Biopython

Рома Курилов

Содержание

- Введение
- Работа с последовательностями
- Работа с файлами (FASTA)
- Возможности визуализации
- Разное

Биопитон?

- Бесплатная, открытая библиотека для работы с биологическими данными.
- Поддерживается организацией Open Bioinformatics Foundation.
- Международная команда разработчиков
- Выходит около четырёх релизов в год.

История

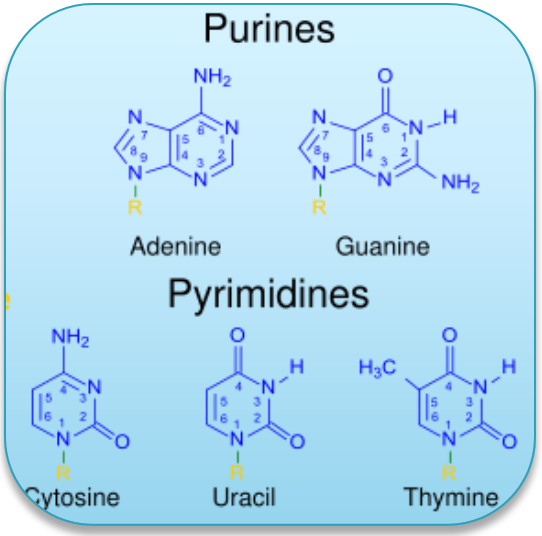
- 1999 – начало проекта (Jeff Chang & Andrew Dalke)
- 2000 – первый релиз
- 2001 – Biopython 1.00

…

- 2008 – Biopython 1.45
- 2009 – Biopython 1.50
- 2010 – Biopython 1.56, …

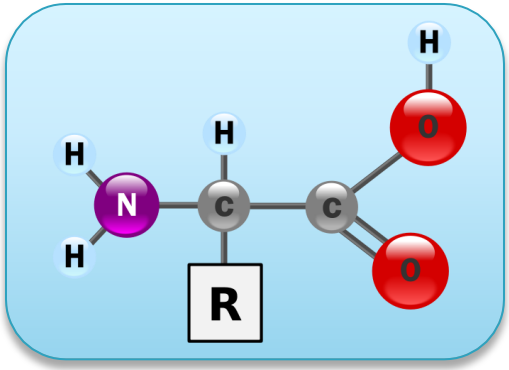
Главный Объект

Последовательность:



- нуклеотидная
  - ДНК: A, G, C, T
  - РНК: A, G, C, U

- аминокислотная
  - A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V ,W, Y



Seq object

```
>>> from Bio.Seq import Seq
>>> from Bio.Alphabet import generic_dna
>>> dna = Seq("GATCGATGGGCCTATATAGGATCGAAAATCGC", generic_dna)
>>> print dna, dna.alphabet
GATCGATGGGCCTATATAGGATCGAAAATCGC DNAAlphabet()

>>> len(dna)
32
>>> dna.count('c')
0
>>> dna.count('C')
6
>>> dna.find("TATAT")
12
>>> print dna[:12] + "-----" + dna[17:]
GATCGATGGGCC-----AGGATCGAAAATCGC
```

«Биологические» методы

- ДНК -> РНК -> Белок

```
>>> from Bio.Seq import Seq
>>> from Bio.Alphabet import generic_dna
>>> dna = Seq("GATCGATGGGCCTATATAGGATCGAAAATCGC", generic_dna)
>>> print dna, dna.alphabet
GATCGATGGGCCTATATAGGATCGAAAATCGC DNAAlphabet()

>>> print dna.complement()                    // синтез комплиментарной цепи
CTAGCTACCCGGATATATCCTAGCTTTTAGCG
>>> print dna.reverse_complement()
GCGATTTTCGATCCTATATAGGCCCATCGATC

>>> rna = dna.transcribe()                    // транскрипция
>>> print rna, rna.alphabet
GAUCGAUGGGCCUAUAUAGGAUCGAAAAUCGC RNAAlphabet()

>>> protein = rna.translate()                 // трансляция
>>> print protein, protein.alphabet
DRWAYIGSKI ExtendedIUPACProtein()
```

Работа с файлами последовательностей (parsing)

SeqIO включает 4 функции:

- parse: итеративно парсит все элементы в файле
- read: парсит одноэлементный файл и возвращает элемент
- write: записывает элементы в файл
- convert: парсит один формат и одновременно конвертирует в другой

Работа с форматом FASTA: чтение

```
>HLA:HLA00939
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
TGGCAGCGGGATGGGGAGGACCAGACCCAGGACGTGGAGCTCGTGGAGACCAGGCCTGCA
>HLA:HLA02283
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GTAGTCACTGGAGCTGCGGTCGCTGCTGTGCTGTGGAGAAAGAAGAGCTCAGATTGA
>HLA:HLA02288
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GAGACCTGGGCGGGCT…
```

```
>>> from Bio import SeqIO
>>> for rec in SeqIO.parse("C:\Users\Roma\Downloads\G_nuc.fasta", "fasta"): print rec.id,
    len(rec.seq), rec.seq[:10]+"..."
```

```
HLA:HLA00939 1017 ATGGTGGTCA...
HLA:HLA02283 1017 ATGGTGGTCA...
HLA:HLA02288 1017 ATGGTGGTCA...
HLA:HLA00942 546 GGCTCCCACT...
HLA:HLA00943 1017 ATGGTGGTCA...
HLA:HLA00944 1017 ATGGTGGTCA...
HLA:HLA00945 821 GCTCCCACTC...
```

Запись в файл

```
>HLA:HLA00939
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
TGGCAGCGGGATGGGGAGGACCAGACCCAGGACGTGGAGCTCGTGGAGACCAGGCCTGCA
>HLA:HLA02283
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GTAGTCACTGGAGCTGCGGTCGCTGCTGTGCTGTGGAGAAAGAAGAGCTCAGATTGA
>HLA:HLA02288
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GAGACCTGGGCGGGCT…
```

```
>>> from Bio import SeqIO
>>> recs = (r[:10] for r in SeqIO.parse("C:\Users\Roma\Downloads\G_nuc.fasta", "fasta“)
>>>SeqIO.write(recs, "C:\Users\Roma\Downloads\long.fasta", "fasta")
```

```
HLA:HLA00939>
ATGGTGGTCA
HLA:HLA02283>
ATGGTGGTCA
HLA:HLA02288>
ATGGTGGTCA
HLA:HLA00942>
GGCTCCCACT
```

Фильтрация

```
>HLA:HLA00939
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
TGGCAGCGGGATGGGGAGGACCAGACCCAGGACGTGGAGCTCGTGGAGACCAGGCCTGCA
>HLA:HLA02283
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GTAGTCACTGGAGCTGCGGTCGCTGCTGTGCTGTGGAGAAAGAAGAGCTCAGATTGA
>HLA:HLA02288
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GAGACCTGGGCGGGCT…
```

```
>>> from Bio import SeqIO
>>> recs = (r for r in SeqIO.parse("C:\Users\Roma\Downloads\G_nuc.fasta", "fasta") if len(r)>20)
>>> SeqIO.write(recs, "C:\Users\Roma\Downloads\long.fasta", "fasta")
46
```

```
>HLA:HLA00939
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
TGGCAGCGGGATGGGGAGGACCAGACCCAGGACGTGGAGCTCGTGGAGACCAGGCCTGCA
>HLA:HLA02283
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GTAGTCACTGGAGCTGCGGTCGCTGCTGTGCTGTGGAGAAAGAAGAGCTCAGATTGA
>HLA:HLA02288
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
ATGGTGGTCATGGCGCCCCGAACCCTCTTCCTGCTGCTCTCGGGGGCCCTGACCCTGACC
GAGACCTGGGCGGGCT…
```

Преобразование формата файла

- from Bio import SeqIO
  recs = SeqIO.parse("roche.sff", "sff")
  SeqIO.write(recs, "reads.fastq", "fastq")

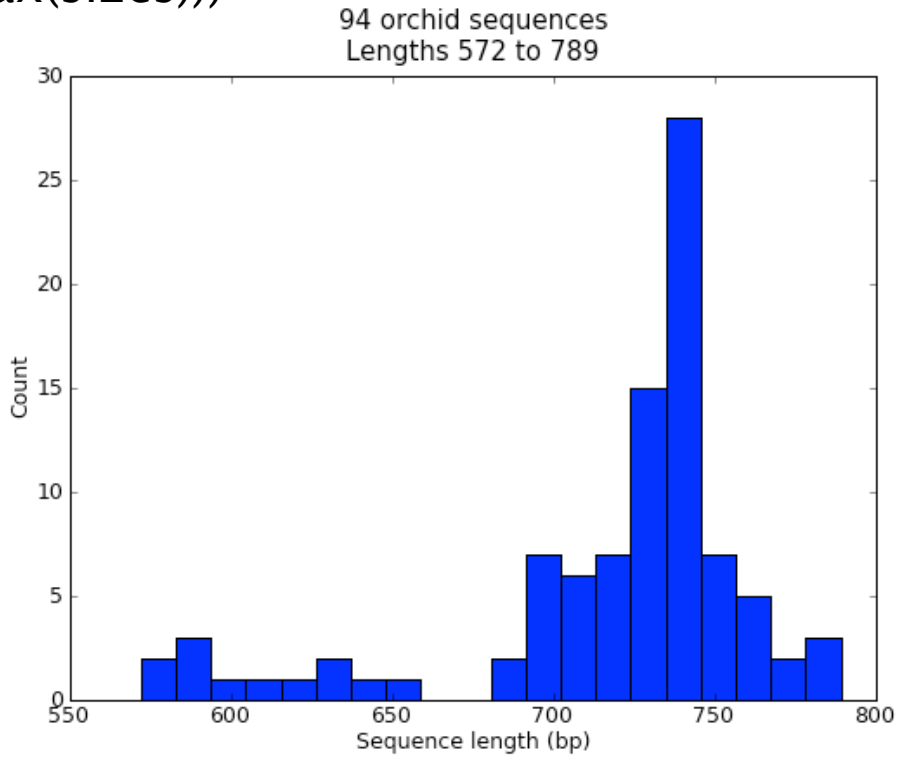
- from Bio import SeqIO
  SeqIO.convert("roche.sff", "sff", "reads.fastq", "fastq")

Анализ содержимого файла

```
>>> from Bio import SeqIO
>>> sizes = [len(r) for r in SeqIO.parse("C:\Users\Roma\Downloads\G_nuc.fasta", "fasta")]
>>> len(sizes), min(sizes), max(sizes)
(46, 546, 1017)
>>> sizes
[1017, 1017, 1017, 1017, 1017, 1017, 1017, 1017, 1017, 1017, 546, 1017, 1017, 821, 1017, 1017,
    895, 1017, 546, 822, 822, 822, 822, 822, 822, 1017, 945, 1017, 1017, 821, 1017, 1017, 822,
    1016, 1017, 1017, 1017, 546, 546, 546, 822, 822, 822, 822, 822, 546]
```

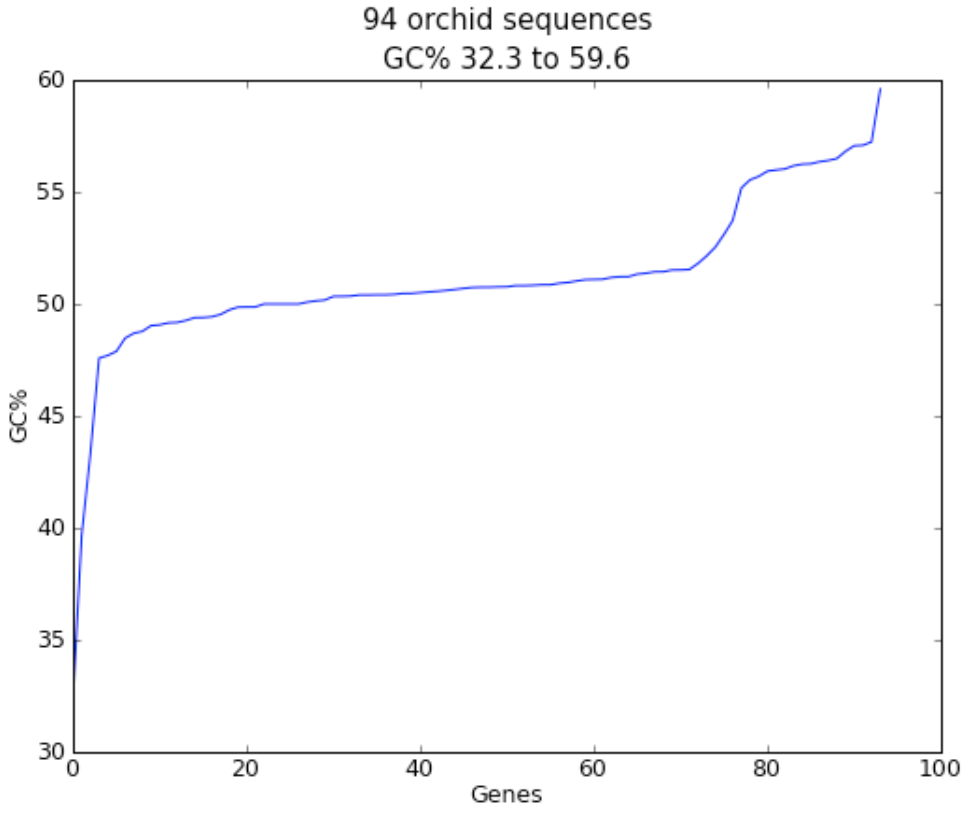
Анализ и визуализация

```
from Bio import SeqIO
sizes = [len(r) for r in SeqIO.parse("ls_orchid.fasta", "fasta")]
import pylab
pylab.hist(sizes, bins=20)
pylab.title("%i orchid sequences\nLengths %i to %i" \
           % (len(sizes),min(sizes),max(sizes)))
pylab.xlabel("Sequence length (bp)")
pylab.ylabel("Count")
pylab.show()
```



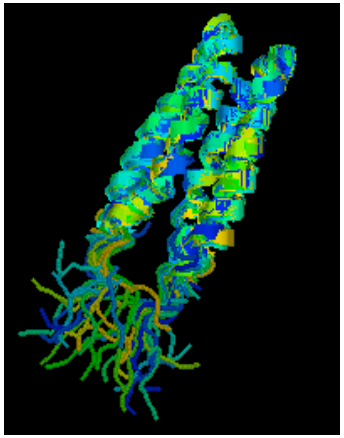
Анализ и визуализация

```
from Bio import SeqIO
from Bio.SeqUtils import GC
val = sorted(GC(r.seq) for r in SeqIO.parse("ls_orchid.fasta","fasta"))
import pylab
pylab.plot(val)
pylab.title("%i orchid sequences\
% (len(val), min(val), max(val)))
pylab.xlabel("Genes")
pylab.ylabel("GC%")
pylab.show()
```

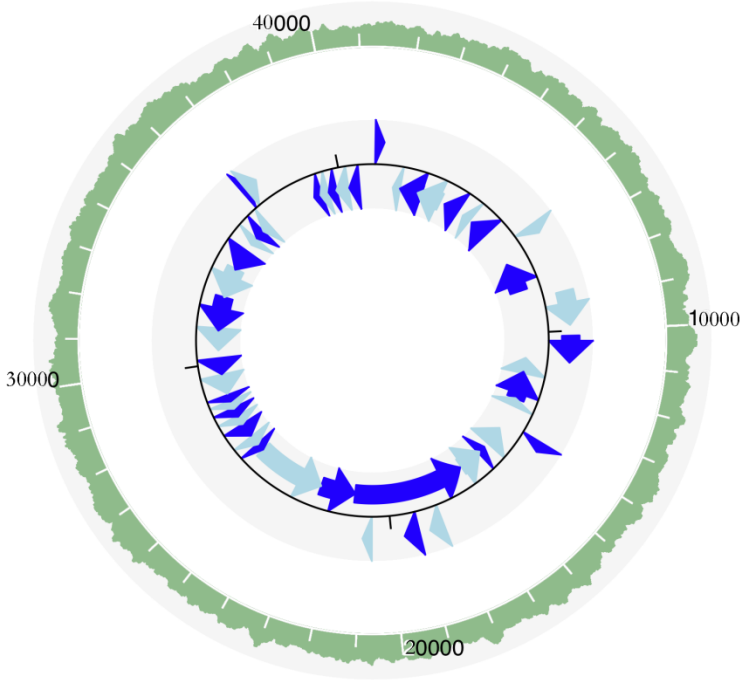


Визуализация

- Пространственное выравнивание белков



- Сборка геномов



Ещё возможности:

- Работа с BLAST, базами данных NCBI
- Множественное выравнивание последовательностей
- Кластерный анализ (Bio.cluster)
- Филогенетика (Bio.Phylo, Bio.Nexus)
- Популяционная генетика (Bio.PopGen)