MACHINE LEARNING PROGRAMMING COURSE

Mentors

People @ Anagha Agile Systems are specialist in Extreme Programming on High End Computing, they carry 15 years of wide variety of industry experience on complex programming like scientific computing, rapid development using Agile Methodologies and hardcore concurrent programming.

- Have experience on developing coding features of Scientific Computing tool like Unscrambler.
- Developed optimized solutions of time critical and optimized financial transaction based products for world best FII.

Have experience in Machine Learning Software Development for clients like Oracle and Davita. Have nearly two experience on Advanced Python Training.

Mentors

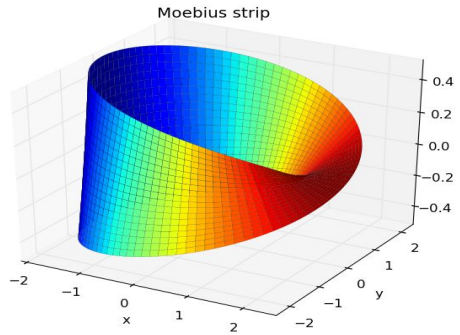
People @ Anagha Agile Systems are specialist in Extreme Programming on High End Computing, they carry 15 years of wide variety of industry experience on complex programming like scientific computing, rapid development using Agile Methodologies and hardcore concurrent programming.

- Have experience on developing coding features of Scientific Computing tool like Unscrambler.
- Developed optimized solutions of time critical and optimized financial transaction based products for world best FII.

Contact Us

Email: info@anaghaagilesystems.com
Web: http://www.anaghaagilesystems.com

Machine Learning Programming using Python



ANAGHA AGILE SYSTEMS
Imagine Inspire Implement!

Make IPython Your Armour

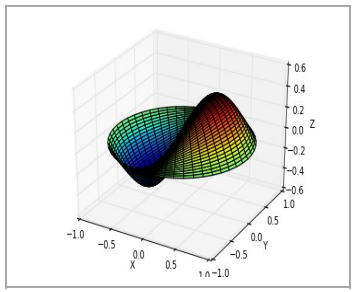
This class is intended for the scientist, engineer or analyst interested in using Python for their day-to-day computational tasks.

- It begins with a first week introduction to the Python language focusing on standard data structures, control constructs, and text/file processing. Object-oriented development is briefly discussed.
- Second Week introduces numeric data processing and rapidly manipulating and processing large data sets using NumPy arrays. Day three covers several advanced topics in NumPy such as structured arrays and memory-mapped arrays for dealing with large data.
- This is followed by a survey of the scientific algorithms available in SciPy including interpolation, integration, linear algebra, signal/image processing, optimization, and others. Third week finishes with an introduction to time series and data wrangling with Pandas.
- The fourth week covers the necessary tools to write robust and efficient Python code: a unit test framework, the Python debugger, and more.
- The last week with a one-day module on Building Scientific Graphical User Interfaces (GUIs).

SciPy (pronounced "Sigh Pie") is open-source software for mathematics, science, and engineering. It's all about scientific programming with Python.

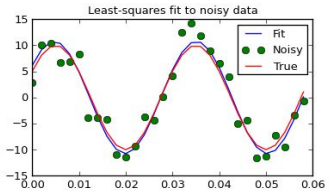
Overview of Workshops

NumPy Workshop



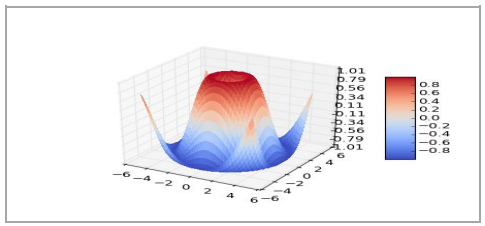
- Array manipulations
- Understanding the N-dimensional data structure
- Higher-order patterns with ufunc methods
- Broadcasting of array operations
- Working with "structured" arrays
- Memory mapping large data sets
- Data statistics
- Crude integral approximations

SciPy Workshop



- 3D plotting
- Polynomials
- Statistics Optimization
- Linear Algebra
- Interpolation
- Signal Processing
- Integration

Developing Matplotlib & Sci-Computing Apps



- Basic plotting
- 2D plots
- Histograms
- Scatter plots
- Displaying images

Getting Started

Scientific Course Includes

The first half of the course introduces the Python programming language to scientists. Pace of training is dependent on learning ability of class and geared toward individuals who are freshers to programming language such as C or Matlab or C++.

The first half of the course has two section, in the first section attendees will learn the basic constructs of the language and how to do basic numerical analysis with Python. In the second section covers the SciPy library that provides modules for linear algebra, signal processing, optimization, statistics, genetic algorithms, interpolation, ODE solvers, special functions, etc. We also cover scientific plotting with python using matplotlib.

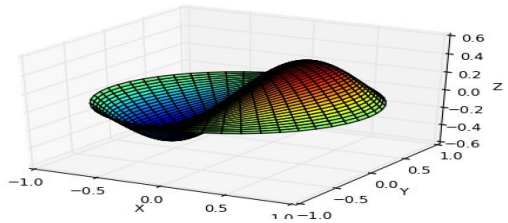
This second half of the course has two section, the first section covers advanced topics in scientific computing such as integrating Python with other languages and parallel programming. The second section covers Tools such as IPython, SWIG, f2py, and Boost Python are all discussed along with common pitfalls and good design practices.

The final session covers testing and performance testing with an emphasis on python SciPy & Nose package.

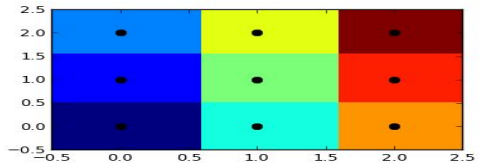
"The SciPy Stack, a collection of open source software for scientific computing in Python, and particularly a specified set of core packages like NumPy, SciPy, MatPlotlib, Pandas, SymPy, nose and IPython."

Software: The tools used for this course are : Python (2.x >= 2.6 or 3.x >= 3.2)

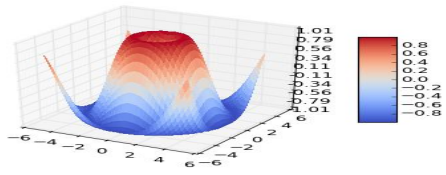
- NumPy (>= 1.6)



- SciPy library (>= 0.10)



- Matplotlib (>= 1.1)



Mission: To empower students with well-developed and powerful modules of scientific computing python packages to unleash the power of high performance computing.

Scientific Computing Course Outline

1: Introduction to Python and Scientific Computing.
2: NumPy arrays: tips and tricks:
3: Linear Algebra
4: Interpolation:
5: Optimization and fitting techniques:
6: Ordinary differential equations
7: Root finding
8: Data rebinning Examples of rebinning data to produce smaller arrays with and without interpolation.
9: Histograms 2D histograms with variable bin width
10: Convex Hull Finds the convex hull around a set of data points.
11: Minimum Point of a Convex Hull Finds the minimum point of the convex hull of a finite set of points.
12: Kalman Filtering Example from the Welch & Bishop Introduction to the Kalman Filter.
13: Comm Theory Example of BPSK simulation.
14: Smoothing a signal Performing smoothing of 1D and 2D signals by convolving them with a window.
15: KDTree Searching
16: Butterworth Bandpass Filter Create and apply a Butterworth bandpass filter.
17: FIR Filter Design, Design a low-pass FIR filter using the window method.
18: How to apply a FIR filter: signal. Convolve, signal.fftconvolve, ndimage.convolve1d or signal. Filter?
19: Multithreading Easy multithreading for embarrassingly parallel multidimensional space using kd-trees.
20: Particle Filter A simple particle filter algorithm for tracking objects in a video sequence.
21: Brownian motion Compute Brownian motion (i.e. the Wiener process).
22: Correlated Random Samples Generate correlated normally distributed random samples.

Scientific Computing Course Outline

23: Large Markov Chains Find the stationary distribution of a large Markov chain; the M/M/1 tandem queue
24: Watershed algorithm Apply the watershed algorithm in order to split an array into distinct components.
25: Linear Classification Fisher's discriminant function and Probabilistic Generative model
26: Plot an eye diagram using numpy and matplotlib.
27: Introduction to SciPy
28: Top-level SciPy
29: SciPy for Linear Algebra
30: SciPy for Numerical Analysis
31: SciPy for Signal Processing
32: SciPy for Data Mining
33: SciPy for Computational Geometry
34: Interaction with Other Languages
35: Introduction to Matplotlib
36: Getting Started with Matplotlib
37: Decorate Graphs with Plot Styles and Types
38: Advanced Matplotlib
39: Embedding Matplotlib in GTK+
40: Embedding Matplotlib in Qt 4
41: Embedding Matplotlib in wxWidgets
42: Matplotlib for the Web
43: Matplotlib in the Real World

The great benefit of this course is its fast learning curve allowing students to quickly start working on interesting motivating problems.