Harnessing the Power of Python in ArcGIS Using the Conda Distribution

Shaun Walbridge Mark Janikas Ting Lee

https://github.com/scw/conda-devsummit-2016-talk

Handout PDF

High Quality PDF (2MB)

Conda

Conda

- Brand new: thought it was more important to show it to you than to focus on telling you about it
- Time today to discuss your needs and what we might do to solve your problems

Why Python?

- Accessible for new-comers, and the most taught first language in US universites
- Extensive package collection (56 thousand on PyPI), broad user-base
- Strong glue language used to bind together many environments, both open source and commercial
- Open source with liberal license — do what you want

Package Management for Python

Why not `pip`, wheels, virtualenvs?

- Don't handle the harder problem of system dependencies, considered out of scope by Python packagers — does it end up in `site-packages`?
- Package devs: On OSX and Linux, 'easy' to get the deps! Use a system package manager (e.g. `apt`, `brew`, `yum`) and the included compiler (e.g. `clang`, `gcc`).
- It's still not easy to make reproducible builds, and what about Windows?

What about Windows?

- We are particularly stuck on Windows which lacks broadly used package management
  - NuGet is great, but not a system-level package manager
  - If managing applications, try Chocolatey
- Only devs have a C compiler on their machine
  - The essential model is compilers for few, runtimes for all
- Package management is hard! (Except on JavaScript – universal compilers are a leg-up)

. . .

- Enter Conda

Why Conda?

- Scientific Python community identified that there was a gap not being addressed by the core Python infrastructure, limiting their ability to get packages into the hands of users
- Industry standard built by people who care about this space – Continuum Analytics

Why Conda?

- It solves a hard problem:
- Handles dependencies for many languages (C, C++, R and of course Python)
- Built for Python first, but it really solves a much broader infrastructural issue.

Figure 1:

Figure 2:

Conda

Conda

- Cross-platform: simply develop recipes for building and installing software on Linux, OS X and Windows. All it takes: a `meta.yaml`, and a build recipe.
- Open source (BSD): Esri is using it, you can use it in your own projects for other contexts

Conda

What can it install? Not just scientific packages. It can help with:

- GUI toolkits (PyQt, TKinter)
- C++ Libraries (Boost)
- IDEs (Spyder, Juptyer)

See conda-recipes for a comprehensive set of build recipes. Everything from applications to compilers to Python modules, hundreds of maintained recipes across many problem domains.

Conda

- Environments: Can isolate a Python environment, flexibly make changes withot affecting installed software.
- Requirements – include explicit state information, not just the package name. Names aren't enough!
- Also handles platforms and Jupyter notebooks

How Does it Work?

Conda packages can come from a variety of locations:

- On disk (`file://`)
- Public repositories hosted on Anaconda Cloud
- Public repositories self-hosted
- Private repositories
- Paid private repositories

Conda Basics

Figure 3:

Command line interface

Will show what we're working on to make this easier, especially for non-developers

Conda Cheatsheet

Conda Basics

To start:

```
conda --help
```

- A collection of packages and Python install is called an *environment* or *env*, the building block for managing Python with Conda
- Can have multiple environments and seamlessly switch between them

Conda Basics

Activating environments, a couple ways:

- Use the shortcuts
- Manually activate the environment:

```
cd /d C:\ArcGIS\bin\Python\Scripts
activate arcgispro-py3
```

Conda Basics

Once you're in an environment get details with `info`:

```
conda info
```

Conda info is the starting point – it tells you the state of the environment.

Conda Basics

```
conda info
```

```
Current conda install:

             platform : win-64
        conda version : 4.0.4
  conda-build version : not installed
       python version : 3.5.1.final.0
     requests version : 2.9.1
     root environment : C:\ArcGIS\bin\Python  (writable)
```

```
  default environment : C:\ArcGIS\bin\Python\envs\arcgispro-py3
     envs directories : C:\ArcGIS\bin\Python\envs
        package cache : C:\ArcGIS\bin\Python\pkgs
         channel URLs : https://conda.anaconda.org/esri/win-64/
                        https://conda.anaconda.org/esri/noarch/
                    https://repo.continuum.io/pkgs/free/win-64/
                    https://repo.continuum.io/pkgs/free/noarch/
          config file : C:\ArcGIS\bin\Python\.condarc
```

Conda Basics

```
conda list
```

```
# packages in environment at C:\ArcGIS\bin\Python\envs\arcgispro-py3:
#
arcgispro                 1.0                          0    esri
matplotlib              1.4.3             np19py34_0    defaults
nose                    1.3.7                 py34_0    defaults
numpy                 1.9.3             py34_0e [arcgispro] esri
pandas                  0.17.1              np19py34_0    esri
pip                     8.0.3                 py34_0    defaults
pyparsing               2.0.3                 py34_0    defaults
pypdf2                    1.25.1                   py_0    esri
python                  3.4.4                      2    defaults
python-dateutil          2.4.2                 py34_0    defaults
pytz                    2015.7                 py34_0    defaults
scipy                0.16.1          np19py34_0e [arcgispro] esri
setuptools              20.1.1                 py34_0    defaults
six                     1.10.0                 py34_0    defaults
sympy                   0.7.6.1                py34_0    defaults
wheel                   0.29.0                 py34_0    defaults
xlrd                    0.9.4                  py34_0    defaults
xlwt                    1.0.0                  py34_0    defaults
```

Conda Basics

Creating new environments:

- A few different ways. Can manually specify the dependencies:

```
conda create --name my_env python=3.4 numpy flask dask
```

- Can also use a file which includes all the dependencies:

```
conda create --name my_env --file my_sweet_depends.txt
```

  These can contain explcit information about channels, to ensure that the new environment precisely matches the requirements.

Conda vs…

Name	Means	Will Ship?
Conda	The command itself	☐
Miniconda	A minimum set of Python packages to build and run Conda.	☐
Anaconda	A distribution 200+ packages built with Conda	
Anaconda Server	Host the full infrastructure internally	

Conda Demo

Deeper Dive

Conda Behind Firewall

- How's it work?
- Lock it down: Don't use network
- Can vet the installation
- Will work out of the box with default packages without any network connectivity

`.condarc`

- Modify defaults with a simple simple YAML file for configuration
- Can be updated with `conda config`, just like using `git config` to update the default configuration

A detailed example `.condarc`

Creating packages

Straightforward:

- A metadata document (`meta.yaml`) specifying the contents and dependencies
- A build command (`bld.bat`, `build.sh`) specifying how to build

Creating packages

`meta.yaml`:

```yaml
package:
  name: pypdf2
  version: "1.25.1"

source:
  fn: PyPDF2-1.25.1.tar.gz
  url: https://pypi.python.org/packages/source/P/PyPDF2/PyPDF2-1.25.1.tar.gz
  md5: ee5e5b01d00b120805e5049e56c6fd7c

requirements:
  run:
    - python
```

Creating packages

`bld.bat`:

```
"%PYTHON%" setup.py install
```

Multiple Pythons

Currently:

Platform	Python version
Desktop	Python 2.7.x (2.7.10)
Pro	Python 3.4.x (3.4.3)

Multiple Pythons

Upgrade code? Python migration for ArcGIS Pro

- Do it already! You can support 2 + 3 without that much work
- If you hit an issue, it's probably because you don't understand Unicode yet – Watch this PyCon talk, *Pragmatic Unicode, or, How do I stop the pain?*

. . .

But… this can be costly. For many organizations, a significant burden, even if the language changes are relatively small.

Multiple Pythons with Conda

With Conda, we can support multiple platforms:

- Py 2.7, 3.4, 3.5 in Pro 1.3

Create a new environment, target a different Python, users can now use that with the Py2 code

Still need to change `arcpy.mapping` to `arcpy.mp` when moving from Desktop to Pro, but no Python language level changes needed.

Challenges

Have to make sure you're running the right Python (*what happens when you type `python` at the command line?*)

- We will make this easy as possible
- It'll be easy to tell in app
- Isolated installation fixes a variety of issues

Requires some user education over the "only one Python on the box" model

What Do I Get Out of the Box?

- Conda command and a Conda root Python install
- New modules (e.g. `requests`)
- Conda environment with all of the ArcGIS Pro dependencies as Conda packages

How can I use this?

- We already ship you the SciPy stack – powerful and out of the box, can use today (Pro and 10.4)
- Can start using `conda` today. Miniconda is fully stand-alone, won't affect your global Python (unless you tell it to)
- Package your work: this is an opportunity to distribute it, possibly including commercial side as well.

Where Can I Run This?

Figure 4:

- ArcGIS Pro 1.3 (Release: 2016 UC)
  - Will be *the* Python install.
  - UI for interaction
- Future:
  - Take advantage of more features
  - Integration with platform

from future import *

Effectively manage complex software dependencies with Conda. Thousands of packages exist today, can integrate it into your organization's needs.

Resources

Resources

Conda Recipes

Anaconda.org

Conda Cheatsheet

Closing

Thanks

Esri Conda Team:

Figure 5:

Continuum Analytics for creating and open sourcing Conda

Rate This Session

iOS, Android: Feedback from within the app

Windows Phone, don't use a smartphone?: Cuniform tablets accepted (sorry! limitation).

. . .

Windows Phone, or no smartphone? Cuneiform tablets accepted.

fin

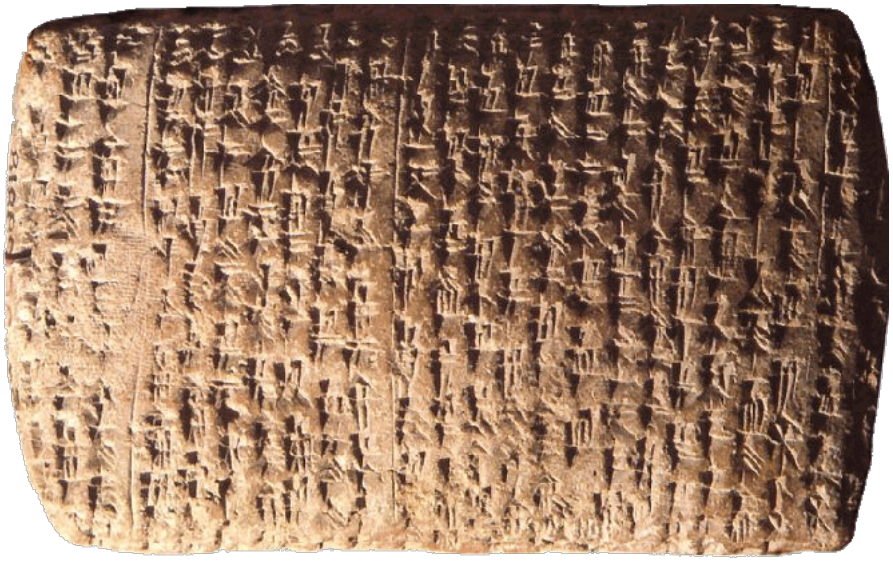

Figure 6: