Accessing NCBI's Entrez databases

using Biopython's `Bio.Entrez` module

Antonio Benítez Hidalgo

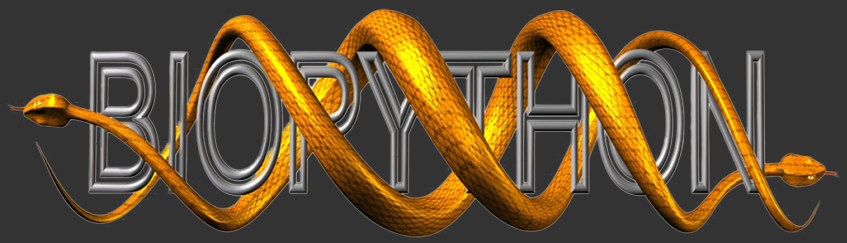
Health Engineering, Advanced Programming in Bioinformatics
University of Málaga

April 19, 2017

Overview

In a Nutshell, BioPython...

1. ...BioPython <3 Opensource
2. ...is a set of freely available tools for biological computation written in Python
3. ...is under active development!



You can find everything about how to use `Bio.Entrez` on

`http://biopython.org/DIST/docs/tutorial/Tutorial.html#htoc110`

NCBI's Guidelines

If the NCBI finds you are abusing their systems, they can **AND** will ban your access!

Gold rule: Make no more than three requests every seconds (relaxed from at most one request every three seconds in early 2009).

Use the optional email parameter so the NCBI can contact you if there is a problem.

e.g.

```
1 • from Bio import Entrez
2 • Entrez.email = "antonio.b@uma.es"  # !!
```

NCBI's Guidelines

Also, if you are using BioPython within some larger software suite, use the tool parameter to specify this.

e.g.

```
1 • from Bio import Entrez
2 • Entrez.tool = "MyLocalScript"
```

Searching the Entrez databases

ESearch.

```
handle = Entrez.esearch(db="nucleotide", term="Mycoplasma leachii complete genome")
record = Entrez.read(handle) # to parse the XML data
print(record)
{'Count': '5', 'RetMax': '5', 'RetStart': '0', 'IdList': ['313664890', '526641971', '526641931', '526641891', '312949189'], 'TranslationSet': [...], 'TranslationStack': [...], 'QueryTranslation': '("Mycoplasma leachii PG50"[Organism] OR Mycoplasma leachii pg50[All Fields]) AND clone[All Fields] AND complete[All Fields] AND genome[All Fields]'}
```

Searching the Entrez databases

ESearch.

record is an object, so you can use:

```
• print(record["IdList"])
['313664890', '526641971', '526641931', '526641891'
    , '312949189']
```

We will use EFetch to obtain more information for each of these entries IDs.

Downloading full records from Entrez

EFetch.

```
handle = Entrez.efetch(db="nucleotide", id="313664890", rettype="gb", retmode="text")
print handle.read()
```

The result is:

```
LOCUS       NC_014751            1008951 bp    DNA     circular CON 30-JUL-2015
DEFINITION  Mycoplasma leachii PG50 clone MU clone A8, complete genome.
ACCESSION   NC_014751
VERSION     NC_014751.1
DBLINK      BioProject: PRJNA224116
...
```

The arguments `retmode="text"` and `rettype="gb"` let us download this record in the GenBank format.

`retmode`

Retrieval mode. This parameter specifies the data format of the records returned, such as plain text, HMTL or XML.

`rettype`

Retrieval type. This parameter specifies the record view returned, such as Abstract or MEDLINE from PubMed, or GenPept or FASTA from protein.

Demo code

Function to get a list of id's from file.

```
def get_list(filename):
    idList = []
    with open(filename, 'r') as f:
        for line in f:
            idList.append(line.strip())   # a newline character (\n) is left at the end of the string
    return idList
```

Demo code

Function to retrieve a record from NCBI using his id number.

```
def get_record_from_id(id_code):
    handle = Entrez.efetch(db="nucleotide", id=id_code, rettype="fasta", retmode="text") # this time we use rettype="fasta"
    record = handle.read()
    handle.close()
    return record
```

Demo code

Main program.

```
if __name__ == "__main__":
    idList = get_list('idList.txt')

    for id_code in idList:
        record = get_record_from_id(id_code)
        with open(id_code+".fasta", "w") as output_fasta:
            output_fasta.write(record)
```

Demo code (2)

Better main program.

```
if __name__ == "__main__":
    idList = get_list('idList.txt')

    ids = ",".join(idList)
    records = get_record_from_id(ids)

    with open("records.fasta","w") as output_fasta:
        output_fasta.write(records)
```

Now you can use `Bio.SeqIO` module to parse the multifasta file.

...When you make a request with EFetch your list of IDs, the database etc, are all turned into a long URL sent to the server. If your list of IDs is long, this URL gets long, and long URLs can break.

Sol:
We can use `EPost` to upload a list of identifiers, which starts a new history session. You then download the records with `EFetch` by referring to the session (instead of the identifiers).

Using the history and WebEnv

EPost.

```
handle = Entrez.epost("pubmed", id=ids)
print(handle.read())
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE ePostResult PUBLIC "-//NLM//DTD epost 20090526//EN" "https://eutils.ncbi.nlm.nih..."><ePostResult>
    <QueryKey>1</QueryKey>
    <WebEnv>NCID_1_69751425_130.14.18.34_9001_1490028628_783976133_0MetA0_S_MegaStore_F_1</WebEnv>
</ePostResult>
```

```
session = Entrez.read(handle)
webenv = session["WebEnv"]
query_key = session["QueryKey"]
print(webenv + ';' + query_key)
NCID_1_53970299_..._0MetA0_S_MegaStore_F_1;1
```

Demo code (3)

The best main program.

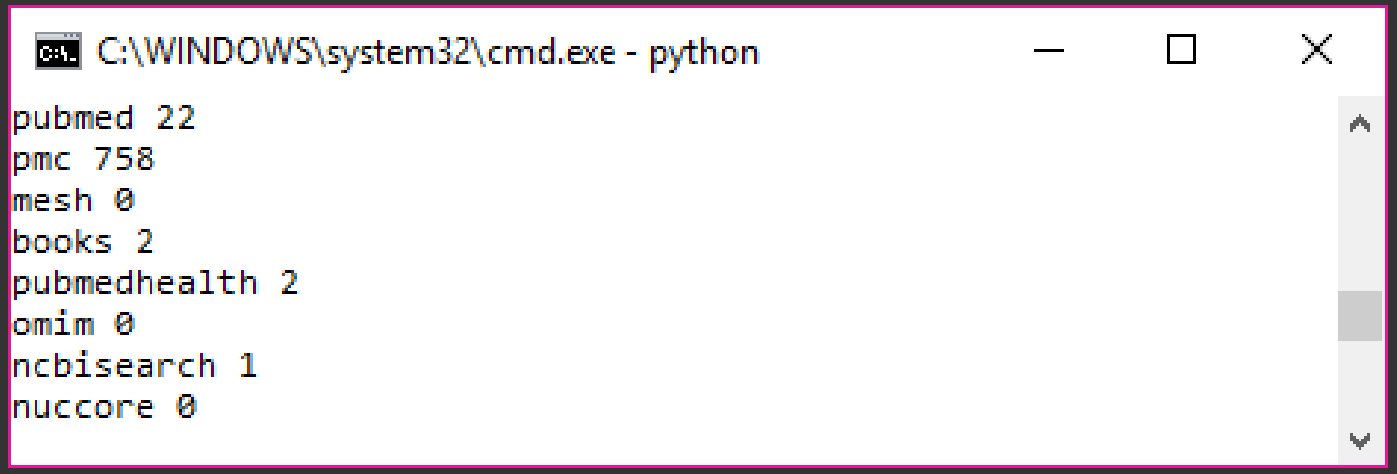
```
if __name__ == "__main__":
    idList = get_list('idList.txt')
    ids = ",".join(idList)

    handle = Entrez.epost("nucleotide", id=ids)
    session = Entrez.read(handle)

    webenv = session["WebEnv"]
    query_key = session["QueryKey"]

    records = get_record_from_id(ids,webenv,
    query_key)

    with open("records.fasta","w") as output_fasta:
        output_fasta.write(records)
```

Global Query

EGQuery.

This is particularly useful to find out how many items your search terms would find in each database without actually performing lots of separate searches with ESearch.

```
handle = Entrez.egquery(term="biopython")
search_results = Entrez.read(handle)
for result in search_results["eGQueryResult"]:
    print(result["DbName"], result["Count"])
```



Obtaining spelling suggestions

ESpell.

```
handle = Entrez.espell(term="biopythooon")
spell_suggestion = Entrez.read(handle)
spell_suggestion["Query"]
'biopythooon'
spell_suggestion["CorrectedQuery"]
'biopython'
```

To sum up.

1. `Bio.Entrez` provides users access to NCBI's databases.
2. It's written in Python, thus is easy to learn and use.
3. We need to follow some rules (as required by the NCBI).
4. If you plan to download lots of data, consider other options!


LOONEY TUNES
"That's all Folks!"

Source from Bitbucket.

You can download the code used in this example (along with this slides) from:

`https://github.com/benhid/demo_entrez_biopython.git`

Prerequisites:

1. BioPython 1.68
   `conda install -c anaconda biopython=1.68`
2. Python 3.x
3. Internet connection!!
   You can set a proxy within Python at the start of your code using `os.environ`