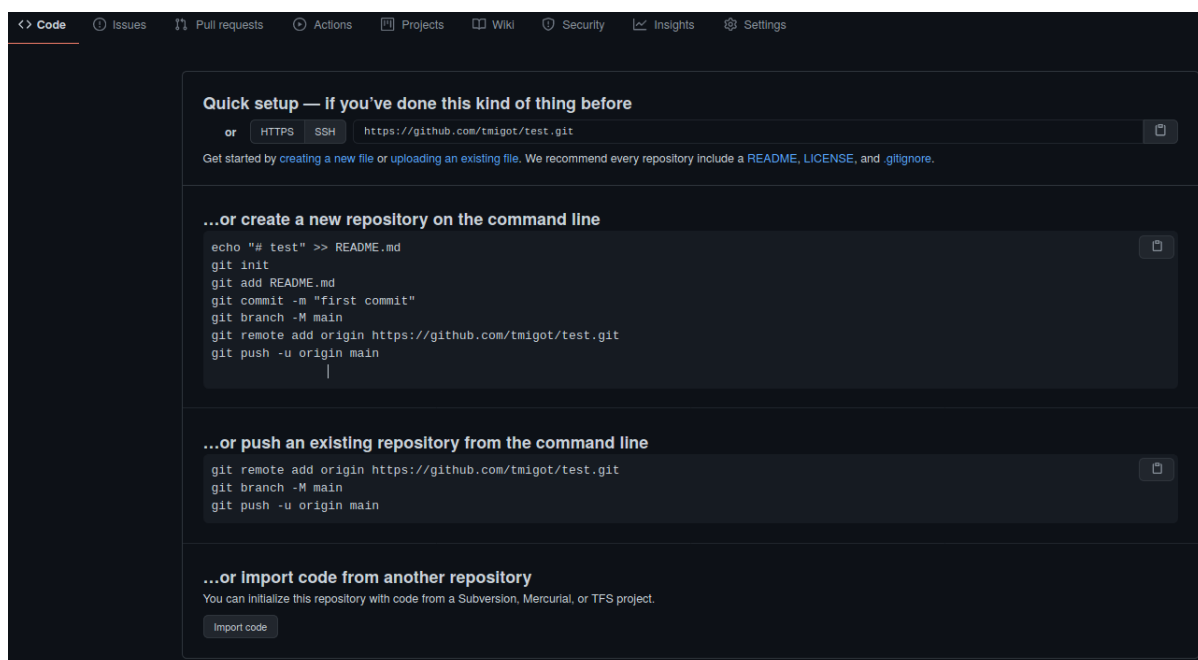
et *New*. Il suffit de remplir le nom et une description de votre dépôt et de choisir si tout le monde peut y accéder ou pas (dépôt public ou privé).



The screenshot shows the 'Create a new repository' page on GitHub. At the top, it says 'Create a new repository' and 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.' Below this, there are two input fields: 'Owner' with a dropdown menu showing 'tmigot' and 'Repository name' with the text 'test2' and a green checkmark. A hint text says 'Great repository names are short and memorable. Need inspiration? How about **psychic-couscous**?'. There is a 'Description (optional)' text area. Below that, there are two radio buttons for 'Public' (selected) and 'Private'. The 'Public' option says 'Anyone on the internet can see this repository. You choose who can commit.' and the 'Private' option says 'You choose who can see and commit to this repository.' Below this, there is a section 'Initialize this repository with:' with the text 'Skip this step if you're importing an existing repository.' and three checkboxes: 'Add a README file' (selected), 'Add .gitignore' (selected), and 'Choose a license' (selected). Each checkbox has a brief description and a 'Learn more' link. At the bottom, there is a green 'Create repository' button.

Félicitations vous avez créé votre premier dépôt (!) Github qui ne contient pour l'instant aucun fichier.

A noter l'adresse de votre dépôt (du type `https://github.com/username/name.git`) qui va s'avérer très utile pour connecter notre code en local (sur notre ordinateur) et GITHUB.



The screenshot shows the 'Quick setup' section on the GitHub repository page. It has a navigation bar at the top with links for 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. The main content area has a 'Quick setup — If you've done this kind of thing before' section with a dropdown menu for 'HTTPS' and 'SSH' and a text input field containing 'https://github.com/tmigot/test.git'. Below this, it says 'Get started by creating a new file or uploading an existing file. We recommend every repository include a README, LICENSE, and .gitignore.' There are three sections for creating a new repository on the command line: '...or create a new repository on the command line' with a code block containing the following commands:

```
echo "# test" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/tmigot/test.git
git push -u origin main
```

; '...or push an existing repository from the command line' with a code block containing the following commands:

```
git remote add origin https://github.com/tmigot/test.git
git branch -M main
git push -u origin main
```

; and '...or import code from another repository' with a text input field and an 'Import code' button.

6.2 Github en ligne de commande

Un terminal *git bash* peut-être installé au lien suivant

<https://gitforwindows.org/>

à partir duquel les commandes git peuvent être exécutées.

6.3 Créer un premier commit

Maintenant, on crée un premier fichier README.md sur notre ordinateur (un fichier en MARKDOWN d'où l'extension .md) puis on va connecter notre dossier de travail à notre compte GITHUB.

```
git init
git add README.md
git commit -m "first commit"
git branch -M main #par défaut la branche principale s'appelle master sur git et main sur github .
git remote add origin https://github.com/tmigot/test.git
git push -u origin main
```

Félicitations, vous avez votre premier fichier sur le dépôt ! Il ne reste plus qu'à customiser votre fichier README.md mettre à jour la branche local puis à nouveau celui en ligne pour avoir un dépôt présentable.

6.4 Github dans VS Code

VS Code a par défaut les extensions nécessaires à l'utilisation de Git. Un supplément utile est d'installer l'extension *GitHub Pull Requests and Issues*

<https://code.visualstudio.com/docs/editor/github>

Dans les prochaines étapes nous allons cloner un dépôt, réaliser un commit puis créer un push/pull. Plus d'informations concernant ces opérations dans VS Code ici :

<https://code.visualstudio.com/docs/editor/versioncontrol>

et des informations sur git et github :

[https://openclassrooms.com/fr/courses/](https://openclassrooms.com/fr/courses/5641721-utilisez-git-et-github-pour-vos-projets-de-developpement?archived-source=2342361)

[5641721-utilisez-git-et-github-pour-vos-projets-de-developpement?archived-source=2342361](https://openclassrooms.com/fr/courses/5641721-utilisez-git-et-github-pour-vos-projets-de-developpement?archived-source=2342361)

6.4.1 Clone un dépôt github

Pour gagner en pratique nous allons démarrer un nouveau projet à partir d'un dépôt déjà existant en ligne. Dans ce cas de figure, la première opération consiste à *clone* le dépôt existant, pour cet exemple

<https://github.com/tmigot/TPO-MTH8Poly>

Créer une branche de ce dépôt (en cliquant sur *fork*) permet d'avoir une version sur votre compte. On va maintenant cloner cette version sur votre machine.

Dans VS Code :

```
Ctrl + Shift + P
clone from github
```

A ce stade (si ce n'est pas déjà fait) VS Code va probablement demander à vous authentifier à votre compte GITHUB et demander à pouvoir communiquer librement avec ce dernier.

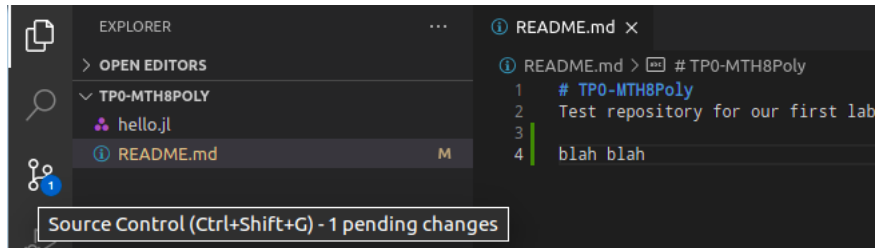
Ensuite, VS Code va demander un lieu sur votre machine où créer le dossier contenant le dépôt copier. Puis *open the clone repository* dans un pop-up en bas à droite de l'écran.

C'est fait le nouveau projet est créer !

6.4.2 Commit et push

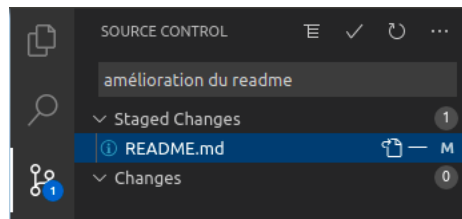
A vous de jouer, modifier le fichier README.md (.md implique que c'est du MARKDOWN) pour proposer un message d'accueil accueillant à notre projet.

C'est fait ? Il est temps de vérifier votre travail, le valider puis le proposer au dépôt GITHUB. Dans la colonne de gauche c'est l'icone *source control* (les trois cercles reliés) qui propose l'interface de gestion des versions.



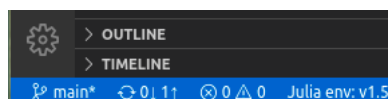
Sous l'onglet, *changes*, on peut observer la liste des fichiers modifiés. Et en cliquant sur un des fichiers, on obtient un *diff* qui récapitule les modifications réalisées.

Si la modification vous convient (sinon retour à la modification du fichier), alors clique droit puis *Stage changes* (ou alors le + à droite du nom du fichier). Votre fichier vient de passer dans la catégorie *Stage changes*, i.e. qu'il est validé est prêt à être *commit*.



Dans la zone de texte au dessus de *Stage changes* décrivez en un court message la nature de votre modification, puis la "flèche" *commit* au dessus va confirmer votre modification. A noter qu'on *commit* plusieurs fichiers, tous ceux qui étaient en *stage changes* iront dans la même modification.

Le *commit* que l'on vient de réaliser a modifié votre version local du projet. Il est temps maintenant de l'envoyer vers le dépôt GITHUB :



en cliquant sur la flèche avec le numéro 1 (qui correspond au nombre de *commit*) vous envoyez une requête au dépôt GITHUB relié de mise à jour.