

# Démarrer avec Python

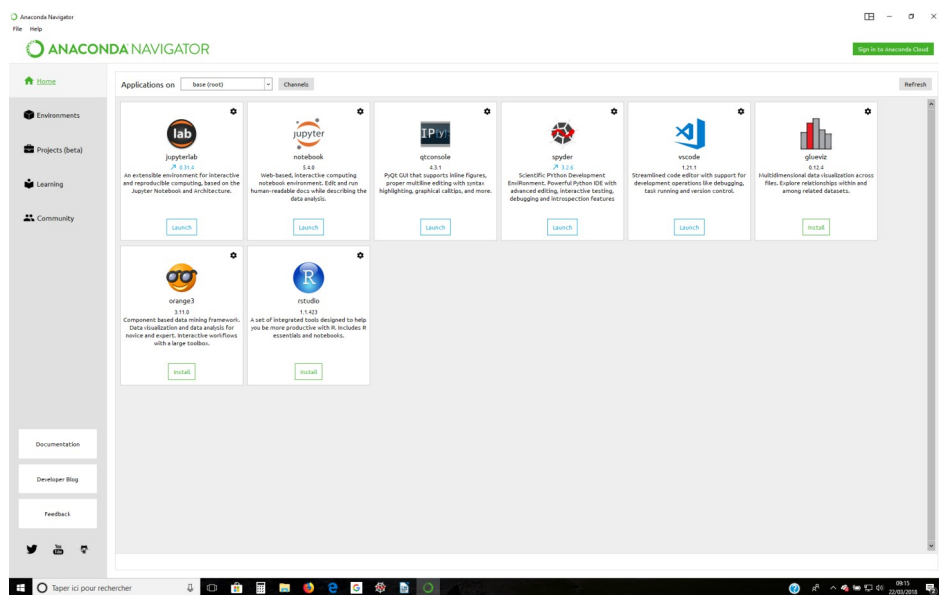
## Installation- démarrage :

Télécharger Anaconda avec Python version 3.6 sur le site :

<https://www.anaconda.com/download/>

Anaconda est une distribution Python libre qui intègre directement lors de l'installation un grand nombre de modules. Si vous souhaitez en installer d'autres, il faudra utiliser le gestionnaire de modules (packages) Conda : <https://conda.io/docs/>

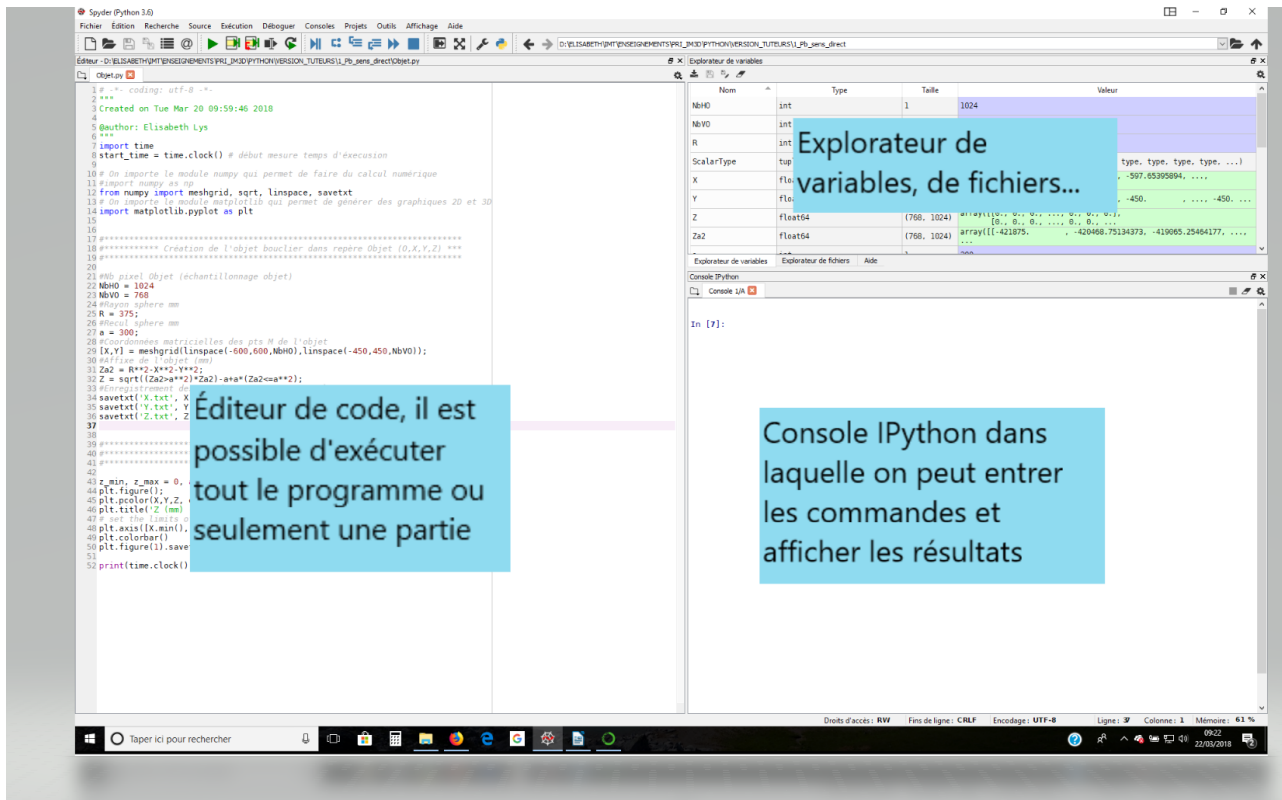
Lorsque l'installation est terminée, vous lancerez l'interface de navigation Anaconda, elle permet de faire les mise à jour des différentes applications disponibles.



*Interface Anaconda*

Pour développer les programmes, nous utiliserons l'interface de développement Spyder





Interface Spyder

## Les commandes magiques dans la console Ipython :

Pour afficher la liste des commandes magiques taper : `%lsmagic` dans la fenêtre de commande

A retenir : `%clear` ou bien `Ctrl+L` permet de nettoyer la console

Pour plus d'information cf :

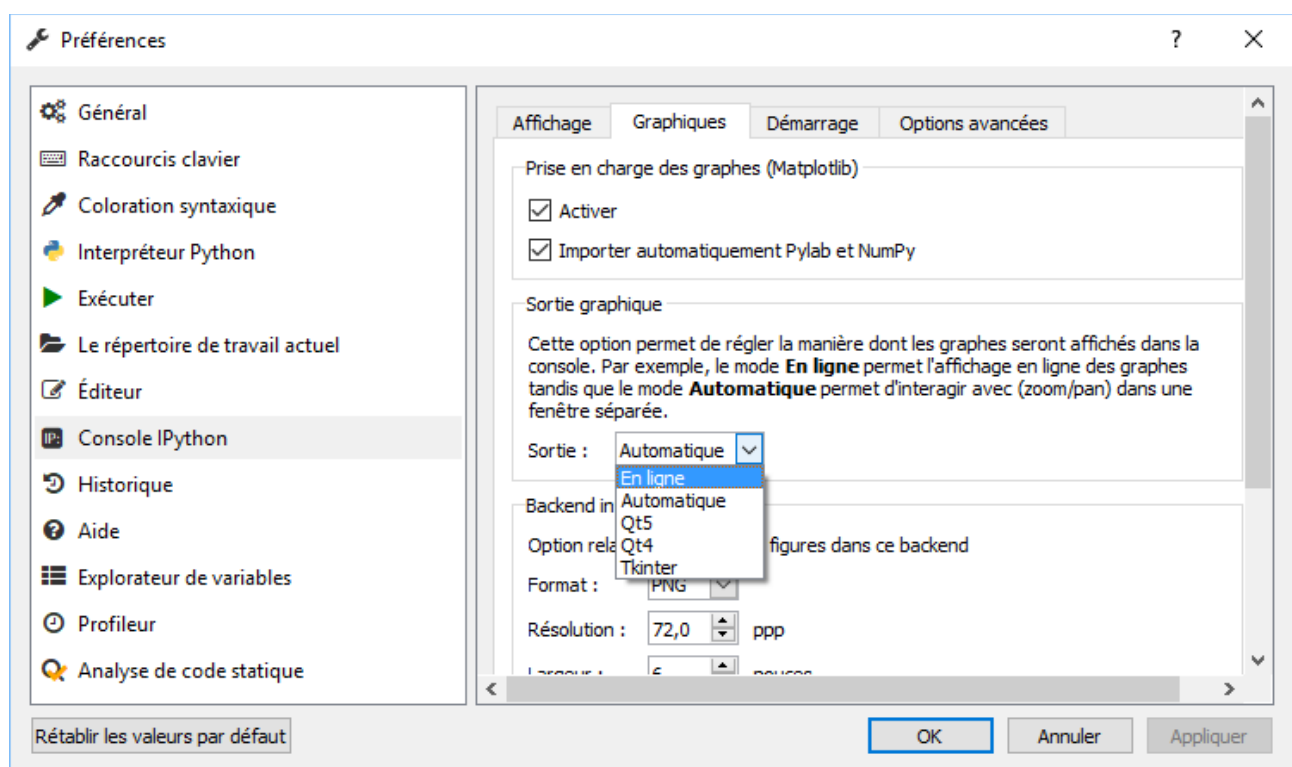
<http://nbviewer.jupyter.org/github/ipython/ipython/blob/master/examples/IPython%20Kernel/Cell%20Magics.ipynb>

## Affichage des figures

Lorsque vous lancerez vos programmes, par défaut les figures s'affichent dans la console Ipython. Pour que les figures s'affichent dans une nouvelle fenêtre il faut modifier les préférences graphiques de la console en allant dans le menu : Outils> Préférences puis « Console IPython » sélectionner ensuite l'onglet « Graphique ».

Par défaut la « Sortie graphique » est « En ligne ». Il faut mettre « Automatique » puis appliquer.

Pour que les modifications soit appliquées, vous devrez fermer la console Ipython courante et en redémarrer une nouvelle.



## Quelques fonctions utiles :

Pour afficher la liste des commandes magiques taper : %lsmagic dans la fenêtre de commande Ipython

Dans l'éditeur de code :

<p><b>Opérations sur les matrices</b></p> <p>- <u>Concaténation</u> : <b>Module numpy</b>  a = array([1,2,3])  a = array([1,2,3])  c = concatenate((a,b), axis=0)  axis=0 → ligne  axis=1 → colonne</p> <p>- <u>Multiplication</u> : <b>Module numpy</b>  a = array([[1,2],[2,3]])  b = array([[0,1],[2,3]])  c = a.dot(b)</p> <p>- <u>Inverse</u> : <b>Module pylab</b>  inv(c) ou bien <b>Module numpy</b>  linalg.inv(c)</p> <p><u>Pseudo inverse</u> : <b>Module numpy</b>  linalg.pinv()</p> <p><u>Matrice transposée</u> : <b>Module numpy</b>  transpose(array([[1, 2, 3]]))</p>	<p><b>Mémorisation</b></p> <p><b>Module numpy</b>  savetxt  loadtxt</p> <p><b>Module skimage</b>  imsave</p> <p><b>Libération de mémoire</b>  a = None</p>	<p><b>Gestion affichage</b></p> <p><b>Module matplotlib.pyplot</b>  figure  pcolor  imshow  plot_surface  scatter  xlabel, ylabel, title  colorbar  colormap</p>
<p><b>Format de données</b></p> <p><u>Nombre de ligne</u> : len(a) ou bien  size(a,axis=0)  <u>Nombre de colonne</u> : size(a,axis=1)</p> <p>shape (a) → Nb Ligne colonne  d'une matrice</p>	<p><b>Déclaration de données</b></p> <p><b>Module numpy</b>  meshgrid,  linspace,  zeros,  ones</p>	<p><b>Opération sur les données</b></p> <p><b>Module numpy</b>  sqrt,  linalg.norm</p> <p><b>Infos sur les données</b></p> <p><b>Module numpy</b>  a.max()  a.min()</p>
<p><b>Interpolation données</b></p> <p>- map_coordinates : <b>Module scipy.ndimage</b>  - interpolate.griddata : <b>Module scipy</b></p>	<p><b>Résolution numérique</b></p> <p>fmin : <b>Module scipy.optimize</b></p>	<p><b>Opération logique</b></p> <p>&lt;, &gt;, ≤, ≥  ==, !=</p>