Python - All A Scientist Needs

openwetware.org/wiki/python

Julius B. Lucks

Miller Institute and Berkeley Center for Synthetic Biology

University of California, Berkeley

Python

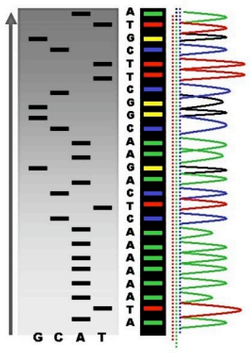
- Python is a complete scientific programming platform
- Python promotes good scientific practice

The Scientist's Dilemma

Hypothesis

Raw Data

ABI-3100 16 Capillary DNA Sequencer [1]



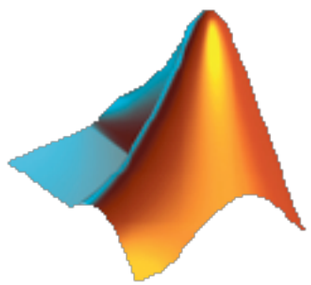
[2]

Data Processing

ATGC...

Data Management

ATGC...
ATGC...
ATGC...
ATGC...
ATGC...
ATGC...

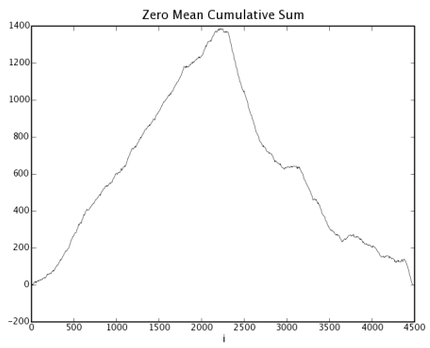


Matlab

Mathematica

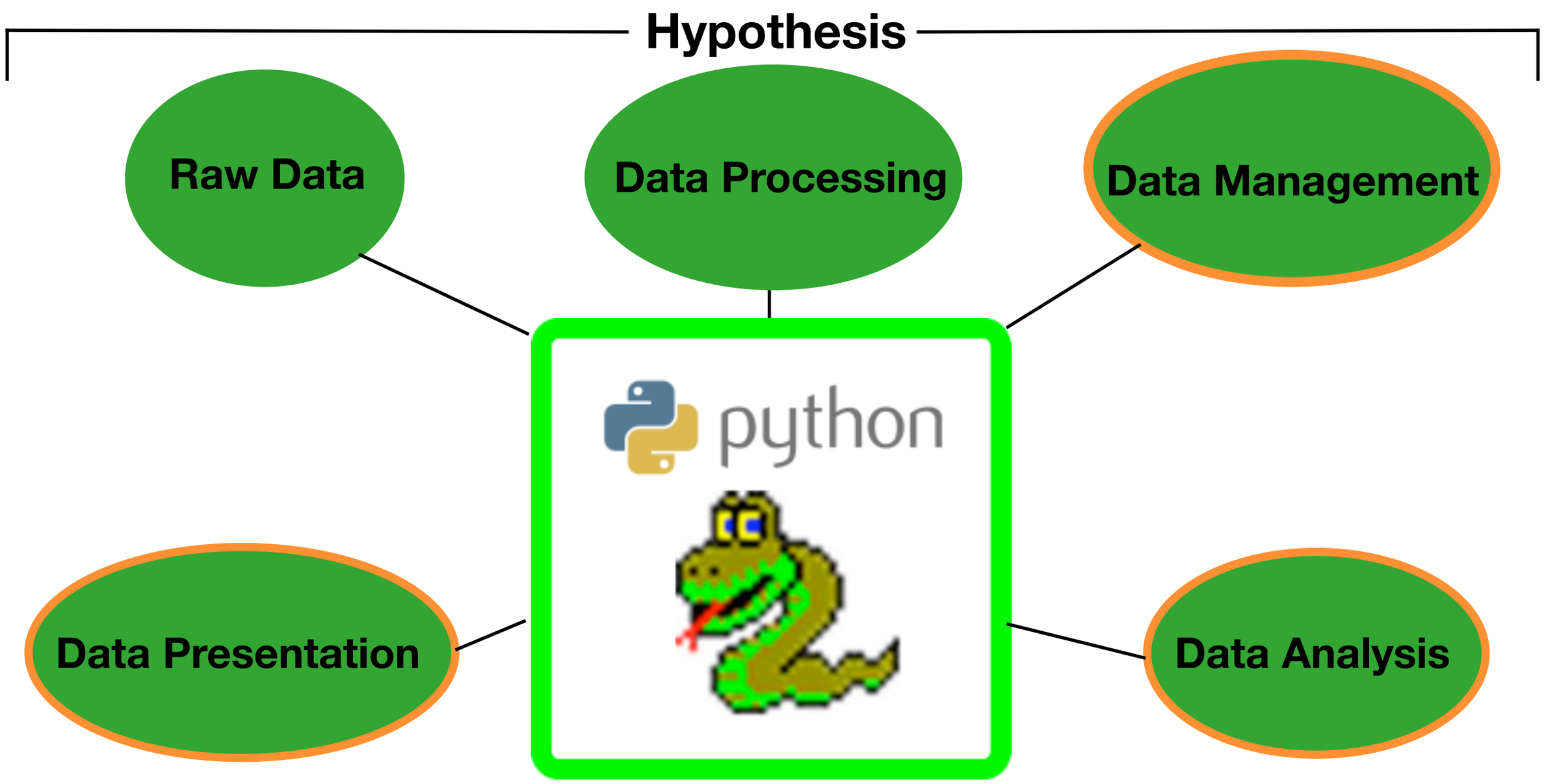
Excel

Data Analysis



Data Presentation

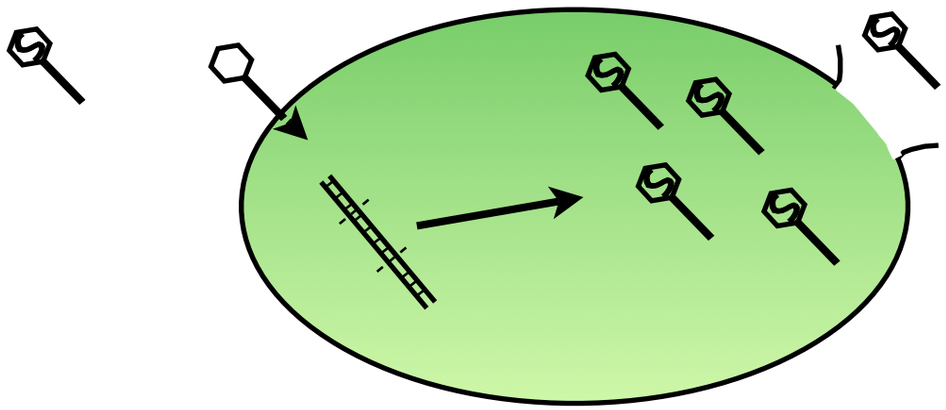
The Scientist's Dilemma



Case Study

Comparative Genomics of Viruses

Viral Life Cycle



Protein ····· Ser Pro Thr Ala ·····

mRNA1 ·····UCUCCUACUGCU·····

mRNA2 ·····UCCCCCACCGCC·····

⋮

The Genetic Code

UUU, UUC } Phe	UCU, UCC, UCA, UCG } Ser	UAU, UAC } Tyr	UGU, UGC } Cys
UUA, UUG } Leu		UAA Stop	UGA Stop
		UAG Stop	UGG Trp
CUU, CUC, CUA, CUG } Leu	CCU, CCC, CCA, CCG } Pro	CAU, CAC } His	CGU, CGC, CGA, CGG } Arg
		CAA, CAG } Gln	
AUU, AUC, AUA } Ile	ACU, ACC, ACA, ACG } Thr	AAU, AAC } Asn	AGU, AGC } Ser
AUG Met		AAA, AAG } Lys	AGA, AGG } Arg
GUU, GUC, GUA, GUG } Val	GCU, GCC, GCA, GCG } Ala	GAU, GAC } Asp	GGU, GGC, GGA, GGG } Gly
		GAA, GAG } Glu	

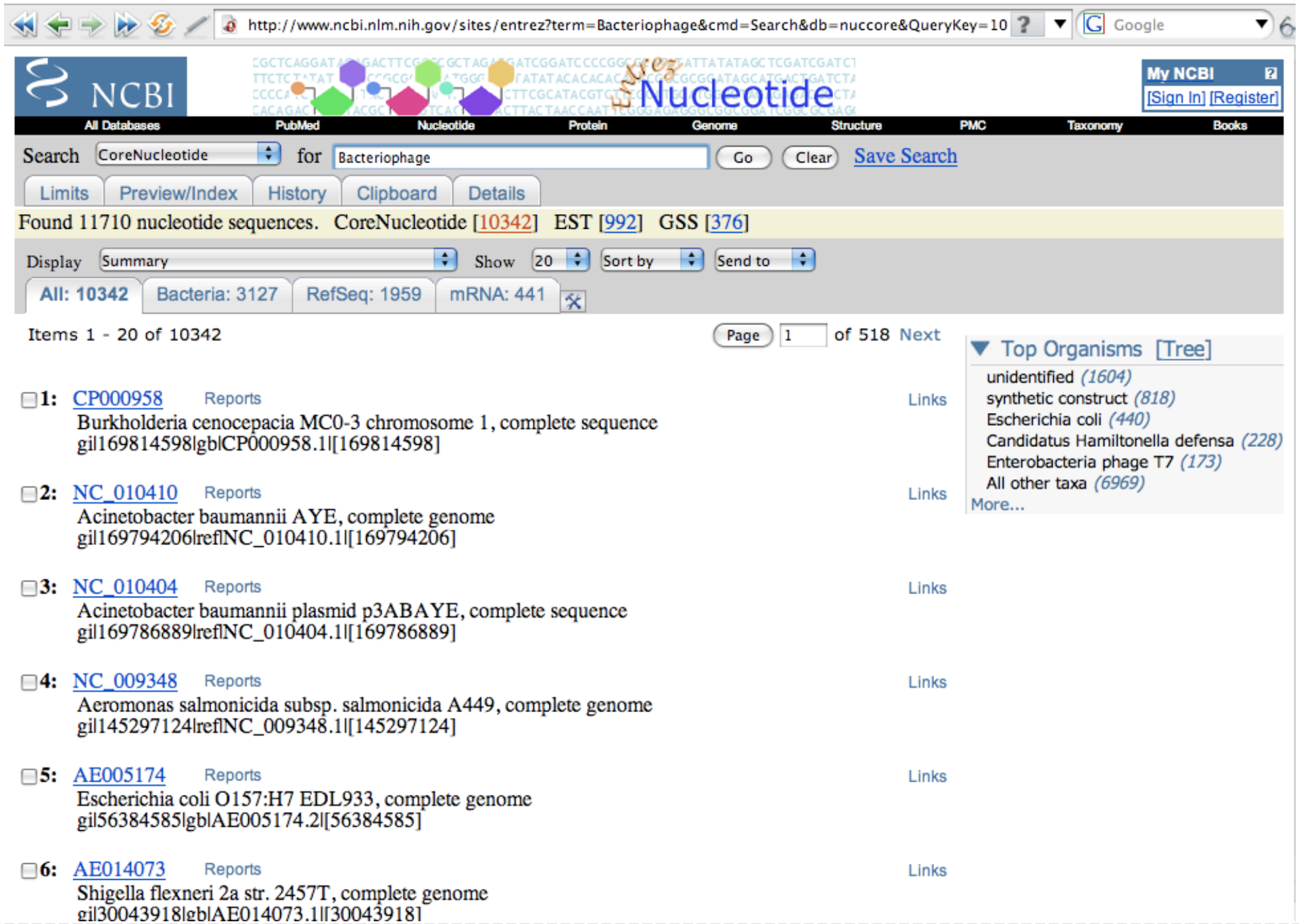
[3]

Synonymous spellings give us a way to detect selection pressures via preferred spellings

Comparative Genomics Tasks

- Download, store, and parse genome files
- Visualize sequences with 'genome landscape' plots
- Draw random genomes to compare to the sequenced genome

GenBank



Biopython

Task: Download and Parse GenBank files

```
LOCUS       NC_001416              48502 bp    DNA     linear   PHG 28-NOV-2007
DEFINITION  Enterobacteria phage lambda, complete genome.
ACCESSION   NC_001416
VERSION     NC_001416.1  GI:9626243
...
FEATURES             Location/Qualifiers
     source          1..48502
                     /organism="Enterobacteria phage lambda"
                     /specific_host="Escherichia coli"
                     /db_xref="taxon:10710"
     gene            191..736
                     /gene="nu1"
                     /db_xref="GeneID:2703523"
     CDS             191..736
                     /gene="nu1"
                     /codon_start=1
                     /transl_table=11
                     /product="DNA packaging protein"
                     /protein_id="NP_040580.1"
                     /db_xref="GI:9626244"
                     /db_xref="GeneID:2703523"
                     /translation="MEVNKKQLADIFGASIRTIQNWQEQGMPVLRGGGKGNEVLYDSA
                     AVIKWYAERDAEIENEKLRREVEELRQASEADLQPGTIEYERHRLTRAQADAQELKNA
                ...
ORIGIN
        1 gggcggcgac ctcgcgggtt ttcgctattt atgaaaattt tccggtttaa ggcgtttccg
       61 ttcttcttcg tcataactta atgtttttat ttaaaatacc ctctgaaaag aaaggaaacg
```

Record Information

Genomic Features

DNA Sequence

Biopython

Task: Download and Parse GenBank files

```
from Bio import GenBank

def download(accession_list):

    try:
        handle = GenBank.download_many(accession_list)
    except:
        ...

    genbank_strings = handle.read().split('//\n')
    for i in range(len(genbank_strings)):
        #Save record in file
        ...
```

Biopython modules for GenBank files

Download multiple GenBank files

Split and save

Usage

```
import genbank
genbank.download(['NC_001416'])
```

Matplotlib

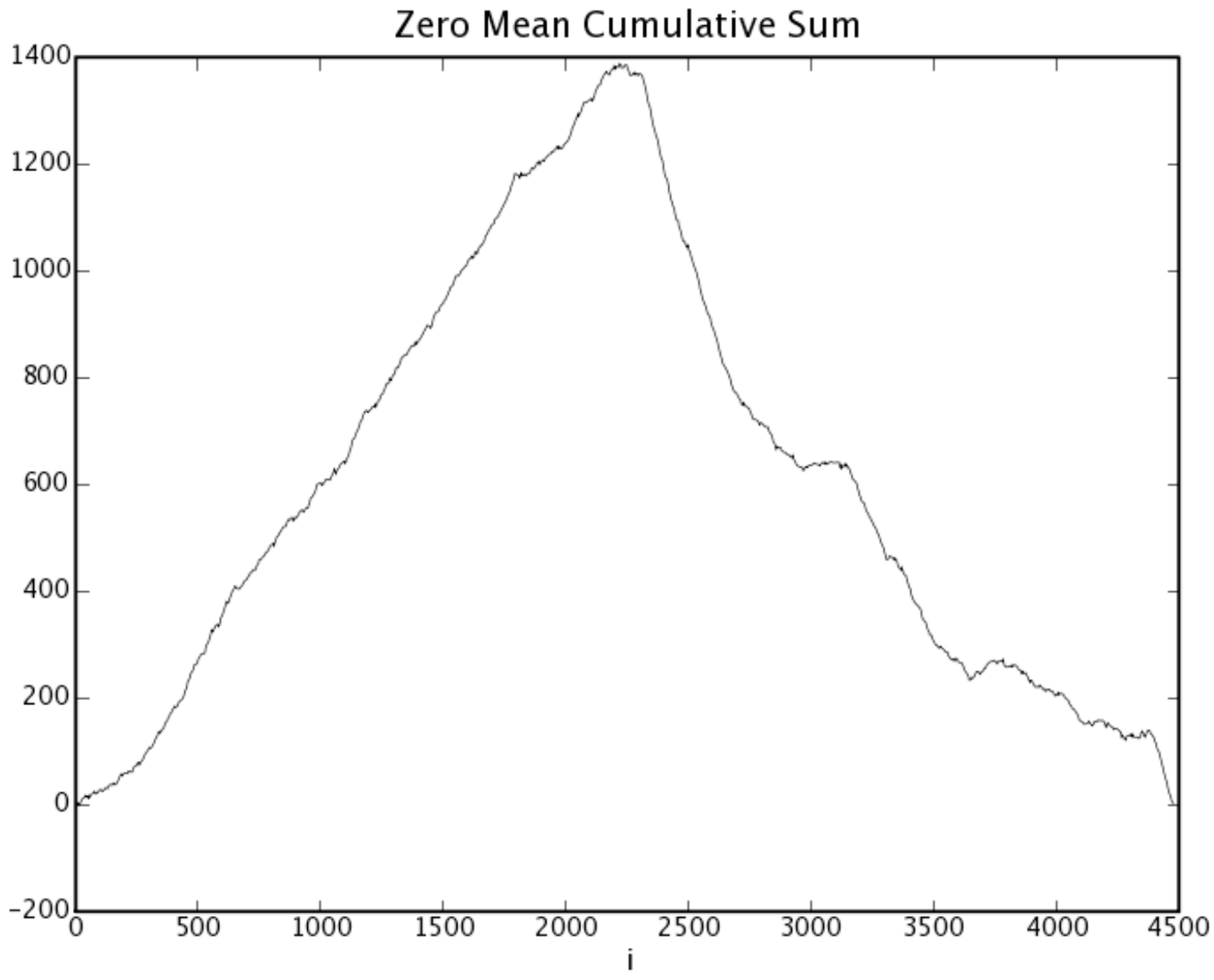
Task: Plot Genome Landscapes

DNA ·····TCTAAGACTGCT·····

$\eta(i)$ ·····0 1 0 0 0 1 0 1 0 1 1 0·····

$$\eta(i) = 1(\mathrm{GC}), 0(\mathrm{AT})$$

$$F(m) = \sum_{i=0}^{m} (\eta(i) - \overline{\eta})$$



Matplotlib

Task: Plot Genome Landscapes

```
import fileinput
import numpy
from matplotlib import pylab

def plot(filename):

    numbers = []
    for line in fileinput.input(filename):
        numbers.append(float(line.split('\n')[0]))

    mean = numpy.mean(numbers)
    cumulative_sum = numpy.cumsum([number - mean for number in numbers])

    pylab.plot(cumulative_sum[0::10],'k-')
...
```

Standard Library
Numerical
Matlab-style plotting

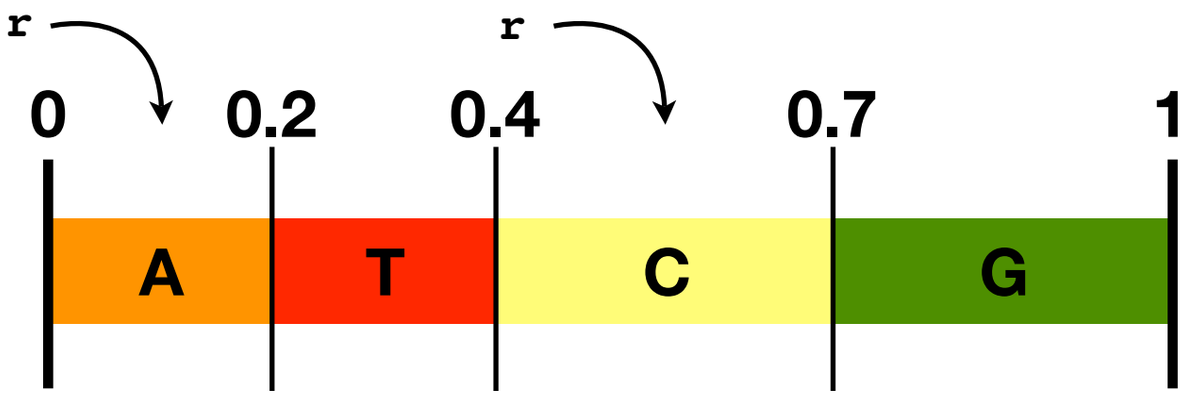
Parse input file

Cumulative sum

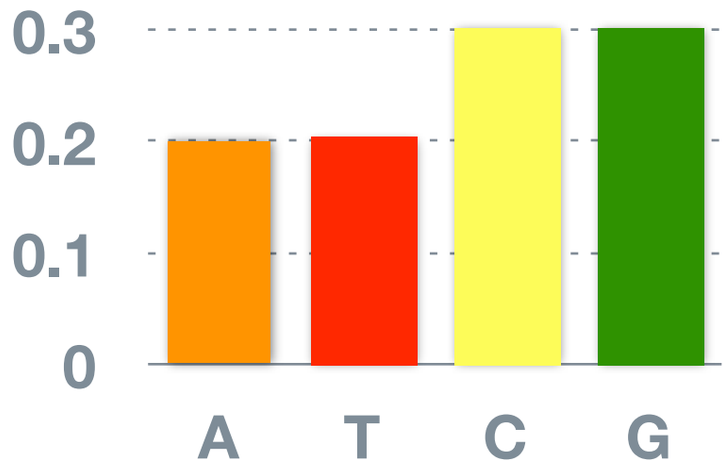
Plot

SWIG

Task: Speed up the code!



Draw random genomes according to distribution of nucleotide frequencies




```
import random

def seed():
    random.seed()

def draw(distribution):
        sum = 0
        r = random.random()
        for i in range(0,len(distribution)):
          sum += distribution[i]
          if r < sum:
              return i
```

Standard library module

Seed random number generator

Draw according to input distribution

SWIG

Step 1: Create a C implementation

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

void seed() {
    srand((unsigned) time(NULL) * getpid());
}

int draw(float distribution[4]) {
    float r= ((float) rand() / (float) RAND_MAX);
    float sum = 0.;
    int i = 0;
    for(i = 0; i < 4; i++) {
        sum += distribution[i];
        if (r < sum) {
            return i;
        }
    }
}
```

SWIG

Step 2: Create a SWIG Interface File

Module Name

Convert Python List to C-Array

```
%module c_discrete_distribution
%typemap(in) float[4](float temp[4]) {
  int i;
  if (PyList_Check($input)) {
    PyObject* input_to_tuple = PyList_AsTuple($input);
    if (!PyArg_ParseTuple(input_to_tuple,"ffff",temp,temp+1,temp+2,temp+3)) {
      PyErr_SetString(PyExc_TypeError,"tuple must have 4 elements");
      return NULL;
    }
    $1 = &temp[0];
  } else {
    PyErr_SetString(PyExc_TypeError,"expected a tuple.");
    return NULL;
  }
}

void seed();
int draw(float distribution[4]);
```

Declare interface

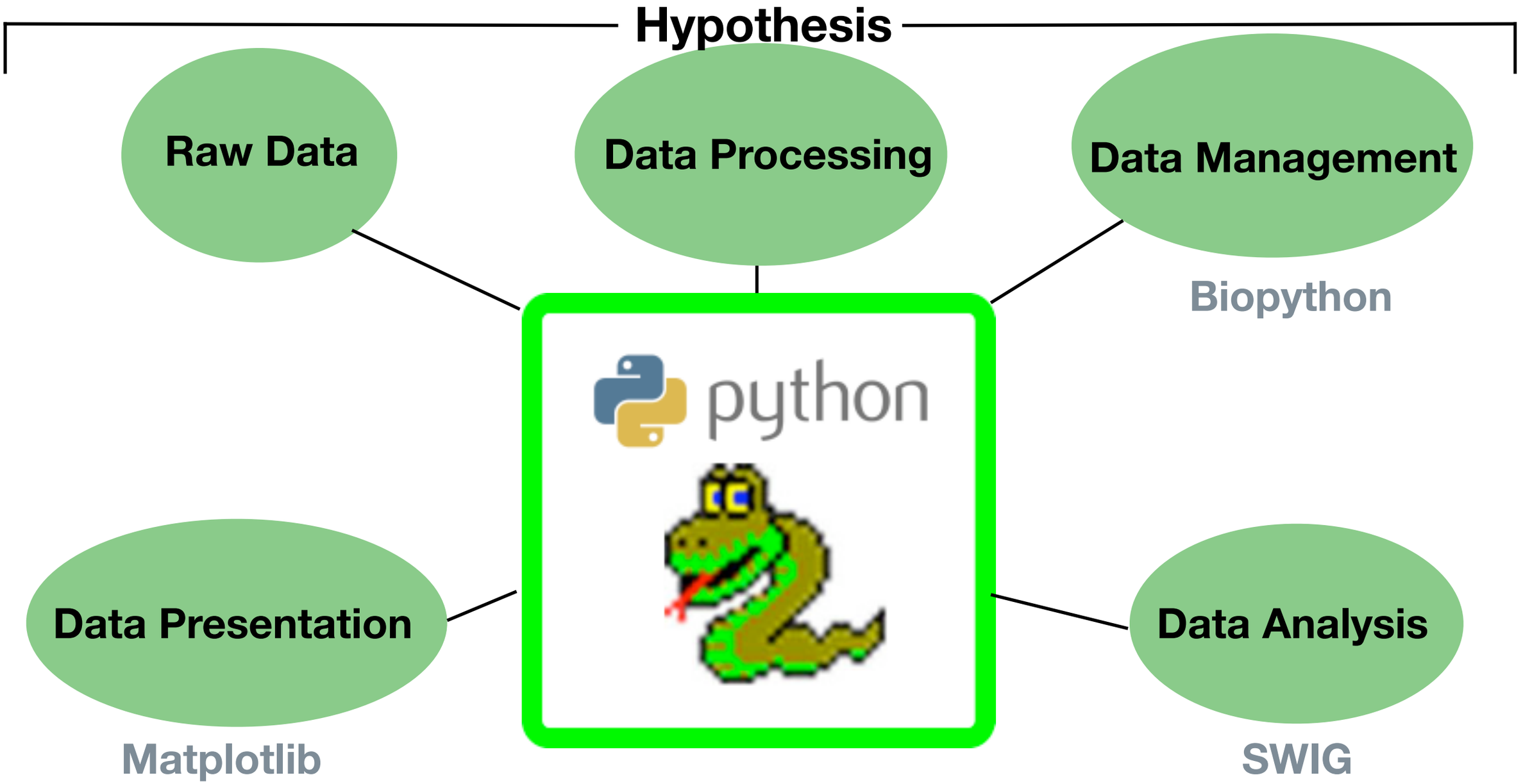
SWIG

```
import discrete_distribution

...
```



```
import c_discrete_distribution as discrete_distribution

...
```

Conclusions



References/Credits

Thank you to the Miller Institute for Basic Scientific Research

openwetware.org/wiki/python

[1] http://ccg.biology.uiowa.edu/equipment_ABI3100.php

[2] (Abizar Lakdawalla)
http://en.wikipedia.org/wiki/Image:Radioactive_Fluorescent_Seq.jpg

[3] http://www.mun.ca/biology/scarr/MGA2_03-20.html

[4] http://www.ncbi.nlm.nih.gov/Genbank/