How I am getting Python into a multi-billion $ enterprise
Then and now...

• I started 9 months ago, no Python in use, anywhere
• Now, Python is part of our stack moving forward as we build out our IT and analytics capabilities
• I have played an important role in defining this stack, especially with respect to Python
• I am lucky to work with smart people who are open to new tools and processes
• This talk discusses what I have done over recent months to move from then to now...
800 lbs gorillas...
FUD I have to deal with...

• Lack of commercial support
  • Difficult for IT to deal with standard installation
  • No single throat to choke
  • Analytics: no longer trust issues with open source, but still perception that coverage is weaker (e.g. compared to SAS or R)

• Big data
  • Python only works with data that is in-memory

• Integration with Enterprise systems
  • How do we know we’ll be able to connect Python to our database system (which includes Teradata, Oracle, SQL Server, IBM iSeries...)
PyMSL™ Studio

Prototype to Production Development Environment for Analytic Applications

PyMSL Studio is a collection of development tools and numerical algorithms for modeling and implementation teams creating analytic applications. With PyMSL Studio, prototypes can be transformed into production-ready and with less complexity, cost and risk than using multiple tools. The core component of PyMSL Studio is PyMSL, a set of Python wrappers to the algorithms in the MSL C Library. PyMSL is included with PyMSL Studio and is also available as a separate download. To request the separate download, please contact us at PyMSL.Studio.technquiries@roguewave.com.

Welcome, Guest | Sign in | My Account | Bids | Cart

1.866.631.4581

Need Help? Email Sales

Request Info/Quote

Which Python Edition?

Watch Webinar

ActiveState

The Dynamic Language Experts

SOLUTIONS

PRODUCTS

SUPPORT & SERVICES

PARTNERS

COMMUNITY

RESOURCES

COMPANY

Overview

ActivePerl

ActivePython

PyMSL

ActiveTcl

Language Editions

Developer Bundles

Kernelements

Perl Dev Kit

Tcl Dev Kit

ActivePython Business Edition

Save Time and Ensure Reliability with ActivePython Business and Enterprise Editions

ActivePython Business and Enterprise Editions feature our pre-compiled, supported, quality-assured Python distribution used by millions of developers around the world for easy Python installation and quality-assured code. When you're using Python on production servers or mission-critical applications, ActivePython Business and Enterprise Editions offer significant time savings, over an open source Python for installing, managing, and standardizing your Python.

Want Python on the cloud? Stackato, new from ActiveState, is the first enterprise cloud platform for Python and Perl. Stackato's Cloud Foundry allows enterprises to deploy, manage and scale applications in the private or public cloud. Learn More.

Not sure which edition is right for you? Check our Compare Editions chart.

The Enthought Python Distribution provides a comprehensive, cross-platform environment for scientific computing with the Python programming language. A single-click installer allows immediate access to over 75 libraries and tools. Our open source initiatives include SciPy and the Enthought Tool Suite.

We work with you to develop software for interactive data manipulation and visualization. In addition to full application development, we also offer short-term consultation to address critical technical computing needs. Our areas of expertise include geoscience, financial analysis, 3D modeling, fluid dynamics and microtechnology.

EPD 7.0

Enthought Python Distribution 7.0

DOWNLOAD | PURCHASE

SciPy 2011

JULY 11-16 | AUSTIN, TX

PyCon Au 2011

8.26-21 | SYDNEY, AU

EVENTS 2011
I do a lot of statistical work and use Python as my main language. Some of the data sets I work with though can take 20GB of memory, which makes operating on them using in-memory functions in numpy, scipy, and PyIMSL nearly impossible. The statistical analysis language SAS has a big advantage here in that it can operate on data from hard disk as opposed to strictly in-memory processing. But, I want to avoid having to write a lot of code in SAS (for a variety of reasons) and am therefore trying to determine what options I have with Python (besides buying more hardware and memory).

I should clarify that approaches like map-reduce will not help in much of my work because I need to operate on complete sets of data (e.g. computing quantiles or fitting a logistic regression model).

Recently I started playing with h5py and think it is the best option I have found for allowing Python to act like SAS and operate on data from disk (via hdf5 files), while still being able to leverage numpy/scipy /matplotlib, etc. I would like to hear if anyone has experience using Python and h5py in a similar setting and what they have found. Has anyone been able to use Python in "big data" settings heretofore dominated by SAS?

EDIT: Buying more hardware/memory certainly can help, but from an IT perspective it is hard for me to sell Python to an organization that needs to analyze huge data sets when Python (or R, or MATLAB etc) need to hold data in memory. SAS continues to have a strong selling point here because while disk-based analytics may be slower, you can confidently deal with huge data sets. So, I am hoping that Stackoverflow-ers can help me figure out how to reduce the perceived risk around using Python as a mainstay big-data analytics language.
What is PyTables?

PyTables is a package for managing hierarchical datasets and designed to efficiently and easily cope with extremely large amounts of data. You can download PyTables and use it for free. You can access documentation, some examples of use and presentations in the HowToUse section.

PyTables is built on top of the HDF5 library, using the Python language and the NumPy package. It features an object-oriented interface that, combined with C extensions for the performance-critical parts of the code (generated using Cython), makes it a fast, yet extremely easy to use tool for interactively browse, process and search very large amounts of data. One important feature of PyTables is that it optimizes memory and disk resources so that data takes much less space (specially if on-flight compression is used) than other solutions such as relational or object oriented databases.

You can have a look at the MainFeatures of PyTables. Also, find more info by reading the PyTables FAQ.

PyTables is developed, maintained and supported by Francesc Alted, with contributions from Ivan Vilela and the community.

How does PyTables compare with the h5py project?

Well, they are similar in that both packages are Python interfaces to the HDF5 library, but there are some important differences to be noted. h5py is an attempt to map the HDF5 feature set to NumPy as closely as possible. In addition, it also provides access to nearly all of the HDF5 C API.

Instead, PyTables builds up an additional abstraction layer on top of HDF5 and NumPy where it implements things like an enhanced type system, an engine for enabling complex queries, an efficient computational kernel, advanced indexing capabilities or an undo/redo feature, to name just a few. This additional layer also allows PyTables to be relatively independent of its underlying libraries (and their possible limitations). For example, PyTables can support HDF5 data types like enumerated or time that are available in the HDF5 library but not in the NumPy package, or even perform powerful complex queries that are not implemented directly in neither HDF5 nor NumPy.

Furthermore, PyTables also tries hard to be a high performance interface to HDF5/NumPy, implementing niceties like internal LRU caches for nodes and other data and metadata, automatic computation of optimal chunk sizes for the datasets, a variety of compressors, ranging from slow but efficient (bzip2) to extremely fast ones (Blosc) in addition to the standard zlib. Another difference is that PyTables makes use of numexpr so as to accelerate internal computations (for example, in evaluating complex queries) to a maximum.
Integration with Enterprise Systems

I currently have no answer to this, at least not one that I am willing to stake my career on (despite SAS’s cost, no one will ever be fired for choosing that technology stack – it works) ...

pyodbc and sqlalchemy? I think so, but I have not done much testing yet, and it’s not like people are blogging about connecting Python to Teradata ...

Different tact needed, more on this in a minute...
FUD I am eagerly spreading...

• Trends in Analytics
  • Businesses have increasing need for data science and not just statistics...
  • ... but there is a dearth of people with these skills, but to the extent they exist, Python dominates as the lingua franca...
  • ... ergo, if we want to grow internal, analytical competency, we need to look to draw from the right pools of talent

• Flexibility and speed to development
  • Rapid analytical prototyping lets us better assess the cost and the value of turning data into information
  • Python is also well-suited for high performance, production level work, and we don’t need multiple licenses for every new functional area
My three lines of attack...

This approach is tempting, but...

http://browsertoolkit.com/fault-tolerance.png
Talk!

• Find other Pythonistas, hopefully in other parts of the organization
  • This was critical! I found friends in high places, like our IT/IS groups
• Python is becoming not only application glue but also social glue:
  statisticians can learn from IT about web site development and how to interact
directly with our midframe data systems, and people like me can help them with
analytical needs (e.g. analyzing and visualizing server loads)

Examples:
Talk some more!

- Evangelize growing need for *data science* (and Python follows pretty easily)

http://www.drewconway.com/zia/?p=2378
Let’s end this talk with three recent examples of my team’s analytical work that used Python...

- Text analysis of customer survey data
- Real Estate team visualizing store performance
- Geospatial analysis of customer value and attrition

(Note that we use Python for true statistical modeling too but logistic regression is less exciting to show than the examples above, which do not involve modeling per se)
heatmap.py: create heatmaps in python

Download:  heatmap-1.0.tar.gz

http://www.jjguy.com/heatmap/
Static Excel graphics – an easy target for improvement
Questions?

P.S. Feel free to contact me if you want moral support in your quest to have Python take over the world.

P.P.S. We are hiring! (Denver, CO)

Josh Hemann
jhemann@sportsauthority.com