Instructions pour l'installation et l'utilisation de Julia

Installer Julia : [Download Julia (julialang.org)]

Julia peut être installé via le Microsoft store mais il est plus simple de le télécharger manuellement. Pour cela, il faut chercher sur la page Download « Manual Download ».

Exemple pour Windows ci-dessous : cliquer sur « installer »

Manual Download

Please see platform specific instructions for further manual installation instructions. If the official binaries do not work for you, please file an issue in the Julia project.

Current stable release: v1.11.2 (December 1, 2024)

Release notes | GitHub tag | SHA256 checksums | MD5 checksums

Platform	64-bit	32-bit
Windows [help]	installer, portable	installer, portable
macOS (Apple Silicon) [help]	.dmg, .tar.gz	
macOS x86 (Intel or Rosetta) [help]	.dmg, .tar.gz	
Generic Linux on x86 [help]	glibc (GPG), musl[1] (GPG)	glibc (GPG)
Generic Linux on ARM [help]	.tar.gz (GPG)	

Attention : penser à cocher la case qui permet de mettre julia dans le PATH

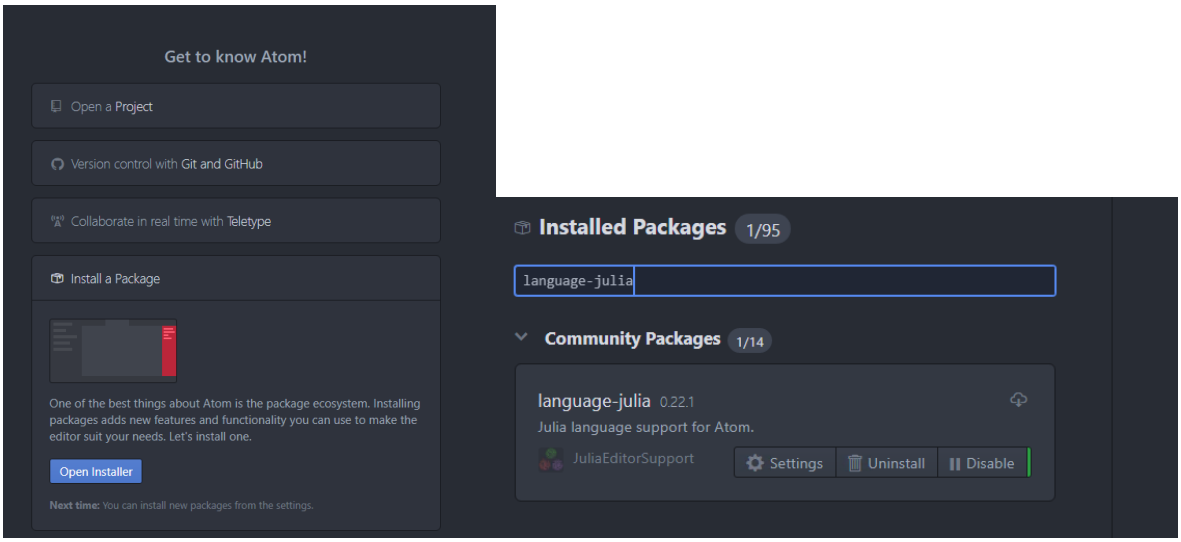
La variable PATH contient tous les dossiers dans lesquels il faut chercher la commande exécutée.

Si vous avez installé julia sans cocher la case pour modifier le PATH, la commande julia ne sera pas reconnue à part dans le dossier d'installation de julia. Vous pouvez cependant modifier vous-même la variable PATH pour ajouter le dossier qui contient l'exécutable de julia (chercher sur internet comment faire – par exemple Procédure : ajouter des emplacements d'outils à la variable d'environnement PATH | Microsoft Learn)

NB : ce problème de PATH peut arriver avec d'autres logiciels ou d'autres langages, par exemple pour Python c'est pareil il faut cocher une case au moment de l'installation

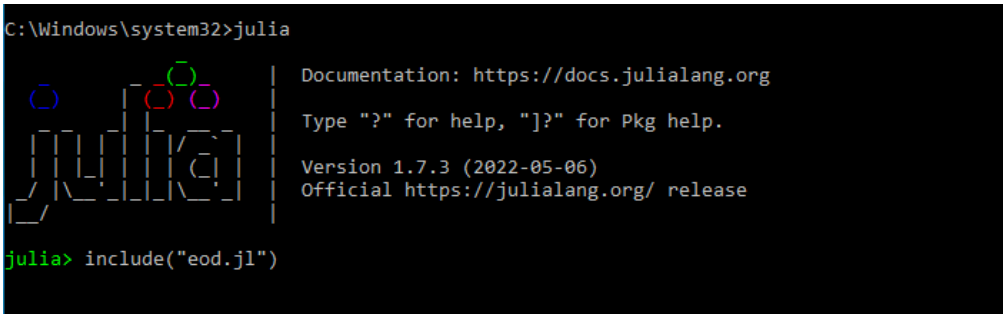
Installer Atom : https://atom.en.softonic.com/ et installer le package « language-julia »

Attention : ce package n'est plus directement disponible dans Atom, il faut télécharger le zip sur la page github du package (GitHub - JuliaEditorSupport/atom-language-julia: A Julia language support package for the Atom editor) et le dézipper dans le dossier des packages atom (dossier .atom dans votre dossier utilisateur)



Utilisation de Julia :

- Ouvrir l'invite de commandes (vous pouvez ouvrir File Explorer à l'endroit souhaité et taper « cmd » à la place du chemin du dossier et appuyer sur Entrée)
- Taper « julia » dans l'invite de commandes
- Pour lancer un script, taper include(« nom_du_script.jl ») dans l'invite de commandes Julia



-

Installer des packages (Pkg · The Julia Language) :

- Appuyer sur la touche «] » (Alt Gr + touche °) dans l'invite de commandes Julia
- Taper « add XXXX.jl » avec XXXX le nom du package à installer

- Appuyer sur la touche Retour en arrière pour revenir à l'invite de commandes Julia

A faire : installer les packages JuMP, HiGHS et XLSX

JuMP est le package Julia pour l'optimisation (Introduction · JuMP https://jump.dev/JuMP.jl/stable/tutorials/getting_started/getting_started_with_JuMP/). Il est facile à prendre en main et permet de résoudre un grand nombre de problèmes (exemples : Economic Dispatch & Unit Commitment : Power Systems · JuMP).

HiGHS est l'un des solveurs open-source compatibles avec JUMP pour résoudre des problèmes d'optimisation linéaires continus ou avec des variables entières. Il existe cependant beaucoup de solveurs compatibles avec JuMP (Installation Guide · JuMP). Il faudra donc dans ce projet réfléchir au choix du solveur. Evidemment il faudra choisir un solveur compatible avec le type de problème d'optimisation à résoudre. Ensuite, il faudra choisir un solveur avec un algorithme adapté : a-t-on besoin d'obtenir l'optimum global ou local ou bien une solution approchée obtenue par une heuristique est-elle acceptable ? A-t-on un problème difficile à résoudre (par exemple problème de grande taille) ? Il faudra se documenter pour savoir quels solveurs sont considérés comme l'état de l'art pour telle ou telle classe de problème. Il ne faudra pas hésiter à comparer des solveurs entre eux si on n'est pas satisfaits du solveur choisi (sur des petits problèmes il n'y aura peut-être pas de différence significative mais sur des problèmes réels il y aura sûrement des différences importantes).

XLSX est un package qui permet de lire des fichiers Excel, ce qui peut être très pratique pour lire des données d'entrée et ainsi éviter de les écrire à la main. Attention : pour lire un fichier Excel, celui-ci doit être fermé sinon cela entraîne des problèmes d'accès.

TP phase n°1 : EOD pas à pas

Afin de prendre en main Julia et son module optimisation JuMP, nous allons construire un problème d'Equilibre Offre-Demande (EOD) simple portant uniquement sur le vecteur électricité pour une seule et unique zone. Nous allons construire ce problème pas à pas.

1. EOD simple

Nous commençons par une approche simple : nous avons une consommation d'électricité à satisfaire (2200 MWh) et cinq moyens de production d'électricité pour satisfaire cette demande :

- Une centrale nucléaire avec une capacité de 900 MW et un prix de marché égal à 14€/MWh,
- Une deuxième centrale nucléaire avec une capacité de 900 MW et un prix de marché égal à 16€/MWh,
- Une centrale CCG (Cycle Combiné Gaz) avec une capacité de 300 MW et un prix de marché égal à 45€/MWh
- Une centrale hydraulique avec une capacité de 300 MW et un prix de marché égal à 48€/MWh,
- Un parc éolien avec une capacité de 300 MW et un prix de marché égal à 0€/MWh.

L'objectif consiste à déterminer le plan de production qui satisfait cette demande au moindre coût.

Un cas aussi simple peut évidemment se résoudre à la main. Cependant, il peut aussi être formulé comme un problème d'optimisation qui est implémenté en Julia dans le fichier « eod_step_by_step_to_complete.jl » entre les lignes 10 et 52.

2. EOD avec contrainte de puissance minimum

Le premier problème d'EOD que nous avons vu a le mérite d'être simple mais il n'est pas réaliste. En effet, pour chaque moyen de production, il existe des contraintes techniques de fonctionnement qu'il faut prendre en compte. La première contrainte de fonctionnement que nous étudions est la contrainte de puissance minimale de fonctionnement : certains moyens de production, s'ils sont démarrés, doivent fonctionner à une puissance minimale. Dans notre exemple, les deux centrales nucléaires doivent fonctionner à 300 MW minimum si elles sont démarrées et la centrale CCG doit fonctionner à 150 MW minimum si elle est démarrée.

Exercice : modéliser cette contrainte de puissance minimum (indice : introduire des variables binaires qui indiquent si chaque groupe de production est en fonctionnement ou à l'arrêt) et l'ajouter dans le problème d'optimisation dans le fichier « eod_step_by_step_to_complete.jl » entre les lignes 56 et 101.

3. EOD avec contrainte de rampe

Les deux problèmes d'EOD que nous venons de voir portaient uniquement sur une heure mais en pratique, il faut prévoir un plan de production pour plusieurs heures consécutives. Dans cette troisième version du problème d'EOD, nous cherchons donc un plan de production pour trois heures consécutives (consommations à satisfaire : 2200 MWh, 2450 MWh et 1900 MWh). De plus, nous introduisons une nouvelle contrainte technique de fonctionnement : la contrainte de rampe. Cette contrainte limite la variation de puissance d'un moyen de production entre deux heures consécutives. Dans notre exemple, nous imposons une variation maximale de 350 MW/h pour les centrales nucléaires et une variation maximale de 200 MW/h pour la centrale CCG.

Remarque : ces données sont fictives et ont pour seul objectif de complexifier progressivement le problème d'optimisation, en réalité la contrainte de rampe sur ces moyens de production n'est pas limitante au pas de temps horaire.

Exercice : modéliser cette contrainte de rampe et l'ajouter dans le problème d'optimisation dans le fichier « eod_step_by_step_to_complete.jl » entre les lignes 105 et 157.

4. EOD avec contrainte de durée de fonctionnement

Une autre contrainte de fonctionnement importante concerne la durée de fonctionnement des moyens de production. En effet, certains moyens de production ont une contrainte de durée minimum de fonctionnement et/ou d'arrêt. Dans notre exemple, les centrales nucléaires ont une durée minimum de fonctionnement de vingt-quatre heures et la centrale CCG a une durée minimum de fonctionnement de trois heures.

Exercice : modéliser cette contrainte de durée de fonctionnement (indice : introduire de nouvelles variables binaires UP égales à 1 si le groupe démarre à l'instant t et 0 sinon) et l'ajouter dans le problème d'optimisation dans le fichier « eod_step_by_step_to_complete.jl » entre les lignes 161 et 226.

TP phase n°2 : EOD hebdomadaire

Nous allons maintenant étudier un problème d'EOD un peu plus réaliste mais portant toujours uniquement sur le vecteur électricité pour une seule zone : un problème d'EOD sur une semaine entière (soit 168 heures) avec

- quatre moyens de production renouvelable :
    - solaire,
    - éolien,
    - hydraulique fil de l'eau,
    - une dernière catégorie fatale regroupant la biomasse, les déchets et la cogénération) ;
- cinq moyens de production thermique :
    - nucléaire,
    - charbon,
    - gaz semi-base,
    - gaz pointe,
    - fioul ;
- un moyen de production hydraulique de type lac ;
- une STEP (Station de Transfert d'Energie par Pompage) ;
- une batterie.

Dans toute la suite, nous utilisons les variables suivantes décrites dans le tableau suivant.

Indices	**Variables**
t : pas de temps g : centrale thermique h : centrale hydraulique **Exposants** th : moyen de production thermique hy : moyen de production hydraulique res : moyen de production renouvelable	$P_{g,t}^{th}$: production des centrales thermiques P_t^{hy} : production de la centrale hydraulique P_t^{res} : production de la centrale éolienne $\mathrm{UC}_{g,t}^{\mathrm{th}}$: binaire indiquant si la centrale thermique g est en fonctionnement ou à l'arrêt à l'instant t $\mathrm{UP}_{g,t}^{\mathrm{th}}$: binaire indiquant si la centrale thermique g est démarrée à l'instant t $DO_{g,t}^{\mathrm{th}}$: binaire indiquant si la centrale thermique g est arrêtée à l'instant t

1. Introduction des variables d'énergie non fournie et d'énergie en excès

Dans ce problème d'EOD, nous introduisons deux nouvelles variables : P_t^{uns} et P_t^{exc} qui représentent respectivement l'énergie non fournie (unsupplied energy) et l'énergie en excès (spilled energy) à chaque pas de temps t. En effet, dans certaines situations, il est possible de ne pas réussir à satisfaire la demande (par exemple en cas de panne) ou bien d'avoir trop d'énergie (par exemple trop d'énergies renouvelables ou de centrales ne pouvant s'arrêter). Ces situations ont un coût car ce sont des situations que l'on souhaite éviter : en particulier, une situation d'énergie non fournie peut coûter très cher. Ainsi, ces deux nouvelles variables apparaissent dans la fonction objectif et dans la contrainte d'équilibre offre-demande.

$$\min \sum_{t=1}^{T} \left(\sum_{h=1}^{N_{hy}} c_{h,t}^{hy} P_{h,t}^{hy} + \sum_{g=1}^{N_{th}} c_{g,t} P_{g,t}^{th} + c^{uns} P_t^{uns} + c^{exc} P_t^{exc} \right)$$

SOUS CONTRAINTES

Equilibre offre-demande :

$$\sum_{g=1}^{N_{th}} P_{g,t}^{th} + \sum_{h=1}^{N_{hy}} P_{h,t}^{hy} + P_t^{res} + P_t^{uns} = load_t + P_t^{exc} \qquad \forall t = 1..T$$

Ce problème est modélisé dans le fichier « eod_1_week_to_complete.jl ».

2. Modélisation de l'hydraulique lac

Jusqu'à présent nous avons modélisé la production hydraulique comme une production pilotable mais sans stock. Or, certains ouvrages hydrauliques comme les centrales « lac » permettent de stocker une certaine quantité d'eau. De manière simplifiée, cela signifie que la quantité d'énergie disponible pendant la semaine est limitée. Ainsi, il faut utiliser l'énergie disponible aux moments les plus judicieux. Dans notre exemple, le réservoir de l'hydraulique lac est équivalent à 80GWh électrique stocké pour la semaine de juillet modélisée.

Remarque : cette valeur de réservoir équivalent est déjà le résultat d'un premier calcul (ici, l'historique, mais cela pourrait être le résultat d'une première phase d'optimisation prenant en compte les contraintes propres du système hydraulique, ou une heuristique).

Exercice : modéliser cette contrainte de stock et l'ajouter dans le problème d'optimisation dans le fichier « eod_1_week_to_complete.jl » ligne 120.

3. Modélisation d'une STEP hebdomadaire

Un autre type d'ouvrage hydraulique, les STEP (stations de pompage turbinage) permettent de stocker de l'énergie. Plus précisément, les STEP sont composées de deux bassins aux altitudes différentes et l'idée est de pomper l'eau du bassin inférieur vers le bassin supérieur lorsque la demande en électricité est faible. A l'inverse, lorsque la demande est élevée, l'eau du bassin supérieur est transférée dans le bassin inférieur pour activer la turbine et générer de l'électricité (comme une centrale hydraulique classique). Dans notre exemple, une STEP hebdomadaire est disponible avec une capacité de pompage/turbinage de 1200 MW et un rendement de 75% pour le pompage.

Exercice : modéliser cette STEP (indice : utiliser trois variables Pcharge_STEP, Pdecharge_STEP et stock_STEP qui indiquent respectivement le pompage de la STEP à un instant t, le turbinage de la STEP à un instant t et le stock disponible à un instant t) et l'ajouter dans le problème d'optimisation dans le fichier « eod_1_week_to_complete.jl » ligne 122. Analyser l'impact de la STEP sur les résultats et sur le coût total

4. Modélisation d'une batterie de quelques heures

Le dernier élément que nous ajoutons est une batterie de quelques heures. En termes de modélisation, une batterie fonctionne comme une STEP, seule la durée de stockage est différente.

Dans notre exemple, nous avons une batterie de 280 MW qui a un rendement de 85% (pour la charge et pour la décharge) et un stock de deux heures.

Exercice : modéliser cette batterie et l'ajouter dans le problème d'optimisation dans le fichier « eod_1_week_to_complete.jl » ligne 125. Analyser l'impact de la batterie sur les résultats et sur le coût total

5. Bonus : modélisation de la cogénération

L'exemple le plus courant pour la cogénération est la production simultanée d'électricité et de chaleur par une turbine à gaz. Ainsi, la production par cogénération a la particularité d'être corrélée à la demande en chaleur, ce qui se traduit par une plage fonctionnement [Pmin, Pmax] qui varie en fonction de la saison.

Dans notre exemple, nous avons une centrale de cogénération d'une capacité de 882MW avec un coût de 70€/MWh avec une plage de fonctionnement de [6%Pmax,14%Pmax] en été et de [40%Pmax,70%Pmax] en hiver.

Exercice : modéliser cette cogénération et l'ajouter dans le problème d'optimisation dans le fichier « eod_1_week_to_complete.jl »