TODAY'S TOPICS:

- Objects and Methods Review
- Sequence File Formats Review
- BioPerl Modules
  - Bio::Seq
  - Bio::SeqIO

MODULE = (OBJECTS, METHODS)

- **Objects provide methods that allow you to view or change attributes stored in an instance of an object.**
- **For example, the Bio::Seq object can store a sequence, accession number, alphabet (DNA/RNA/protein), description, features, etc., as attributes.**
- **Without methods you could not see the values of the attributes in an object!** ***The only sensible thing you can do with an object is to call one of its methods.***
- **Example:**

```
$seqObj = $seqIOobj->next_seq;
$acc_num = $seqObj->accession_number;
$desc = $seqObj->description;
```

SEQUENCE FILE FORMATS

```
>YMR224C  MRE11 yeast gene
ATGGACTATCCTGATCCAGACACAATAAGGATTTTAATTACTACAGATAATCATGTGGGTTACAACG
AAAATGATCCCATTACTGGCGATGATTCTTGGAAAACTTTCCATGAAGTCAGGTGCCATTATTATTT
CAGAA
```

SEQUENCE FILE FORMATS

```
>YMR224C  MRE11 yeast gene
ATGGACTATCCTGATCCAGACACAATAAGGATTTTAATTACTACAGATAATCATGTGGGTTACAACG
AAAATGATCCCATTACTGGCGATGATTCTTGGAAAACTTTCCATGAAGTCAGGTGCCATTATTATTT
CAGAA

>YNL250W  Rad50p
ATGAGCGCTATCTATAAATTATCTATTCAGGGCATACGGTCTTTTGACTC
CAATGATAGGGAAACTATTGAATTTGGCAAGCCTCTGACTTTAATAGTCG
GCATGAATGGTTCAGGTAAACGACTATCATCGAATGTTTAAAGTACGCTA
CCACAGGTGATCTACCGCCCAACAGAAGGGAGGAGTATTCATTCATGACC
CGAAGATAACTGGTGAAAAGGACATTAGAGCTCAGGTAAACTGGCGTTTA
CGAGTGCCAATGGACTCAATATGATTGTCACCAGAAATATTCAGTTG
```

SEQUENCE FILE FORMATS

```
>YMR224C  MRE11 yeast gene
ATGGACTATCCTGATCCAGACACAATAAGGATTTTAATTACTACAGATAATCATGTGGGTTACAACG
AAAATGATCCCATTACTGGCGATGATTCTTGGAAAACTTTCCATGAAGTCAGGTGCCATTATTATTT
CAGAA

>YNL250W  Rad50p
ATGAGCGCTATCTATAAATTATCTATTCAGGGCATACGGTCTTTTGACTC
CAATGATAGGGAAACTATTGAATTTGGCAAGCCTCTGACTTTAATAGTCG
GCATGAATGGTTCAGGTAAACGACTATCATCGAATGTTTAAAGTACGCTA
CCACAGGTGATCTACCGCCCAACAGAAGGGAGGAGTATTCATTCATGACC
CGAAGATAACTGGTGAAAAGGACATTAGAGCTCAGGTAAACTGGCGTTTA
CGAGTGCCAATGGACTCAATATGATTGTCACCAGAAATATTCAGTTG
```

- The Bio::SeqIO module has a method that reads one whole sequence record at a time from a file. It figures out where one sequence ends and the next begins automatically.

DO WE NEED OBJECTS?

- **A DNA, RNA, or protein sequence could be stored as a string, but only the sequence itself is captured:**

```
my $protein = 'MSDLAPNDARGEETAQSVAPSDVLEDP';
```

DO WE NEED OBJECTS?

- A DNA, RNA, or protein sequence could be stored as a string, but only the sequence itself is captured:

```
my $protein = 'MSDLAPNDARGEETAQSVAPSDVLEDP';
```

- Other associated properties would have to be stored as separate items, e.g. `$gi, $seqname, $seqlen, $accession,`…

DO WE NEED OBJECTS?

- A DNA, RNA, or protein sequence could be stored as a string, but only the sequence itself is captured:

```
    my $protein = 'MSDLAPNDARGEETAQSVAPSDVLEDP';
```

- Other associated properties would have to be stored as separate items, e.g. $gi, $seqname, $seqlen, $accession,…

- A Bio::Seq object can store the sequence along with many properties or attributes. Example:

```
  use Bio::Seq;
  my $seqObj = Bio::Seq->new(-alphabet => 'protein',
       -seq => 'MSDLAPNDARGEETAQSVAPSDVLEDP',
    -display_id => 'CCAP 1055/1',
    -description => 'predicted protein',
    -accession_number => 'XP_002181413');
```

SEQUENCE FILE FORMATS

```
LOCUS  SCU60829  2303 bp  DNA linear  PLN  24-JUN-1997
DEFINITION Saccharomyces cerevisiae Mre11p (MRE11) gene, complete cds.
ACCESSION U60829
VERSION U60829.1  GI:2209264
KEYWORDS .
SOURCE Saccharomyces cerevisiae (baker's yeast)
ORGANISM Saccharomyces cerevisiae
…
FEATURES Location/Qualifiers
 source   1..2303
          /organism="Saccharomyces cerevisiae"
          /mol_type="genomic DNA"
          /strain="SK1"
          /db_xref="taxon:4932"
          /chromosome="XIII"
   gene   168..2246
          /gene="MRE11"
   CDS    168..2246
          /gene="MRE11"
          /note="SK1 derived coding sequence; recombination enzyme"
…
ORIGIN
 1        aaactgactt aaggtttaaa tagtatggcc aatcgaatag aacccaaaca ttatagccat
61        attaaattac tctttacgct tgtaaggaag acaatgtgga aacaacatta agagaatgca
```

BIOPERL MODULES

Bio::Seq #includes a “sequence” object

(sequence + attributes)

Bio::SeqIO # “sequence I/O” object

(e.g. connection to a sequence file)

Bio::SearchIO # “BLAST I/O” object

(e.g. BLAST output)

Bio::DB::Genbank # “Genbank connection” object

(e.g. to run a remote GenBank query)

BIOPERL MODULES

Bio::Seq #includes a “sequence” object
(sequence + attributes)

Bio::SeqIO # “sequence I/O” object
(e.g. connection to a sequence file)

Bio::SearchIO # “BLAST I/O” object
(e.g. BLAST output)

Bio::DB::Genbank # “Genbank connection” object
(e.g. to run a remote GenBank query)

Documentation can be found at
http://doc.bioperl.org/

BIO::SEQ OBJECT

- The Bio::Seq Object can store a description, accession, version, alphabet, species, and features.

BIO::SEQ OBJECT

- The Bio::Seq Object can store a description, accession, version, alphabet, species, and features.

```
LOCUS    ECORHO    1880 bp    DNA linear    BCT 26-APR-1993
DEFINITION  E.coli rho gene coding for transcription termination factor.
ACCESSION  J01673
VERSION  J01673.1 GI:147605
…
ORGANISM  Escherichia coli
    Bacteria; Proteobacteria; Gammaproteobacteria; Enterobacteriales;
 Enterobacteriaceae; Escherichia.
…
ORIGIN 15 bp upstream from HhaI site.
 1        aaccctagca ctgcgccgaa atatggcatc cgtggtatcc cgactctgct gctgttcaaa
61        aacggtgaag tggcggcaac caaagtgggt gcactgtcta aaggtcagtt gaaagagttc
…
```

CREATING BIO::SEQ OBJECTS

- **Bio::Seq new() method**

CREATING BIO::SEQ OBJECTS

- Bio::Seq new() method

or

- Create a Bio::SeqIO object connected to a file, then call the next_seq() method, which returns a Bio::Seq object.

CREATING BIO::SEQ OBJECTS

- **Bio::Seq new() method**

or

- **Create a Bio::SeqIO object connected to a file, then call the next_seq() method, which returns a Bio::Seq object.**

```
use Bio::SeqIO;

$Seq_obj1 = Bio::Seq->new(    -seq => "cagcag",
                        -display_id => "TinySeq");
                              -desc => "six bases");
```

or

```
$SeqIO_obj = Bio::SeqIO->new( -file => ‘file.fa’,
                              -format => ‘fasta’ );

$Seq_obj2 = $SeqIO_obj->next_seq();
```

METHOD RETURNS WHAT?

- The following Bio::Seq methods return a *string:*

```
$seqobj->display_id();
$seqobj->seq();
$seqobj->accession_number();
$seqobj->alphabet();
```

METHOD RETURNS WHAT?

- The following Bio::Seq methods return a *string:*

```
$seqobj->display_id();
$seqobj->seq();
$seqobj->accession_number();
$seqobj->alphabet();
```

What about these?

```
$seqobj->display_id("newname");
$seqobj->subseq(50,100);
```

METHOD RETURNS WHAT?

- The following Bio::Seq methods return a *string:*

```
$seqobj->display_id();
$seqobj->seq();
$seqobj->accession_number();
$seqobj->alphabet();
```

What about these?

```
$seqobj->display_id("newname");
$seqobj->subseq(50,100);
```

- The following return a *Seq object:*

```
$seqobj->trunc(50,100);
 $seqobj->revcom;
 $seqobj->translate;
```

THE BIO::SEQIO OBJECT

- **Creating a Bio::SeqIO object:**

```
$InSeqIO = Bio::SeqIO->new(-file => "$infile",
                          -format => ‘Genbank’);

$OutSeqIO = Bio::SeqIO->new(-file => ">$infile.FASTA",
                           -format => ‘FASTA’);
```

THE BIO::SEQIO OBJECT

- Creating a Bio::SeqIO object:

```
$InSeqIO = Bio::SeqIO->new(-file => "$infile",
                          -format => ‘Genbank’);

$OutSeqIO = Bio::SeqIO->new(-file => ">$infile.FASTA",
                           -format => ‘FASTA’);
```

- Can do either *input* or *output*. Remember the “>“!

THE BIO::SEQIO OBJECT

- Creating a Bio::SeqIO object:

```
$InSeqIO = Bio::SeqIO->new(-file => "$infile",
                          -format => ‘Genbank’);

$OutSeqIO = Bio::SeqIO->new(-file => ">$infile.FASTA",
                           -format => ‘FASTA’);
```

- Can do either *input* or *output*. Remember the “>“!
- The Bio::SeqIO->new() method simply instantiates a connection – no I/O has been done!
- *next_seq*, *write_seq* methods do the I/O

FORMAT CONVERSION

- Bio::SeqIO recognizes many formats: FASTA, Genbank, EMBL, etc.

(check perldoc Bio::SeqIO). From Fasta to EMBL:

```
use Bio::SeqIO;

$in  = Bio::SeqIO->new( -file => "infilename" ,
                      -format => 'Fasta');
$out = Bio::SeqIO->new( -file => ">outfilename" ,
                      -format => 'EMBL');

while ( my $seq = $in->next_seq() ) {
       $out->write_seq($seq);
}
```

HOMEWORK

- Next week we'll review Bio::SeqIO and explore Bio::SearchIO for parsing BLAST output

Homework 8, on Pi and Theta, due Tuesday, Nov 6th

Check answers in Table 1 of the paper (at least for mean values!)

BioPerl Quiz on Tuesday

- Read the BioPerl HOWTO:SearchIO:

http://www.bioperl.org/wiki/HOWTO:SearchIO