Gabbro

Zeste de savoir

Installer un environnement de développement python avec conda

16 janvier 2019

Table des matières

1. Introduction

Ce tutoriel vous explique comment installer un environnement Python (et en particulier un environnement Python dédié aux sciences) à l'aide de Conda.

Conda est un gestionnaire de paquets binaires multiplateforme. Il permet l'installation et la mise à jour de paquets, notamment de paquets Python.

Conda vous permet de télécharger des programmes et bibliothèques pré-compilées et de les installer sur votre ordinateur, dans le dossier d'installation de Conda, sans besoin d'accès administrateur. De par la pré-compilation, l'installation de programmes ou bibliothèques est très simple (compiler une bibliothèque comme Numpy est un calvaire), mais le nombre de paquets disponibles est restreint. Il peut parfaitement s'utiliser en complément de Pip , un autre gestionnaire de paquets.

Conda permet l'installation de python lui-même, de bibliothèques (numpy, scipy, mais aussi des bibliothèques non scientifiques, comme PyQt...), d'un environnement de développement (abrévié IDE en anglais), comme Spyder et d'un système d'environnements virtuels.

! Si vous êtes sous Linux, il est probablement préférable de passer par votre gestionnaire de paquets.
Conda reste cependant utile si vous n'êtes pas administrateur, si les bibliothèques présentes dans le gestionnaire de paquets de votre distribution sont trop vieilles (ou absentes !) ou bien si vous souhaitez utiliser le très efficace système d'environnements virtuels de Conda.

Conda fonctionne aussi bien avec Python 2 que Python 3.

Pour plus d'informations, n'hésitez pas à lire la documentation en ligne , ou les pages `man` (aussi disponibles en ligne).

Ce tutoriel peut n'être lu qu'en parti. La dernière partie est facultative dans tous les cas. La deuxième l'est aussi si vous choisissez Anaconda. Ce tutoriel n'a pas de pré-requis particulier (outre savoir lire).

2. Installation

Vous pouvez au choix installer tout Anaconda (le gestionnaire de paquet Conda, plus les bibliothèques scientifiques, plus un environnement de développement...) ou le gestionnaire de paquet seulement. Ne faites bien évidemment que l'un des deux.

2.1. Installer tout Anaconda

Allez sur la page de téléchargement de Continuum et sélectionnez la version souhaitée parmi les versions disponibles (Windows, OS X ou Linux, 32 ou 64 bits, Python 2 ou Python 3). Lancez l'exécutable téléchargé (double-clique sous Windows ou OS X, à l'aide de `bash` depuis une console sous Linux) et suivez les instructions d'installation. Si vous avez déjà une version Python d'installée, lisez la section « Un mot sur les versions multiples ».

Vous vous retrouvez avec un interpréteur Python, l'éditeur Spyder, ainsi que de nombreuses bibliothèques. Pour créer un script Python, lancer Spyder et taper le code Python directement dedans.

2.2. Installer Miniconda

Si vous voulez limiter la quantité de choses installées, choisissez Miniconda . Sélectionnez la version souhaitée et lancez l'exécutable téléchargé (double-clique sous Windows ou OS X, à l'aide de `bash` depuis une console sous Linux) et suivez les instructions d'installation.

Les paquets dont vous aurez besoin pour une application scientifique sont *numpy*, *scipy*, *matplotlib*, et les éventuels paquets spécifiques à votre domaine. Voir la partie suivante pour les commandes d'installation.

Après l'installation, tapez dans une console `conda update --all` pour mettre à jour[1].

2.3. Un mot sur les versions multiples

Si vous avez déjà une version de Python d'installée, Conda a dû vous demander lors de l'installation s'il fallait mettre la version de Conda par défaut. Cela consiste à redéfinir le `PATH` pour y ajouter le dossier `bin` de Conda.

1. L'installateur n'a pas besoin d'accès internet ; il installe les paquets embarquées dans le script bash. Ces paquets peuvent cependant être un peu anciens, d'où l'utilité d'une mise à jour.

Ainsi, les scripts lancés en tant qu'utilisateur *à la main*, de même que la commande `python` utiliseront le python de Conda, tandis que les scripts python du système continueront à utiliser le python du système.

Si vous ne le faites pas, alors en lançant la commande `python`, vous arriverez sur le `python` système. Les bibliothèques installées avec Conda ne seront pas accessibles. Il vous faudra explicitement lancer la commande `python` du répertoire d'installation Conda pour avoir accès à ces dernières.

3. Gestion des paquets

Si vous avez choisi Anaconda, vous devez avoir tous les paquets nécessaires à la programmation scientifique en Python d'installé. Ce qui suit sert donc à installer d'autres paquets supplémentaires.

Si vous avez choisi Miniconda, c'est une étape obligatoire.

La documentation de référence sur la gestion des paquets est disponible en ligne. De même pour la liste des paquets disponibles . Vous pouvez aussi obtenir cette liste en faisant `conda search > liste_des_paquets` puis en ouvrant le fichier créé (`liste_des_paquets` contient toutes les versions disponibles de tous les paquets, elle est donc *un peu* longue).

L'installation de paquets se fait en ligne de commande à l'aide de l'exécutable `conda`. Si vous n'avez pas mis Conda par défaut lors de l'installation, il vous faudra lancer ces commandes depuis le répertoire `bin` du dossier d'installation.

Pour installer des paquets, faites

```
conda install noms_paquets
```

et pour mettre à jour des paquets,

```
conda update noms_paquets
```

`noms_paquets` étant les noms des paquets. Il peut s'agir du paquet Conda lui-même.

Pour tout mettre à jour, utilisez l'option `--all` plutôt que de donner un nom de paquet.

Pour supprimer des paquets,

```
conda remove noms_paquets
```

et pour lister tous les paquets installés,

```
conda list
```

Avec ces commandes de base, vous disposez d'un environnement fonctionnel.

4. Gestion des environnements

Les environnements permettent d'installer des paquets différents ou dans des versions différentes. Par exemple, si je veux Python 3 et PyQt 5 et Python 2 et PySide, c'est impossible, car le lien `python` ne peut pas pointer vers `python2` et `python3` en même temps : c'est incompatible.

L'idée est alors de créer deux environnements, d'installer dans l'un certains paquets, et d'autres paquets dans l'autre.

> *i* Il y a dans le dossier d'installation un dossier `envs`. Créer un environnement consiste simplement à créer un dossier dans `envs`, qui contiendra la même structure que le dossier principal (avec un dossier `bin`, `lib`...). Par défaut, une bibliothèque installée dans un environnement est donc inaccessible à python (car présente dans un dossier dans lequel le python ne va pas chercher).
> Lorsque qu'un environnement est activé, la commande `python` est redirigée vers le dossier de l'environnement, un script redéfinit les dossiers dans lesquels python devra chercher des bibliothèques, lui donnant ainsi accès aux bibliothèques de l'environnement (mais lui enlevant l'accès des bibliothèques de conda en dehors de l'environnement pour éviter les incompatibilités).

Pour créer un environnement,

```
conda create --name nom_environnement noms_paquets # forme courte : -n
```

et pour installer un paquet dans un environnement,

```
conda install --name nom_environnement noms_paquets # forme courte : -n
```

Pour l'exemple précédant, il faudrait donc taper les commandes :

```
conda create --name monEnvPyQt5  python=3 pyqt
conda create --name monEnvPySide python=2 pyside
```

Listons les environnements pour nous assurer qu'ils sont créés correctement,

```
conda info --envs # forme courte : -e
```

Reste à activer un environnement. La commande diffère selon le système,

```
source activate nom_environnement # Linux, OS X
activate nom_environnement # Windows
```

et pour sortir d'un environnement et retourner dans l'environnement par défaut,

```
source deactivate # Linux, OS X
deactivate # Windows
```

Voilà ! Avec ces commandes, vous pouvez disposer de plusieurs environnements de développement différents qui cohabitent sans heurt.

5. Conclusion

Ce tutoriel visait à vous donner un moyen simple et efficace d'installer un (voir des !) environnement de programmation pour Python, en particulier dans le contexte de la programmation scientifique. C'est chose faite. Conda permet d'autres choses plus subtiles que celles présentées brièvement ici.

Les liens à garder dans un coin sont les suivants :

- Résumé des commandes en deux pages ;
- Une introduction à Conda ;
- La documentation ;
- Les pages `man` ;
- La liste des paquets .

Tout ceci a pour but de vous permettre de coder dans de bonnes conditions et sans prise de tête ; maintenant, à l'abordage !

6. Remerciements

Merci à banco29510 , Vayel et Karnaj pour leurs relectures lors de la bêta, et adri1 pour la validation.