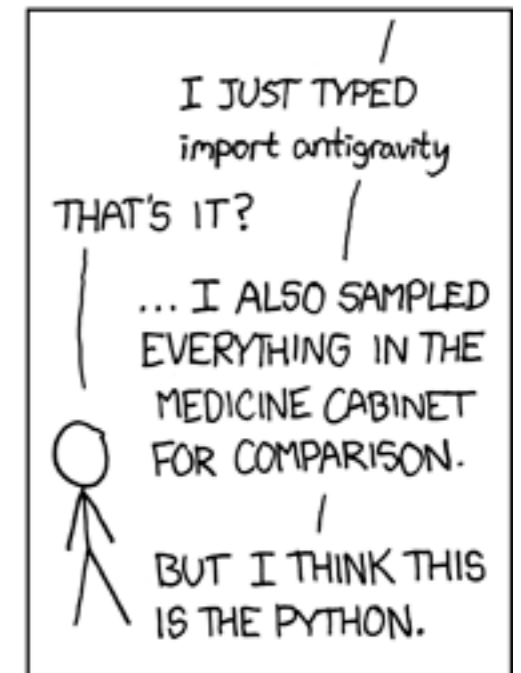
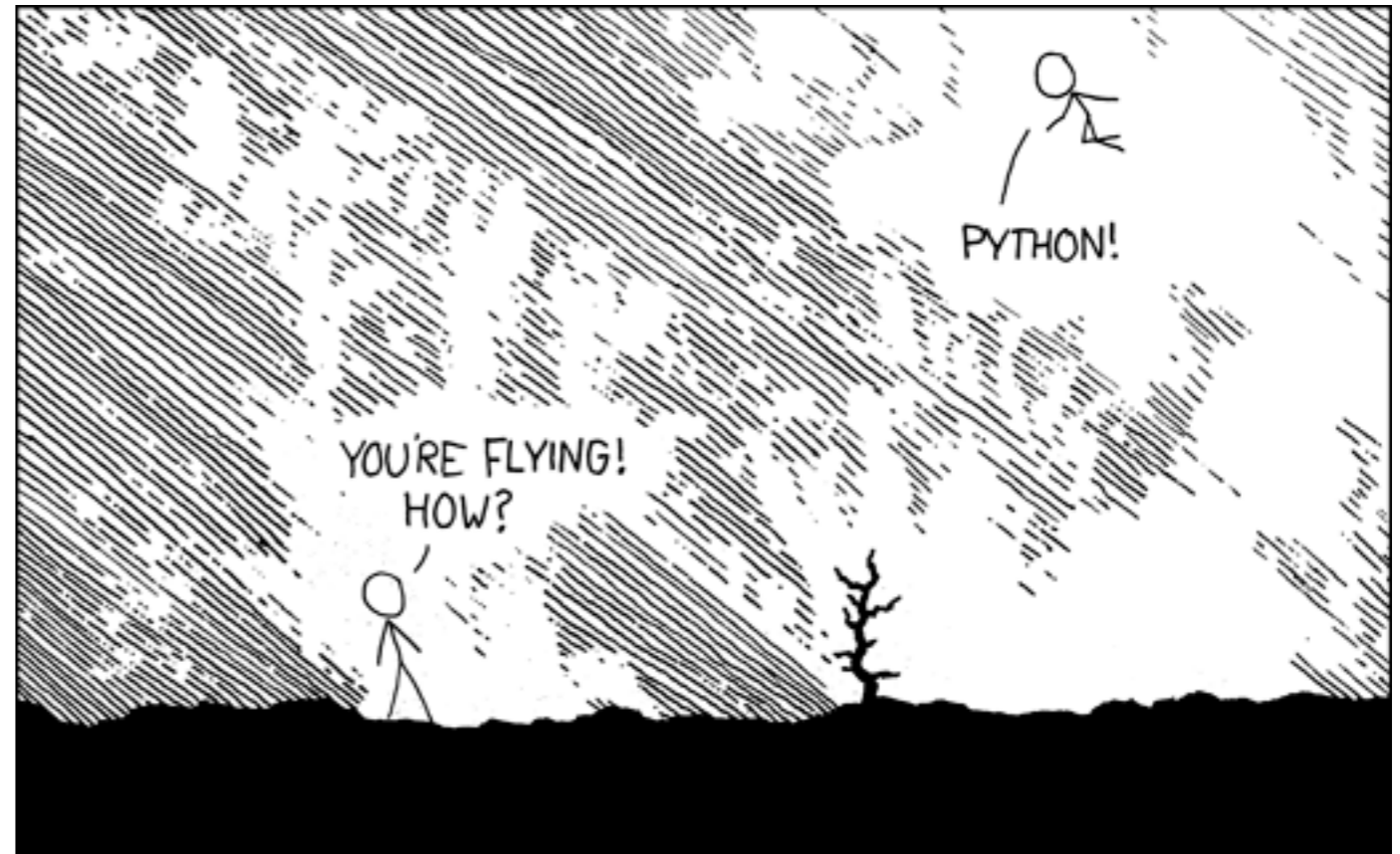


# PyCogent

Greg Caporaso

17 September 2009

gregcaporaso@gmail.com



# PyCogent

- Support for interacting with external databases, many file formats, external applications
- Core objects for common data types (e.g., sequence, alignment, tree)
- Growing user and developer community, with core development taking place at CU Boulder and Australia National University

# Why python?

- Object-oriented language with support for procedural and functional programming
- Easily extensible with (e.g.) Cython to compile slow pieces
- Built-in unittest module, doctest module allow for executable documentation and testing

# Lecture outline: three PyCogent use cases

- Features of PyCogent: building a tree of life
- Applying an ApplicationController object: the RDP Classifier
- Building a new ApplicationController object: formatdb

# PyCogent usage: building a tree of life

See Woese 1987 (*Bacterial Evolution*) and Woese 1990 (*Towards a natural system of organisms*) for discussion of the ideas presented in this example.

# PyCogent usage: building a tree of life

1. Load a collection of archaeal, bacterial, and eukaryotic 16S sequences.
2. Perform a multiple sequence alignment on the sequences.
3. Build a tree from the sequences.
4. Confirm that there are three distinct clusters in the tree, supporting the idea that cellular life on Earth clusters into three domains detectable by distances between 16S sequences.
5. Output a graphic of the tree as a PDF.

# Summary: building a tree of life

## Compile sequences:

I did this externally so the example could be done quickly, but possible with EUtils module (see example on page R171.4 in PyCogent paper)

## Load sequences:

LoadSeqs, SequenceCollection object, DNA

## Align sequences:

The Muscle ApplicationController object

## Build a tree:

The FastTree ApplicationController

The PhyloNode object

## Visualize the tree:

UnrootedDenrogram object, and the cogent drawing functionality

# The PyCogent Application Controller Framework

- Run external application available via PyCogent, and therefore can integrate external tools into your bioinformatics pipeline
- Define common interface for applications

Full documentation at:

[http://pycogent.sourceforge.net/examples/application\\_controller\\_framework.html](http://pycogent.sourceforge.net/examples/application_controller_framework.html)

# Integrate third-party tools into your bioinformatics pipeline

- Example includes steps we ran through in the tree of life application: muscle and fast tree.

# Common interface for third-party multiple sequence aligners

```
caporaso@mentat app> egrep 'def align_unaligned_seqs' *py
```

```
clearcut.py:def align_unaligned_seqs(seqs, moltype, params=None):
```

```
clustalw.py:def align_unaligned_seqs(seqs, moltype, params=None):
```

```
mafft.py:def align_unaligned_seqs(seqs, moltype, params=None, accurate=False):
```

```
muscle.py:def align_unaligned_seqs(seqs, moltype, params=None):
```

```
caporaso@mentat app> egrep 'def build_tree_from_alignment' *py
```

```
clearcut.py:def build_tree_from_alignment(aln, moltype, best_tree=False, params={}, \
```

```
clustalw.py:def build_tree_from_alignment(aln, moltype, best_tree=False, params=None):
```

```
fasttree.py:def build_tree_from_alignment(aln, moltype, best_tree=False, params=None):
```

```
fasttree_v1.py:def build_tree_from_alignment(aln, moltype, best_tree=False, params=None):
```

```
mafft.py:def build_tree_from_alignment(aln, moltype, best_tree=False, params={}, \
```

```
muscle.py:def build_tree_from_alignment(aln, moltype, best_tree=False, params=None):
```

```
raxml.py:def build_tree_from_alignment(aln, moltype, best_tree=False, params={}):
```

# How to control third party applications with PyCogent

- Check the applications which are currently supported. (If yours isn't new application controllers are typically easy to write.)
- The application must be installed and in your search path.

```
caporaso@mentat ~> cd repo/pycogent/cogent/app/
caporaso@mentat app> ls *.py
__init__.py          cove.py              knetfold.py
blast.py             dialign.py           lagan.py
carnac.py            dotur.py             mafft.py
cd_hit.py            dynalign.py          mfold.py
clearcut.py          fasttree.py          muscle.py
clustalw.py          fasttree_v1.py       nupack.py
cmfinder.py          foldalign.py         parameters.py
comrna.py            gctmpca.py           pfold.py
consan.py            ilm.py               pknotsrg.py
contrafold.py        infernal.py          raxml.py
```

# RDP Classifier

- Naive Bayesian classifier to assign taxonomy based on 16S ribosomal RNA sequence
- Frequently applied step in microbial community analysis studies, so convenient to have wrapped in pycogent

# Interacting with the RDP Classifier via the PyCogent ApplicationController

1. Interact directly with the ApplicationController object
2. Interact with the convenience functions
3. Interact with the command line interface

**Defining a new application  
controller: formatdb**

# Learning more about PyCogent

- Python: <http://www.python.org>
- Documentation: <http://pycogent.sourceforge.net>
- Install: <http://pycogent.sourceforge.net/install.html>
- BIOI 7711 students interested in contributing to PyCogent can contact me at [gregcaporaso@gmail.com](mailto:gregcaporaso@gmail.com). (Contributions are peer-reviewed by developers before developer access is granted.) If interested, these pages are for you:
  - [http://pycogent.sourceforge.net/coding\\_guidelines.html](http://pycogent.sourceforge.net/coding_guidelines.html)
  - [http://pycogent.sourceforge.net/developer\\_notes.html](http://pycogent.sourceforge.net/developer_notes.html)