(PRACTICAL) COMPUTATIONAL PHYSICS

Physics 551
Lecture 11

NOTATION

Extra Reading

Optional Exercise

Recommended

- This lecture slides for this course will attempt to use a uniform notation throughout. A normal paragraph looks like this.

✑ *Italicized paragraphs with pen bullets will indicate definitions, with the defined word or phrase shown in* ***SMALL-CAPS****.*

✐ Pencil bullets will indicate the introduction of **new notation**.

☞ Pointing hand bullets indicate important points that might otherwise be overlooked.

ANNOUNCEMENTS

ANNOUNCEMENTS

- To **clone** this week's **Python demonstration materials** please invoke

```
$git clone https://github.com/hughdickinson/CompPhysL11Python.git
   /home/computationalphysics/Documents/python/lecture11
```

- A new version of **IPython Notebook** is available. To **install it**, please ensure you have network connection and invoke

```
$sudo pip install --upgrade ipython[all]
```

☞ The required **password** is the same as the Ubuntu login password.

☞ You can also find these commands on the Blackboard Learn website.

PYTHON

SCIENTIFIC COMPUTATION IN PYTHON USING SCIPY

THE SCIPY PACKAGE

- The `scipy` package provides efficient, **ready-to-use** implementations of numerous **computational tools** and **algorithms** to streamline **scientific analyses**.
- The modules provided by `scipy` are **documented** using **extensive reference** and **tutorial** material.
- The `scipy` package **integrates transparently** with the `numpy` and `matplotlib` packages, and leverage the powerful numerical analysis and plotting functionality they provide.

SUB-PACKAGES OF SCIPY

- The `scipy` package defines **18 sub-packages**, providing **rich functionality** that is tailored for scientific computing.
- The sub-packages provided by `scipy` include:
  - `scipy.constants` - provides numeric values for a large number of **physical** and **mathematical constants**.
  - `scipy.stats` - facilitates the generation of randomly sampled, **simulated datasets** that are consistent with any of **over 90** probability distributions.

SUB-PACKAGES OF SCIPY

- `scipy.fftpack` - provides **signal processing** capability using the **Fast Fourier Transform**.
- `scipy.optimize` - provides abundant facilities for **multidimensional** function **optimization**, fitting of **binned** and **un-binned** datasets, simple curve fitting using **nonlinear least-squares** and function **root finding**.
- `scipy.io` - enables **handling of files** using a number of commonly encountered **proprietary formats** including the **MATLAB** file format and the **IDL** file format.

scipy.constants

https://docs.scipy.org/doc/scipy-0.15.1/reference/constants.html

CONSTANTS PACKAGE OVERVIEW

- Provides high precision **numeric values** for mathematical constants like π.
- Provides the scaling factors that correspond to **unit prefixes** e.g. nano.
- Provides **numeric values** for empirically determined physical values.
- Provides an interface to the table of ***CODATA*** *Recommended Values of the Fundamental Physical Constants 2010* defined by the National Institute of Science and Technology (NIST).
- Defines several functions that perform (slightly) more complex **unit conversion operations**.

DEMONSTRATION

SciPy Examples

1. Physical and Mathematical Constants and Units

Clone the Lecture 11 Python demonstration material from GitHub:

```
$git clone https://github.com/hughdickinson/CompPhysL11Python.git
 /home/computationalphysics/Documents/python/lecture11
```

`scipy.fftpack`

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/fftpack.html

FFTPACK
PACKAGE OVERVIEW

- Implements **signal processing** functions based upon the **Fast Fourier Transform** (FFT).
- Special cases include the **Discrete Cosine Transform** (DCT) and **Discrete Sine Transform** (DST).
- Provides helper functions to **prepare input data** and **interpret generated results**.
- Provides a high-level interface to perform **convolution operations** using the FFT.

DEMONSTRATION

SciPy Examples

2. Signal Processing - The Fast Fourier Transform

Clone the Lecture 11 Python demonstration material from GitHub:

```
$git clone https://github.com/hughdickinson/CompPhysL11Python.git
 /home/computationalphysics/Documents/python/lecture11
```

scipy.stats

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/stats.html

STATS
PACKAGE OVERVIEW

- Provides facilities for working with **probability distributions** e.g. *Poisson*, *Normal*, *Gamma* etc.
- Provides ***Random Variable*** (RV) classes that model over **90** discrete and continuous distributions.
- The RV classes provide methods to **customize** and **characterize** the modeled probability distributions.
- The RV classes provide methods that **generate random variates** using the modeled probability distributions.

scipy.optimize

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/optimize.html

OPTIMIZE PACKAGE OVERVIEW

- Implements numerous algorithms to perform **optimization** of **arbitrary**, **multidimensional** parameterized functions.
- The algorithms can be used to **fit** arbitrary, multidimensional **model** functions to **binned** and **un-binned** datasets.
- Provides **high-level** functions to perform **model-fitting** to **binned** datasets using **nonlinear least-squares**.
- Provides high-level functions to **determine the roots** of arbitrary, multidimensional functions.

DEMONSTRATION

SciPy Examples

3. Simulating Artificial Datasets, Fitting and Optimization

Clone the Lecture 11 Python demonstration material from GitHub:

```
$git clone https://github.com/hughdickinson/CompPhysL11Python.git
 /home/computationalphysics/Documents/python/lecture11
```

scipy.io

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/io.html

IO
PACKAGE OVERVIEW

- Provides functions that enable data to be **loaded** from several **proprietary file formats**.
- **Supported formats** include:
  - MATLAB files
  - IDL files
  - Unformatted FORTRAN output files.
  - Matrix Market files.
- **Writing** is also supported for **some** formats.

DEMONSTRATION

SciPy Examples

4. Handling Special File Formats

Clone the Lecture 11 Python demonstration material from GitHub:

```
$git clone https://github.com/hughdickinson/CompPhysL11Python.git
 /home/computationalphysics/Documents/python/lecture11
```

LECTURE 11 SUMMARY

- After reviewing the material from this lecture (including the demonstration material) **and completing the reading exercises** you should know:
    1. That the `scipy` package defines **18 sub-packages** that provide numerous **ready-to-use tools** and **algorithms** for scientific computation.
    2. That the `scipy.constants` sub-package provides **numeric values** and **associated metadata** for a numerous **physical** and **mathematical** constants.
    3. How to use the interfaces that `scipy.constants` provides.

LECTURE 11 SUMMARY

4. That the `scipy.fftpack` sub-package enables signal data to be processed using the FFT algorithm.
5. How to use the `fftpack.fft(…)` and `fftpack.ifft(…)` functions to perform **forward** and **inverse FFT operations** on data.
6. How to **manipulate** and **interpret** the data that are returned by the `fftpack.fft(…)` function using the `fftpack.fftshift(…)` and `fftpack.fftfreq(…)` functions.
7. How to plot bar charts using the `pyplot.bar(…)` function.

LECTURE 11 SUMMARY

8. That the `scipy.stats` sub-package provides over 90 "*Random Variable*" (RV) classes that model physically relevant probability distributions.
9. How to invoke the `rvs(…)` method provided by all RV classes to **generate random variates** consistent with the modeled distribution.
10. How to compute the probability of obtaining a particular measurement, given a particular probability distribution using the `pmf(…)` or `pdf(…)` methods of an RV class.

LECTURE 11 SUMMARY

11. How to generate and plot a **histogram** of an un-binned dataset using the `pyplot.hist(...)` function.
12. How to use the `scipy.optimize.minimize(...)` function to fit an arbitrary parameterized function to an **un-binned** dataset.
13. How to use the `scipy.optimize.curve_fit(...)` function to fit an arbitrary parameterized function to a **binned** dataset using **nonlinear least-squares**.
14. How to **read** and **write MATLAB** files in ***Python***.

RECOMMENDED READING

`scipy.constants`

https://docs.scipy.org/doc/scipy-0.15.1/reference/constants.html

`scipy.fftpack`

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/fftpack.html

`scipy.stats`

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/stats.html

`scipy.optimize`

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/optimize.html

`scipy.io`

https://docs.scipy.org/doc/scipy-0.15.1/reference/tutorial/io.html

LECTURE 11 HOMEWORK

Review the **Python** demonstration material from Lecture **11**!

- Review the ***Recommended Reading*** items listed on the previous slide.

- **If necessary,** review Reading Assignments from the homework for Lecture 9 and Lecture 10.

- Complete the **Lecture 11 Homework Quiz** that you will find on the course Blackboard Learn website.