CIS 454/554
Winter 2014

Project 2: Biopython

Due Monday Jan. 20
(submit via e-mail to conery@uoregon.edu)

The goal for this project:

- learn about software libraries used to develop bioinformatics pipelines

This project is basically a reprise of the first project, but this time you will be using Biopython or similar library to answer the questions.

Decide on a Programming Language and Computing Environment

The programming library we are going to use is defined by a group called the Open Bioinformatics Foundation, or OIBIF (www.open-bio.org). This group is an umbrella organization for implementations in various languages: BioJava, BioPerl, BioRuby, etc. You can use any implementation you like. These instructions show examples in Biopython.

Installing Biopython on Mac OS X is a bit of a pain, but should be easier on Windows or Linux. Mac users might want to consider running a virtual machine such as Cloud BioLinux (cloudbiolinux.org). I'll talk about workarounds in class this week.

Note: earlier versions of Biopython require Python 2.6, but the very latest version now allows Python 3.3.

Use a Genbank File Parser to Read the *E. coli* Genome

Section 2.4 of the online tutorial for Biopython shows how to read a Genbank file (tutorials for BioRuby and other implementations follow the same outline). This Python statement will read the *E. coli* genome and save the result in a list of records:

```
>>> recs = list(SeqIO.parse("NC_000913.gbk", "genbank"))
```

Our *E. coli* file has just one record, but other genomes may be split into two or more records, *e.g.* one for each chromosome. This will define a variable named `ecoli` to refer to the first record:

```
>>> ecoli = recs[0]
```

Now we can just call methods of the parsed object to learn about the sequence, *e.g.*

```
>>> ecoli.annotations["organism"]
'Escherichia coli str. K-12 substr. MG1655'
```

The genomic features are accessed through a method named `features`, which is a list containing an object for each feature:

```
>>> ecoli.features[0]
SeqFeature(FeatureLocation(ExactPosition(0), ExactPosition(4639675),
strand=1), type='source')

>>> ecoli.features[1]
SeqFeature(FeatureLocation(ExactPosition(189), ExactPosition(255),
strand=1), type='gene')

>>> ecoli.features[2]
SeqFeature(FeatureLocation(ExactPosition(189), ExactPosition(255),
strand=1), type='CDS')
```

Spend some time exploring the other attributes of the record, learning what you can do with individual features, *etc*.

Hint: Python's `dir` function will return a list of attributes of an object.

Exercises

Repeat the problems from the last project, but this time see if you can answer them using Biopython (or whichever version of the OIBIF libraries you use).

For interactive languages (e.g. Ruby or Python) you can answer a question by typing an expression into an interactive session or writing a small function.

1. What is the total size in bp (base pairs) of the chromosome? How did you determine this number?
2. How many CDS features are there?
3. What are the coordinates (starting and ending location) of the gene named *ileV* ?
4. *ileV* is a gene for a tRNA sequence -- note that instead of a CDS feature there is a tRNA feature. How many tRNA genes are there in all?
5. The sum of the number of CDS and tRNA features is less than the total number of genes. What other kinds of features can you find? Do they all add up to the number of things labeled as "genes"?
6. Does the *E. coli* genome contain any pseudogenes?

What to Turn In

Write up your answers and e-mail them to conery@uoregon.edu by 5:00 P.M. Monday, Jan. 20.

For each answer, copy and paste the expression you typed in an interactive session or the source code of a function or method you wrote.

I can read most document formats, but would prefer plain text, RTF, or PDF.

Note: I have a pretty aggressive spam filter. To make sure I get your mail, send it from your uoregon e-mail account (as opposed to gmail or some other free service) and include "CIS 454 Project 2" in your subject line.