Anaconda as a Python environment

Daniel: daniel.andreasen@astro.up.pt
&
João: joao.faria@astro.up.pt

IA-Porto

10th of December, 2015

Anaconda

Anaconda is a freemium distribution of the Python programming language for large-scale data processing, predictive analytics, and scientific computing, that aims to simplify package management and deployment Its package management system is conda
– Wikipedia

What will I get?

- Python 2 or 3

What will I get?

- Python 2 or 3
- A lot of standard scientific packages like numpy, scipy, matplotlib, etc.

What will I get?

- Python 2 or 3
- A lot of standard scientific packages like numpy, scipy, matplotlib, etc.
- Easy access to a ton of packages!

Why should I care?

- Installation of packages without anaconda and/or pip is a pain

Why should I care?

- Installation of packages without anaconda and/or pip is a pain
- Installation of specific version of packages is even more painful

Why should I care?

- Installation of packages without anaconda and/or pip is a pain
- Installation of specific version of packages is even more painful
- With anaconda and pip it is easy and safe

Installation

- Download the right version for you on their web page https://www.continuum.io/downloads

Installation

- Download the right version for you on their web page https://www.continuum.io/downloads
- Follow instruction from the same web page and during the installation

Setup (on *nix based system)

- Setup the path in .bashrc:
  export PATH=$PATH:/home/<username>/anaconda3/bin
- Update: source ~/.bashrc
- At this point you have a working environment (default: root)
- If you are collaborating on projects with others, a new environment might be the best way to go

Setting up an environment

- Create an environment:
  conda create —n astro python astropy

Setting up an environment

- Create an environment:
  conda create —n astro python astropy
- List all environments: conda env list

Setting up an environment

- Create an environment:
  conda create —n astro python astropy
- List all environments: conda env list
- Switch to new environment: source activate astro

Install packages - with conda

- You can use both pip and conda. If not in conda, then use pip

Install packages - with conda

- You can use both pip and conda. If not in conda, then use pip
- Say you want astropy 0.4.3:

Install packages - with conda

- You can use both pip and conda. If not in conda, then use pip
- Say you want astropy 0.4.3:
  - source activate astro
  - conda search astropy
  - conda install astropy=0.4.3

Install packages - with conda

- You can use both pip and conda. If not in conda, then use pip
- Say you want astropy 0.4.3:
  - source activate astro
  - conda search astropy
  - conda install astropy=0.4.3
  - This will automatically update packages to resolve conflicts!

Install packages - with pip

- You can't find the package you want with conda, so pip to the rescue

Install packages - with pip

- You can't find the package you want with conda, so pip to the rescue
- Say you want django 1.2.7

Install packages - with pip

- You can't find the package you want with conda, so pip to the rescue
- Say you want django 1.2.7
  - source activate astro
  - pip install django==1.2.7

Install packages - with pip

- You can't find the package you want with conda, so pip to the rescue
- Say you want django 1.2.7
  - source activate astro
  - pip install django==1.2.7
  - To search available versions:
    - pip install yolk
    - yolk −V django

Bonus slides: Anaconda cheat sheet - 1

- **Most used commands (all starting with conda)**
  - search <pkgname>: Search for a given package and get all available versions
  - install <pkgname>: Install a package
  - install <pkgname>=x.y.z: Install a package with specific version
  - create -n <envname> <pkgname(s)>: Create a new environment, and install a bunch of packages already
  - source activate <envname>: Change to <envname> (without conda in front!)
  - list: List installed packages

Bonus slides: Anaconda cheat sheet - 2

- **Most used commands (all starting with conda)**
  - list <text>: List installed packages containing <text>
  - uninstall/remove <pkgname>: Remove a package (uninstall and remove are the same)
  - clean: Must be followed by one or multiple: –lock, –tarballs, –index-cache, –packages, –source-cache
  - update <pkgname>: Update <pkgname> to latest version
  - update –all: Update all installed packages in the environment

Bonus slides: pip cheat sheet

- **Most used commands (all starting with pip)**
  - search <name>: Search for packages with name, e.g. astro
  - install <pkgname>: Install <pkgname>
  - install <pkgname>==x.y.z: Install <pkgname> with specific version (notice the extra =)
  - uninstall <pkgname>: Uninstall <pkgname>