Python in Machine Learning

Neil D. Lawrence

MLO Lunch

25th March 2009

Outline

1. Perspective

2. What I Love about Matlab

3. Python as an Alternative

4. Scientific Computing in Python

5. Conclusions

Matlab

Ease of scientific computing

- Start of my PhD, 1996, Matlab dominated.
  - Roots in teaching linear algebra.
  - Cleve Moler was a researcher in numerical algorithms (contemporary of Gene Golub and others)
  - Initially it was an interface for linpack and eispack.
  - A lot of the underlying code is available as TOMS papers. Was persecuted for suggesting earth wasn't (Copernicanism) at centre of universe ...
  - Developed as an environment for scientific computing with excellent visualization facilities (3D plotting etc.)
  - They had/have a policy of reduced licence for academia, but charging industry a lot.
  - Sometimes the lag behind state of the art: interface to LAPACK (replacing linpack/eispack) came a little late.

Matlab Characteristics

- Characteristics: slow interpreter (faster with the JVM??), but fast if you vectorize (avoid loops).

```
% Subtract the mean
X = randn(100, 5);
meanX = mean(X, 1);

% instead of
for i = 1:100
  Xhat(i, :) = X(i, :) - meanX;
end

% we use
meanXmat = repmat(meanX, 100, 1);
Xhat = X - meanXmat;

% or worse
meanXmat = ones(100, 1)*meanX;
Xhat = X - meanXmat;
```

Matlab Characteristics II

- Elegant syntax, colon notation for extracting vectors.
- Poor object orientated interface.
- Clumsy interface with C/C++ (it seemed good in 1996, but hasn't changed!)
- Interface to Java??
  - Not tried it personally.
- Fairly poor

Reasons for an Alternative

- Matlab is the only non-free (as in beer and speech!) software on my system.
- For me and fellow MLers the product may be worth the money but,
- For my "clients": students, industry, colleagues in other disciplines.
    - It is not worth it.
- Is there an alternative that gives necessary functionality?

Things I love about Matlab

- Interaction with my data and algorithms:
  - See also: R, Octave
- Great visualization facilities:
  - Here it is ahead of R and Octave which *I think* use gnuplot.
- Ease of interoperation with other researchers.
  - R seems to upgrade and break things every 3 months.
  - Octave is v. close to Matlab. Recent version 3.0 works with most (all!) of my code!
  - But *I think* Matlab is about to undergo a major revamp.
- Most recently (and frustratingly!) ease of parallelization.
  - Gauss demo.

Python

Does it present a viable alternative?

- Python is an intepreted language, developed over the last 15 years.
  - Designed as a scripting language.
  - Aim was not to have the “write only” code feature of perl.
    - Object orientated.
    - Can be used as a funcitonal language?
- Operates from an interpreter.
  - About 10 years ago, I explored it for scientific computing.
  - It wasn't clear what the scientific library was and where it was going.
  - The command line interpreter wasn't up to the job (quite awkward to use)
  - There wasn't an obvious plotting library to use.
- Today the story is very different.

Current state of Affairs

- I returned to Python to help in debugging C++ code.
  - GP code for sparse models is quite complex.
  - Command line debugging is very tiresome.
  - Idea was to see if I could use Python to expose the code internals (i.e. call each method from command line).
  - Answer: Yes, I first tried C++ Boost interface, but discussion with Antti Honkela and Soeren Sonnenburg led me to SWIG.
  - Michalis's sparse variational GPs running in C++ called from Python.

    ```
    ipython -pylab
    cd mlprojects/gp/python
    run demGpSpgp1d5.py
    ```
- More fiddling and partial implementation of netlab.

Conclusions

Is it worth it?

- My worries are:
  - I have a lot of legacy code in Matlab.
  - Most ML researchers are used to Matlab.
- However:
  - When shipping code to “customers”. Python clearly has the edge.
  - It’s definitely worth us thinking about providing a consistent library in Python for all our teaching needs.
  - Students will appreciate consistency, and will be happier to learn Python rather than Matlab. For CPD it prevents clients from spending lots of money on licences. E.g. They could bring their own machines and we could install a Vbox distribution set up as needed.
  - My guess is, if we implement this, it will eventually be “inconvenient” to continue doing research in Matlab.