Modelling and Optimization with Python (Anaconda)

New to modelling and programming? Python is free and easy to learn if you know where to start! This guide will help you to get started quickly.

Anaconda for Data Scientists

Anaconda is a free and open-source distribution of the programming languages Python and R for scientific computing, which aims to simplify package management and deployment. Package versions are managed by the package management system `conda`.

Installing Python on Windows with Anaconda

Before you can do any Python programming and/or using models written in Python you need to install the Python3 interpreter on your computer.

Make sure to download the latest version of Anaconda. Follow the link below to find easy installation instructions for windows:

https://docs.anaconda.com/anaconda/install/windows/

You can confirm that Anaconda is installed and working with Anaconda Navigator or with the `conda` command in the command line. Using Windows, click `Start` and select Anaconda Navigator from the menu. If you prefer using a command line interface (CLI), you can use the `conda` command to verify the installation using Anaconda Prompt on Windows.

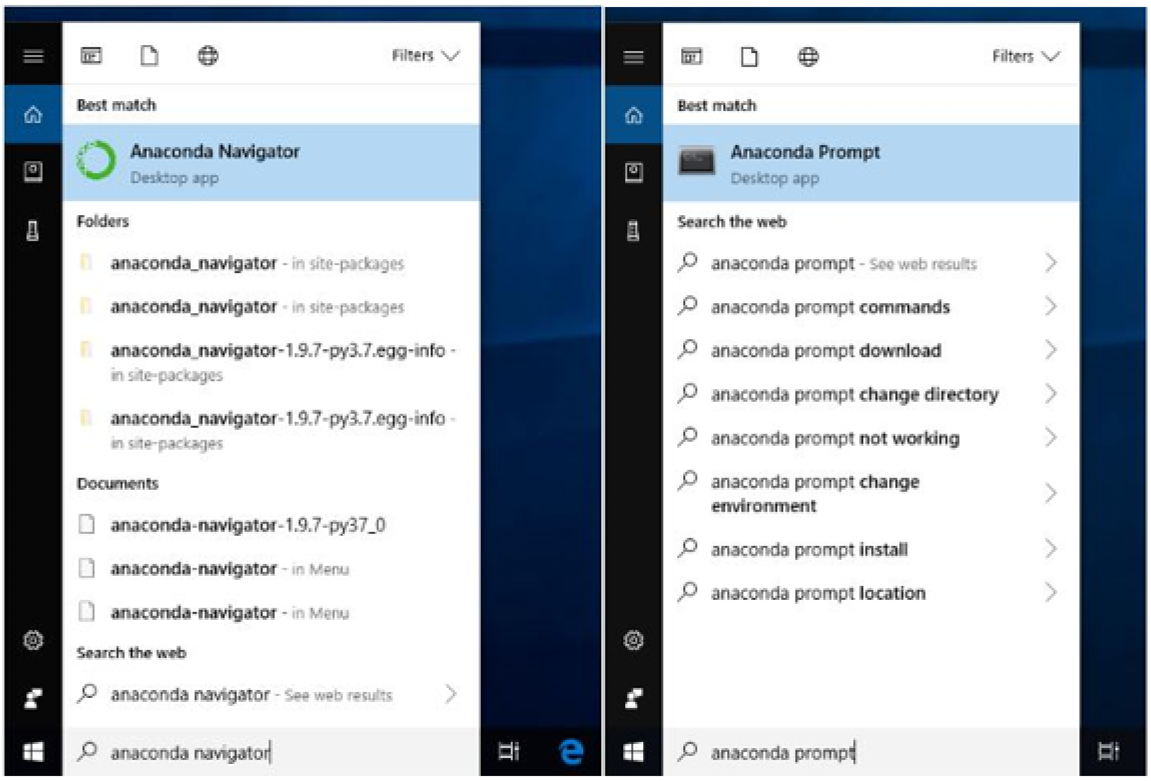
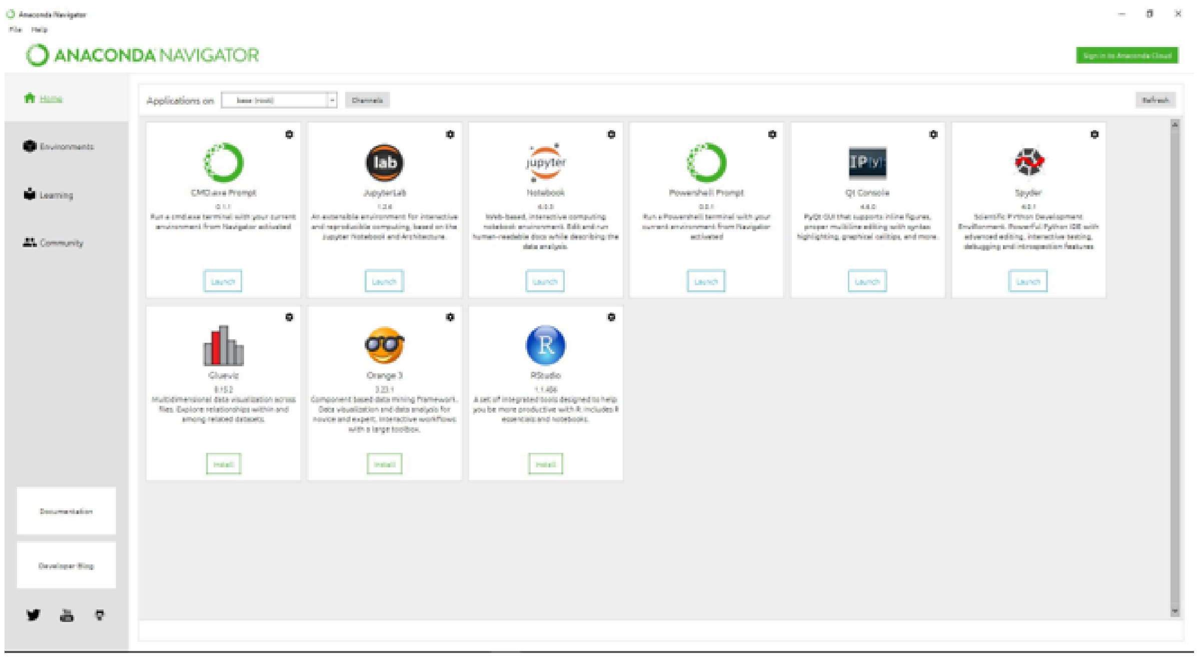

Filters
Best match
Anaconda Navigator
Desktop app
Folders
anaconda_navigator - in site-packages
anaconda_navigator - in site-packages
anaconda_navigator-1.9.7-py3.7.egg-info - in site-packages
anaconda_navigator-1.9.7-py3.7.egg-info - in site-packages
Documents
anaconda-navigator-1.9.7-py37_0
anaconda-navigator - in Menu
anaconda-navigator - in Menu
Search the web
anaconda navigator - See web results
anaconda navigator
Filters
Best match
Anaconda Prompt
Desktop app
Search the web
anaconda prompt - See web results
anaconda prompt commands
anaconda prompt download
anaconda prompt change directory
anaconda prompt not working
anaconda prompt change environment
anaconda prompt install
anaconda prompt location
anaconda prompt

Figure 1: Find Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in the Anaconda distribution, which allows you to launch applications and easily manage conda packages, environments, and channels without using command line commands. With the Navigator you can search for packages on Anaconda Cloud or in a local Anaconda Repository.



Run a cmd.exe terminal with your current environment from Navigator activated

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Run a Powershell terminal with your current environment from Navigator activated

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific Python Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Component based data mining Framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Figure 2: Conda Navigator

Although Anaconda can be managed from the Anaconda Navigator, it is highly recommended using the command line tool. Especially when installing a solver (see below) the command line will be essential. Therefore, the following description is given for command line usage.

Creating a virtual environment with Anaconda

Anaconda uses so-called *virtual environments* to manage your Python ????. These *environments* represent separate spaces??? on your computer, which are used to install specific Python packages. You can create multiple *environments* in which you can use different versions of Python, versions of packages etc. depending the respective application.

Once you have installed Anaconda successfully you can start. For this, you have to open your shell????:

1. On Windows click `Start`
2. In the Search or Run line, type `cmd` (short for command) or `anaconda Prompt` and press Enter
3. Write `python` to show which version of Python and Anaconda is used.

After following 1-2-3, the command line should show:

To create an environment (with the name *example*), write in the Anaconda prompt:

```
conda create -n example python=3.7.6
```

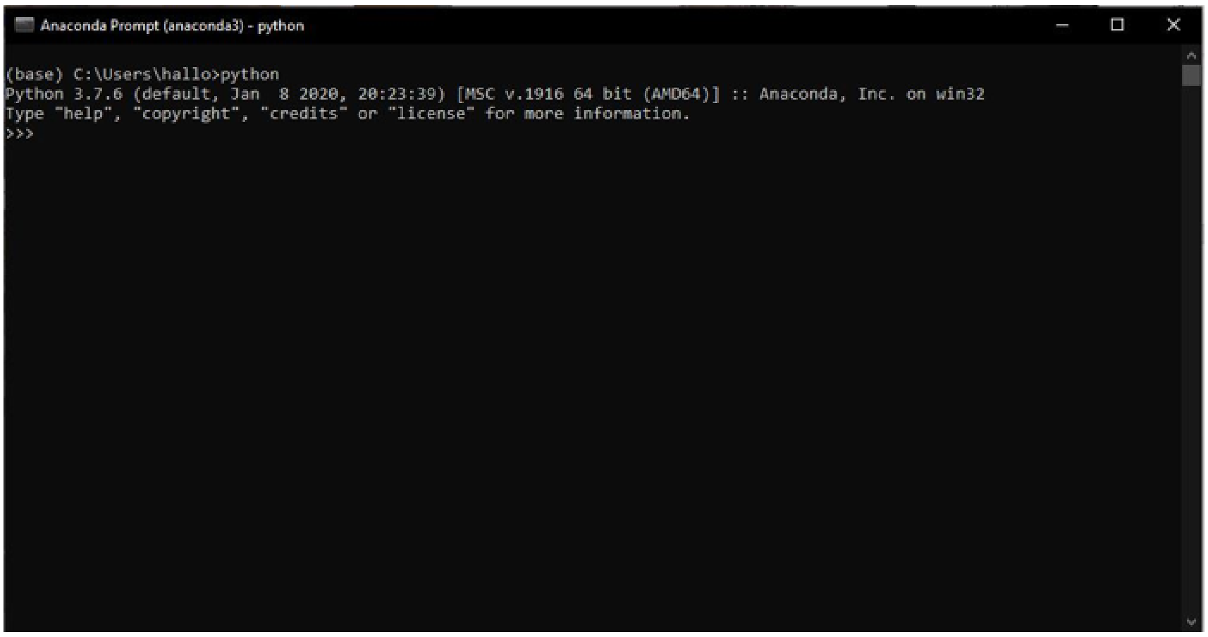
Following instructions, including typing `y` (for yes, proceed with download) and pressing Enter will create an environment with the name example in the Python version 3.7.6. As shown below, before installing any packages into your environment, you will need to activate your virtual environment in Windows:

```
conda activate example
```

To deactivate use:

```
conda deactivate
```

The example below shows how to activate and deactivate an environment (*example* environment).



```
(base) C:\Users\hallo>python
Python 3.7.6 (default, Jan  8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)] :: Anaconda, Inc. on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Figure 3: github

Installing packages and solvers

Packages

Packages are software libraries written for the Python programming language, which can be utilized in a number of contexts including data processing and analysis, writing and reading data as well as solving optimisation problems.

To install any package (here as an example the package *pyomo*) write in the Anaconda command prompt

```
conda install -c conda-forge pyomo
```

This installs from the channel conda-forge the package *pyomo*.

Solvers

Solvers are powerful software libraries to solve large-scale linear programming, quadratic programming, quadratically constrained programming, mixed integer programming and other problems.

To install the (proprietary) gurobi solver run:

```
conda install gurobi
```

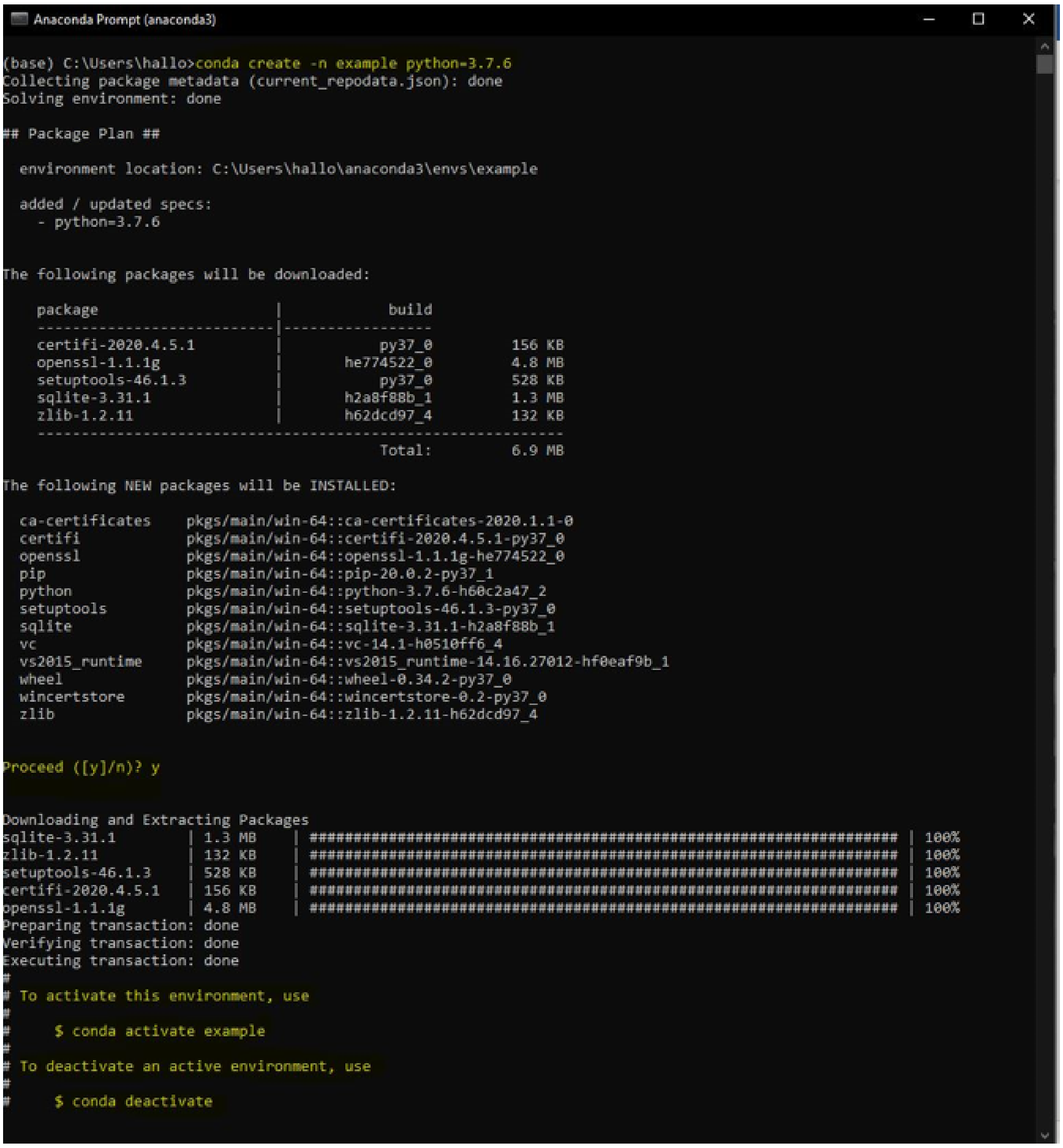
One open source solver is the GLPK solver, which you can install with:

```
conda install -c conda-forge glpk
```

There are various commercial and open-source solvers that can be used.

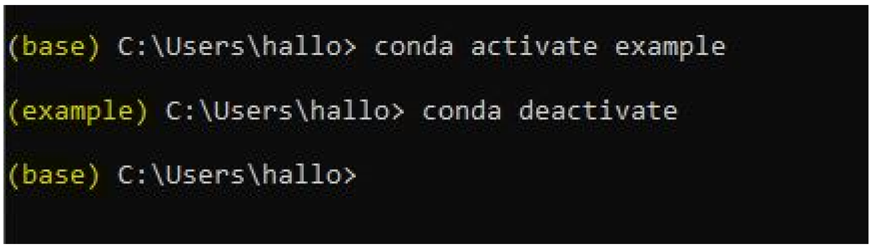
One of the recommended open source solvers is the CBC (Coin-or branch and cut) solver. But sometimes its worth comparing the results of different solvers (e.g. GLPK).

1. Downloaded CBC from here (64 or 32 bit)
2. Download GLPK from
3. Unpacked CBC/GLPK to any folder (e.g. C:/Users/Somebody/my_programs)
4. Add the path of the executable files of both solvers to the PATH variable using this tutorial
5. Restart Windows



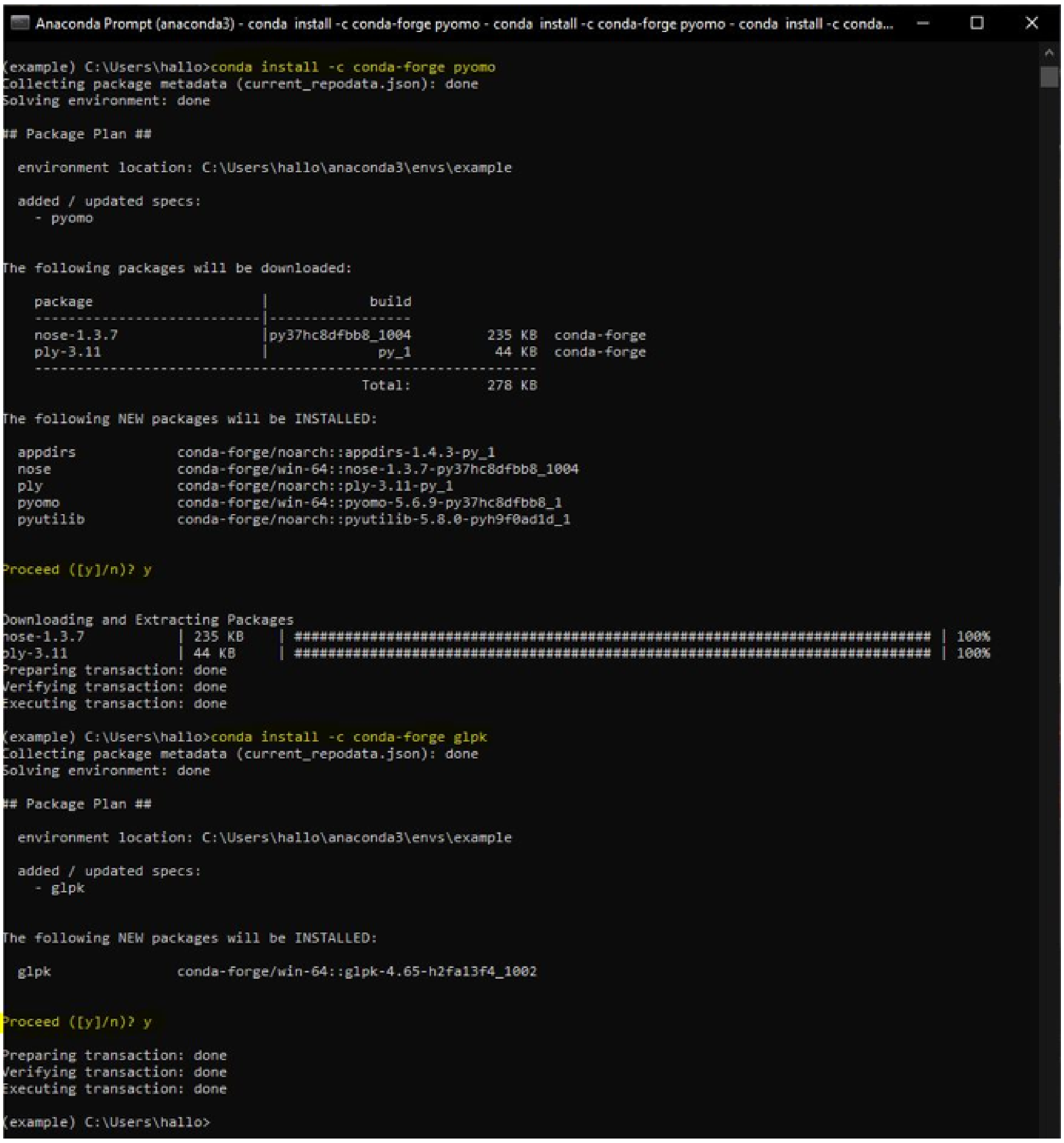
```
(base) C:\Users\hallo>conda create -n example python=3.7.6
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\hallo\anaconda3\envs\example

  added / updated specs:
    - python=3.7.6


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    certifi-2020.4.5.1         |           py37_0         156 KB
    openssl-1.1.1g             |       he774522_0         4.8 MB
    setuptools-46.1.3          |           py37_0         528 KB
    sqlite-3.31.1              |       h2a8f88b_1         1.3 MB
    zlib-1.2.11                |       h62dcd97_4         132 KB
    ------------------------------------------------------------
                                           Total:         6.9 MB

The following NEW packages will be INSTALLED:

  ca-certificates    pkgs/main/win-64::ca-certificates-2020.1.1-0
  certifi            pkgs/main/win-64::certifi-2020.4.5.1-py37_0
  openssl            pkgs/main/win-64::openssl-1.1.1g-he774522_0
  pip                pkgs/main/win-64::pip-20.0.2-py37_1
  python             pkgs/main/win-64::python-3.7.6-h60c2a47_2
  setuptools         pkgs/main/win-64::setuptools-46.1.3-py37_0
  sqlite             pkgs/main/win-64::sqlite-3.31.1-h2a8f88b_1
  vc                 pkgs/main/win-64::vc-14.1-h0510ff6_4
  vs2015_runtime     pkgs/main/win-64::vs2015_runtime-14.16.27012-hf0eaf9b_1
  wheel              pkgs/main/win-64::wheel-0.34.2-py37_0
  wincertstore       pkgs/main/win-64::wincertstore-0.2-py37_0
  zlib               pkgs/main/win-64::zlib-1.2.11-h62dcd97_4


Proceed ([y]/n)? y


Downloading and Extracting Packages
sqlite-3.31.1        | 1.3 MB    | ################################################################### | 100%
zlib-1.2.11          | 132 KB    | ################################################################### | 100%
setuptools-46.1.3    | 528 KB    | ################################################################### | 100%
certifi-2020.4.5.1   | 156 KB    | ################################################################### | 100%
openssl-1.1.1g       | 4.8 MB    | ################################################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate example
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

Figure 4: github



```
(base) C:\Users\hallo> conda activate example

(example) C:\Users\hallo> conda deactivate

(base) C:\Users\hallo>
```

Figure 5: github


Anaconda Prompt (anaconda3) - conda install -c conda-forge pyomo - conda install -c conda-forge pyomo - conda install -c conda...

```
(example) C:\Users\hallo>conda install -c conda-forge pyomo
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\hallo\anaconda3\envs\example

  added / updated specs:
    - pyomo


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    nose-1.3.7                 |py37hc8dfbb8_1004         235 KB  conda-forge
    ply-3.11                   |             py_1          44 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         278 KB

The following NEW packages will be INSTALLED:

  appdirs            conda-forge/noarch::appdirs-1.4.3-py_1
  nose               conda-forge/win-64::nose-1.3.7-py37hc8dfbb8_1004
  ply                conda-forge/noarch::ply-3.11-py_1
  pyomo              conda-forge/win-64::pyomo-5.6.9-py37hc8dfbb8_1
  pyutilib           conda-forge/noarch::pyutilib-5.8.0-pyh9f0ad1d_1


Proceed ([y]/n)? y


Downloading and Extracting Packages
nose-1.3.7           | 235 KB    | ############################################################################ | 100%
ply-3.11             | 44 KB     | ############################################################################ | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(example) C:\Users\hallo>conda install -c conda-forge glpk
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\hallo\anaconda3\envs\example

  added / updated specs:
    - glpk


The following NEW packages will be INSTALLED:

  glpk               conda-forge/win-64::glpk-4.65-h2fa13f4_1002


Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(example) C:\Users\hallo>
```

Figure 6: github

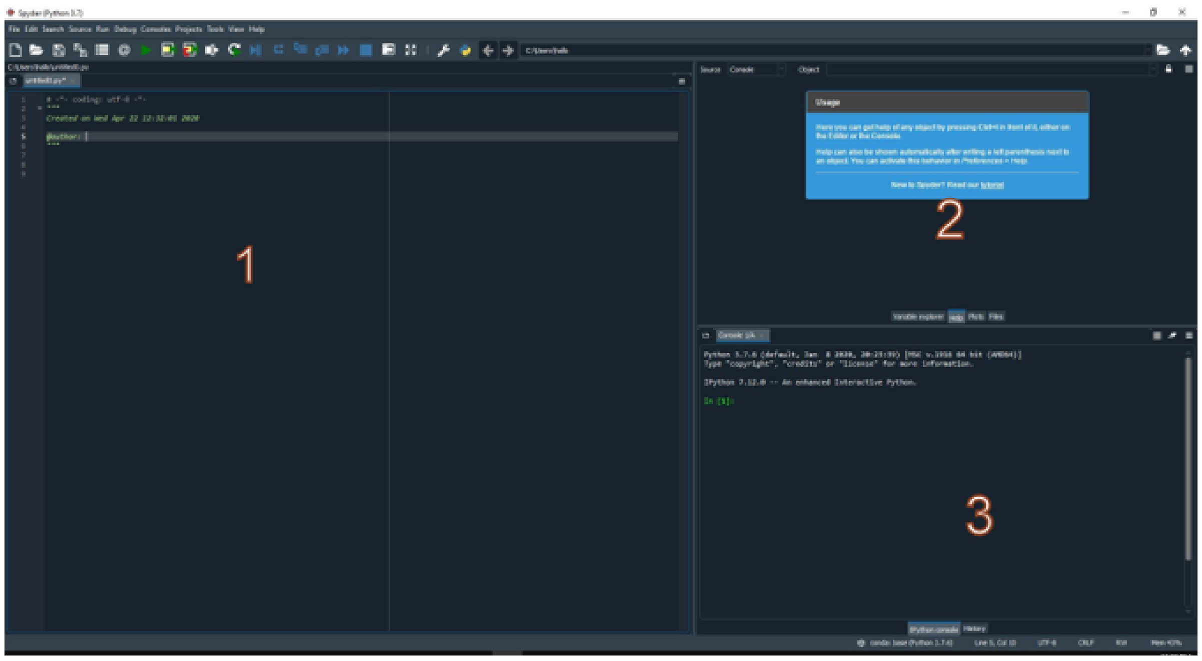
Writing Python Code: *Spyder*

The simplest way to run, edit or write code is by using *Spyder*. *Spyder* is an interactive programming environment for Python. There are many ways to install and launch it. If you are using the Anaconda Navigator, you simply can click the *Spyder* icon.

Alternatively you can open a terminal window and simply launch *Spyder* by typing `spyder` and pressing enter.

You may get a pop-up window saying that spyder is not the latest version, which is just because the version within Anaconda is a few revisions behind.

Spyder


Spyder (Python 3.7)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\holb
C:\Users\holb\untitled0.py
untitled0.py*
Source Console Object
Usage
Variable explorer Help Plots Files
Console 1/A
IPython console History
1
2
3

```
# -*- coding: utf-8 -*-
"""
Created on Wed Apr 22 12:32:01 2020

@author: 
"""
```

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

New to Spyder? Read our tutorial

```
Python 3.7.6 (default, Jan  8 2020, 20:23:39) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.12.0 -- An enhanced Interactive Python.

In [1]:
```

Figure 7: github

1. In the Editor window you can open, read, run, write, edit, etc. your code.
2. In the upper right-hand you can chose between a help browser, a variable explorer or a file explorer. You can change which panels are visible and their layout within the window.
3. In the Python console window you can see the result of your program as well as the history.

You can start working with spyder immediately in the console window. Essentially, it works like Python inside the command line. The big difference is that spyder can inspect the contents of the Python engine and e.g. display variables and their contents within the variable explorer.