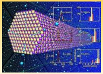
Scientific Computing using Python

Manu Krishnan T.V

07103044

Dept. of CSE, MESCE
Guide: Jahfar Ali

January 4, 2011



Outline

1 Introduction

2 Python

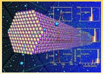
3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion



Outline

1 Introduction

2 Python

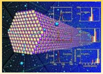
3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion



Outline

1 Introduction

2 Python

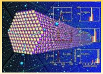
3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion



Outline

1 Introduction

2 Python

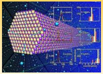
3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion



Outline

1 Introduction

2 Python

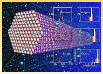
3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion



Outline

1 Introduction

2 Python

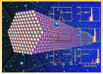
3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion



Outline

1 Introduction

2 Python

3 NumPy

4 SciPy

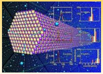
5 Matplotlib

6 Advantages

7 Conclusion

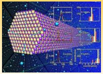
Scientific Computing

The field of study concerned with constructing mathematical models and quantitative analysis techniques and using computers to analyse and solve scientific problems.



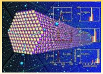
Problem Domains

- Numerical simulations
  - Reconstruct and understand known events.
  - Predict future or unobserved situations.
- Model fitting and data analysis
  - Use graph theory to model networks, especially those connecting individuals, organizations, and websites.
- Computational optimization
  - Optimize known scenarios like technical and manufacturing processes, front-end engineering, etc.



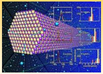
Problem Domains

- Numerical simulations
  - Reconstruct and understand known events.
  - Predict future or unobserved situations.
- Model fitting and data analysis
  - Use graph theory to model networks, especially those connecting individuals, organizations, and websites.
- Computational optimization
  - Optimize known scenarios like technical and manufacturing processes, front-end engineering, etc.



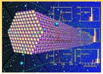
Problem Domains

- Numerical simulations
  - Reconstruct and understand known events.
  - Predict future or unobserved situations.
- Model fitting and data analysis
  - Use graph theory to model networks, especially those connecting individuals, organizations, and websites.
- Computational optimization
  - Optimize known scenarios like technical and manufacturing processes, front-end engineering, etc.



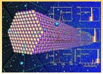
Methods and Algorithms

- Numerical analysis
- Application of Taylor series as convergent and asymptotic series
- Computing derivatives by Automatic differentiation (AD)
- Computing derivatives by finite differences
- Molecular dynamics
- Discrete Fourier transform and applications.
- etc



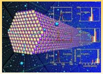
Methods and Algorithms

- Numerical analysis
- Application of Taylor series as convergent and asymptotic series
- Computing derivatives by Automatic differentiation (AD)
- Computing derivatives by finite differences
- Molecular dynamics
- Discrete Fourier transform and applications.
- etc



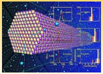
Methods and Algorithms

- Numerical analysis
- Application of Taylor series as convergent and asymptotic series
- Computing derivatives by Automatic differentiation (AD)
- Computing derivatives by finite differences
- Molecular dynamics
- Discrete Fourier transform and applications.
- etc



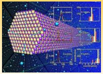
Methods and Algorithms

- Numerical analysis
- Application of Taylor series as convergent and asymptotic series
- Computing derivatives by Automatic differentiation (AD)
- Computing derivatives by finite differences
- Molecular dynamics
- Discrete Fourier transform and applications.
- etc



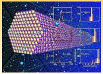
Methods and Algorithms

- Numerical analysis
- Application of Taylor series as convergent and asymptotic series
- Computing derivatives by Automatic differentiation (AD)
- Computing derivatives by finite differences
- Molecular dynamics
- Discrete Fourier transform and applications.
- etc



Methods and Algorithms

- Numerical analysis
- Application of Taylor series as convergent and asymptotic series
- Computing derivatives by Automatic differentiation (AD)
- Computing derivatives by finite differences
- Molecular dynamics
- Discrete Fourier transform and applications.
- etc



Methods and Algorithms

- Numerical analysis
- Application of Taylor series as convergent and asymptotic series
- Computing derivatives by Automatic differentiation (AD)
- Computing derivatives by finite differences
- Molecular dynamics
- Discrete Fourier transform and applications.
- etc

Outline

1 Introduction

2 Python

3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- "batteries included"
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- ”batteries included”
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- "batteries included"
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- "batteries included"
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- "batteries included"
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- ”batteries included”
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- ”batteries included”
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- "batteries included"
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Python

Python is an interpreted, general-purpose high-level programming language.

Features:

- Cross Platfom
- Clear Syntax
- "batteries included"
  - Has a large library
- Easy to code
- Multi Paradigm
- Useful Built-In Objects
- Functions and Classes
- Ease of Extension

Cross Platform

- Python supports multiple operating systems.
  - GNU/Linux
  - Mac
  - Windows
- Runs on even low level embedded systems.

Cross Platform

- Python supports multiple operating systems.
  - GNU/Linux
  - Mac
  - Windows
- Runs on even low level embedded systems.

Clear Syntax

```
def is_digits_even(n):
  if n < 0:
    n = -n
  while n > 0:
    if n % 2 == 1:
      return False
    n /= 10
  return True

a=input()
if is_digits_even(a):
  print "All digits are even!"
```

Batteries Included

- Has a vast library.
- New libraries can be easily added.

Batteries Included

- Has a vast library.
- New libraries can be easily added.

Easy to Code

```
def is_digits_even(n):
  if n < 0:
    n = -n
  while n > 0:
    if n % 2 == 1:
      return False
    n /= 10
  return True

a=input()
if is_digits_even(a):
  print "All digits are even!"
```

Multi Paradigm

- Object Oriented
- Functional
- Imperative

Multi Paradigm

- Object Oriented
- Functional
- Imperative

Multi Paradigm

- Object Oriented
- Functional
- Imperative

Useful Built-In Objects

```
>>> type(1), type(1.0), type(1.0j), type('one')
(<type 'int'>, <type 'float'>, <type 'complex'>,
<type 'str'>)
```

Functions & Classes

```
def signum( r ):
    """returns 0 if r is zero
    -1 if r is negative
    +1 if r is positive"""
    if r < 0:
        return -1
    elif r > 0:
        return 1
    else:
        return 0
```

Ease of extension

- Python is extended with a large C-API.
- Enables faster execution.
- Can easily connect to non-Python compiled code.

Ease of extension

- Python is extended with a large C-API.
- Enables faster execution.
- Can easily connect to non-Python compiled code.

Ease of extension

- Python is extended with a large C-API.
- Enables faster execution.
- Can easily connect to non-Python compiled code.

Outline

1. Introduction
2. Python
3. NumPy
4. SciPy
5. Matplotlib
6. Advantages
7. Conclusion

What is NumPy

- NumPy is a Python Module.
- Supports complex numerical calculations.
- Freely available at http://numpy.org
- It grew out of an original module called Numeric.
- Version 1.0 released in October 2006.
- Current stable version: 1.5.1 (November 21, 2010)

What is NumPy

- NumPy is a Python Module.
- Supports complex numerical calculations.
- Freely available at http://numpy.org
- It grew out of an original module called Numeric.
- Version 1.0 released in October 2006.
- Current stable version: 1.5.1 (November 21, 2010)

What is NumPy

- NumPy is a Python Module.
- Supports complex numerical calculations.
- Freely available at http://numpy.org
- It grew out of an original module called Numeric.
- Version 1.0 released in October 2006.
- Current stable version: 1.5.1 (November 21, 2010)

What is NumPy

- NumPy is a Python Module.
- Supports complex numerical calculations.
- Freely available at http://numpy.org
- It grew out of an original module called Numeric.
- Version 1.0 released in October 2006.
- Current stable version: 1.5.1 (November 21, 2010)

What is NumPy

- NumPy is a Python Module.
- Supports complex numerical calculations.
- Freely available at http://numpy.org
- It grew out of an original module called Numeric.
- Version 1.0 released in October 2006.
- Current stable version: 1.5.1 (November 21, 2010)

What is NumPy

- NumPy is a Python Module.
- Supports complex numerical calculations.
- Freely available at http://numpy.org
- It grew out of an original module called Numeric.
- Version 1.0 released in October 2006.
- Current stable version: 1.5.1 (November 21, 2010)

Data types

- Supports all fundamental C data types.
- Additional support to unicode strings.
- Supports userdefined data types.

Data types

- Supports all fundamental C data types.
- Additional support to unicode strings.
- Supports userdefined data types.

Data types

- Supports all fundamental C data types.
- Additional support to unicode strings.
- Supports userdefined data types.

Data types

```
>>> import numpy as N

>>> dt = N.dtype([(id, i4),
(name, S12), (scores, u1, 4)])

>>> a = N.array([(1001, James, [100,98,97,60]),
(1002, Kathy,[100,100,85,98]),
(1003, Michael,[84,75, 98,100]),
(1004, John, [84,76,82,92])], dtype=dt)
```

Data types

```
>>> a[name]
array([James, Kathy, Michael,
John], dtype=S12)

>>> a[scores]
array([[100, 98, 97, 60],
 [100, 100, 85, 98],
 [ 84, 75, 98, 100],
 [ 84, 76, 82, 92], dtype=uint8)
```

Attributes and methods

- All arrays have several attributes and methods.
- Some attributes can be set to alter the array’s characteristics.

```
>>> b = a.take(a[’name’].argsort())

>>> print b
array([(1001, ’James’, [100, 98, 97, 60]),
   (1004, ’John’, [84, 76, 82, 92]),
   (1002, ’Kathy’, [100, 100, 85, 98]),
   (1003, ’Michael’, [84, 75, 98, 100])],
   dtype=[(’id’, ’<i4’), (’name’, ’|S12’),
   (’scores’, ’|u1’, 4)])
```

Indexing

- Supports multiple kinds of indexing.

```
>>> import numpy as n

>>> a = n.random.randn(50,25)

>>> print a.shape, a[10,15]
(50, 25) 0.5295135653
```

Universal Functions

- Simple to define functions that take N inputs and return M outputs.
- Provides universal function objects (ufuncs).
- More than 50 mathematical functions are defined as universal functions.

Universal Functions

- Simple to define functions that take N inputs and return M outputs.
- Provides universal function objects (ufuncs).
- More than 50 mathematical functions are defined as universal functions.

Universal Functions

- Simple to define functions that take N inputs and return M outputs.
- Provides universal function objects (ufuncs).
- More than 50 mathematical functions are defined as universal functions.

Universal Functions - Features

- **Broadcasting**: Specific method for arrays that don't have the same shape to try and act as if they do.
- Provides features to define output arrays explicitly.
- Memory saving type conversion.
- **Hardware error handling**: Supports querying the result of hardware error flags.

Universal Functions - Features

- **Broadcasting**: Specific method for arrays that don't have the same shape to try and act as if they do.
- Provides features to define output arrays explicitly.
- Memory saving type conversion.
- **Hardware error handling**: Supports querying the result of hardware error flags.

Universal Functions - Features

- **Broadcasting**: Specific method for arrays that don't have the same shape to try and act as if they do.
- Provides features to define output arrays explicitly.
- Memory saving type conversion.
- **Hardware error handling**: Supports querying the result of hardware error flags.

Universal Functions - Features

- **Broadcasting**: Specific method for arrays that don't have the same shape to try and act as if they do.
- Provides features to define output arrays explicitly.
- Memory saving type conversion.
- **Hardware error handling**: Supports querying the result of hardware error flags.

Outline

1 Introduction

2 Python

3 NumPy

4 SciPy

5 Matplotlib

6 Advantages

7 Conclusion

What is SciPy

- SciPy is again a Python Module.
- Supports optimizations, special functions, image processing, etc
- Depends on NumPy.
- Freely available at http://scipy.org
- Current stable version: 0.8.0 (July 28, 2010)

What is SciPy

- SciPy is again a Python Module.
- Supports optimizations, special functions, image processing, etc
- Depends on NumPy.
- Freely available at http://scipy.org
- Current stable version: 0.8.0 (July 28, 2010)

What is SciPy

- SciPy is again a Python Module.
- Supports optimizations, special functions, image processing, etc
- Depends on NumPy.
- Freely available at http://scipy.org
- Current stable version: 0.8.0 (July 28, 2010)

What is SciPy

- SciPy is again a Python Module.
- Supports optimizations, special functions, image processing, etc
- Depends on NumPy.
- Freely available at http://scipy.org
- Current stable version: 0.8.0 (July 28, 2010)

What is SciPy

- SciPy is again a Python Module.
- Supports optimizations, special functions, image processing, etc
- Depends on NumPy.
- Freely available at http://scipy.org
- Current stable version: 0.8.0 (July 28, 2010)

SciPy

IPython

- IPython is an interactive shell for the Python programming language.
- A component of the SciPy package.
- Supports additional shell syntax, syntax highlighting, tab completion and rich history.

IPython

- IPython is an interactive shell for the Python programming language.
- A component of the SciPy package.
- Supports additional shell syntax, syntax highlighting, tab completion and rich history.

IPython

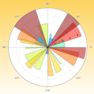
- IPython is an interactive shell for the Python programming language.
- A component of the SciPy package.
- Supports additional shell syntax, syntax highlighting, tab completion and rich history.

Outline

1. Introduction
2. Python
3. NumPy
4. SciPy
5. **Matplotlib**
6. Advantages
7. Conclusion

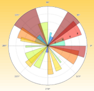
Matplotlib

- Matplotlib is a plotting library for the Python programming language and its NumPy numerical mathematics extension.
- Supports generic GUI toolkits, like wxPython, Qt, or GTK.
- Availiable at http://matplotlib.sourceforge.net/
- Current stable version: 1.0.0 (7 July 2010)



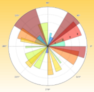
Matplotlib

- Matplotlib is a plotting library for the Python programming language and its NumPy numerical mathematics extension.
- Supports generic GUI toolkits, like wxPython, Qt, or GTK.
- Availiable at http://matplotlib.sourceforge.net/
- Current stable version: 1.0.0 (7 July 2010)



Matplotlib

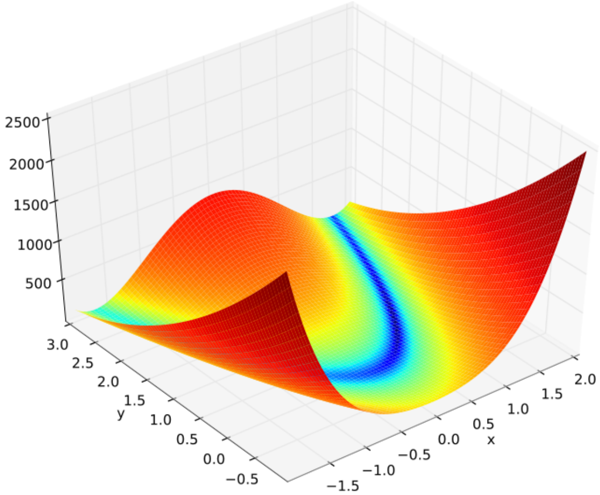
- Matplotlib is a plotting library for the Python programming language and its NumPy numerical mathematics extension.
- Supports generic GUI toolkits, like wxPython, Qt, or GTK.
- Availiable at http://matplotlib.sourceforge.net/
- Current stable version: 1.0.0 (7 July 2010)



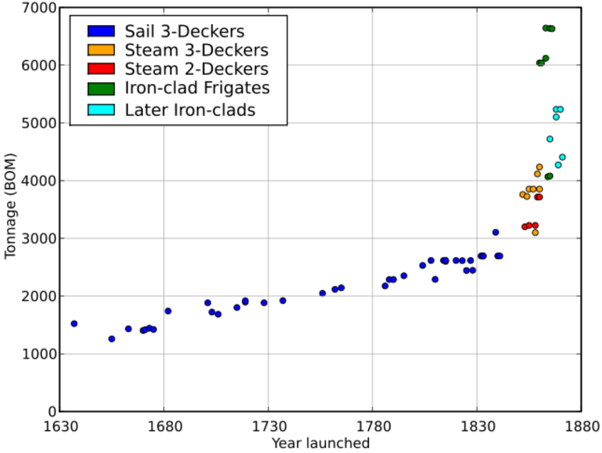
Matplotlib

- Matplotlib is a plotting library for the Python programming language and its NumPy numerical mathematics extension.
- Supports generic GUI toolkits, like wxPython, Qt, or GTK.
- Availiable at http://matplotlib.sourceforge.net/
- Current stable version: 1.0.0 (7 July 2010)

Sample Plot

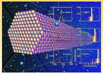

2500
2000
1500
1000
500
3.0
2.5
2.0
1.5
y
1.0
0.5
0.0
−0.5
−1.5
−1.0
−0.5
0.0
0.5
x
1.0
1.5
2.0

Sample Plot


Sail 3-Deckers
Steam 3-Deckers
Steam 2-Deckers
Iron-clad Frigates
Later Iron-clads
Tonnage (BOM)
7000
6000
5000
4000
3000
2000
1000
0
1630
1680
1730
1780
1830
1880
Year launched

Outline

1. Introduction
2. Python
3. NumPy
4. SciPy
5. Matplotlib
6. Advantages
7. Conclusion

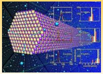


Python + NumPy + Matplotlib

- Based on Python, a full-featured modern OOP language suitable for large-scale software development.
- Suitable for fast scripting, including CGI scripts.
- Free, open source, no license servers.
- Native SVG support

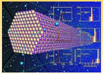
Python + NumPy + Matplotlib

- Based on Python, a full-featured modern OOP language suitable for large-scale software development.
- Suitable for fast scripting, including CGI scripts.
- Free, open source, no license servers.
- Native SVG support



Python + NumPy + Matplotlib

- Based on Python, a full-featured modern OOP language suitable for large-scale software development.
- Suitable for fast scripting, including CGI scripts.
- Free, open source, no license servers.
- Native SVG support



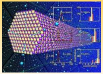
Python + NumPy + Matplotlib

- Based on Python, a full-featured modern OOP language suitable for large-scale software development.
- Suitable for fast scripting, including CGI scripts.
- Free, open source, no license servers.
- Native SVG support

Outline

1. Introduction
2. Python
3. NumPy
4. SciPy
5. Matplotlib
6. Advantages
7. Conclusion

Conclusion



Reference

- **Python for Scientific Computing**
  Travis E Oliphant
  IEEE 1521-9615/07
- **Guide to NumPy**
  Travis E Oliphant
  Trelgol, 2006;
  www.trelgol.org.
- **Wikipedia**
  http://en.wikipedia.org

Thank You !

tvmanukrishnan@gmail.com
http://www.bizzard.info

This document is created using LaTeX