ECOL 553L

Section 5
Command line tools for Genomic Data Analysis

Turning in your homework

Turning in your homework

- Make a folder in your home directory named `ecol553_homework`

Turning in your homework

- Make a folder in your home directory named `ecol553_homework`
- within this folder, make a folder called `homework1`

Turning in your homework

- Make a folder in your home directory named `ecol553_homework`
- within this folder, make a folder called `homework1`
- make a file within this folder called *`<last_name>_<first_initial>`*`.txt` and put the homework answers in it

Turning in your homework

- Make a folder in your home directory named `ecol553_homework`
- within this folder, make a folder called `homework1`
- make a file within this folder called *`<last_name>_<first_initial>`*`.txt` and put the homework answers in it
- when finished move back to `~/ecol553_homework` and run `turnin homework1`

Turning in your homework

- Make a folder in your home directory named `ecol553_homework`
- within this folder, make a folder called `homework1`
- make a file within this folder called `<last_name>_<first_initial>.txt` and put the homework answers in it
- when finished move back to `~/ecol553_homework` and run `turnin homework1`

If you run this more than once, it will overwite your old submittion

Web BLAST

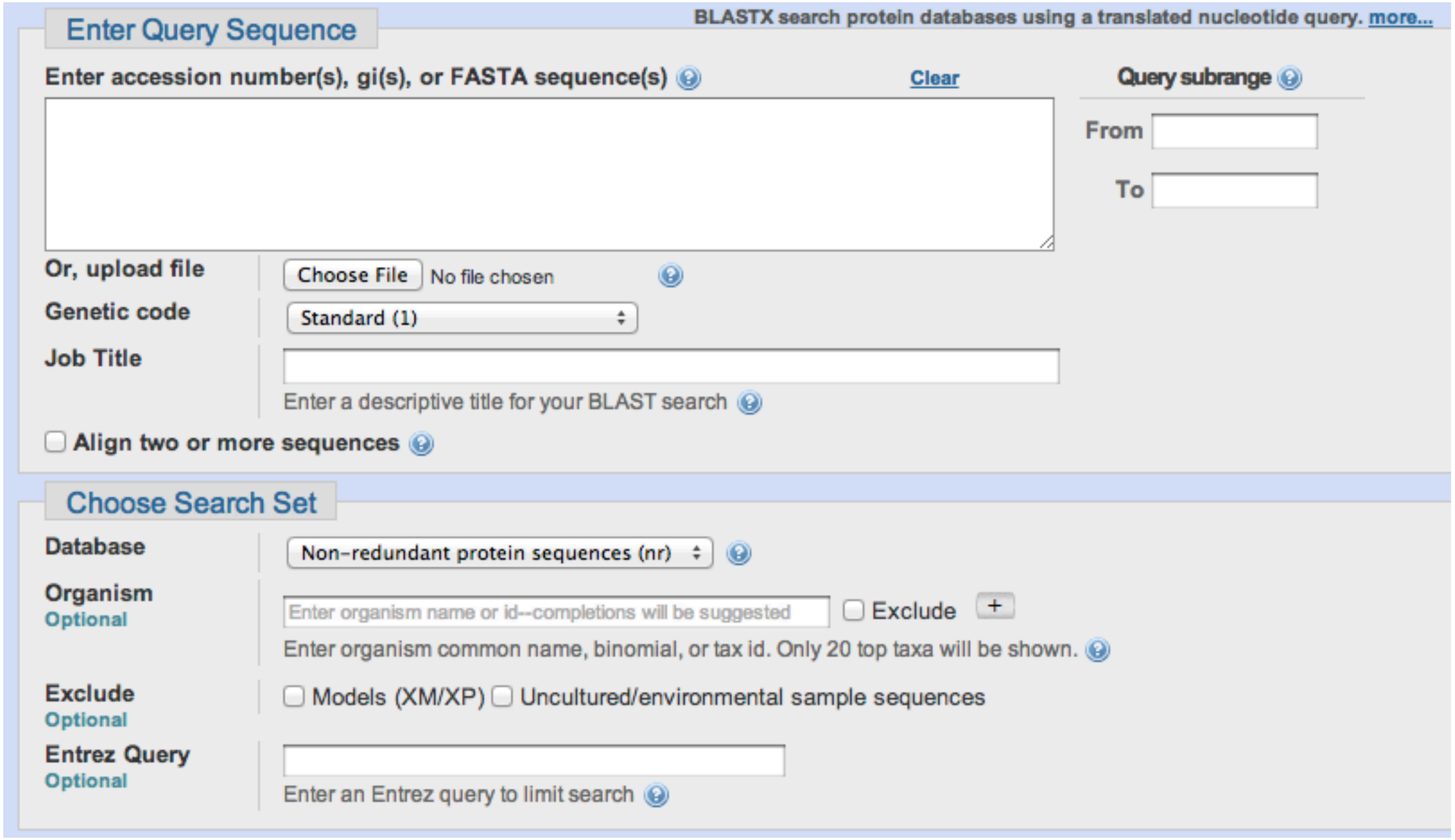
nucleotide blast	Search **nucleotide** database using a **nucleotide** query *Algorithms:* blastn, megablast, discontiguous megablast
protein blast	Search **protein** database using a **protein** query *Algorithms:* blastp, psi-blast, phi-blast, delta-blast
blastx	Search **protein** database using a **translated nucleotide** query
tblastn	Search **translated nucleotide** database using a **protein** query
tblastx	Search **translated nucleotide** database using a **translated nucleotide** query

Web BLAST

- First you have to pick the blast protocol

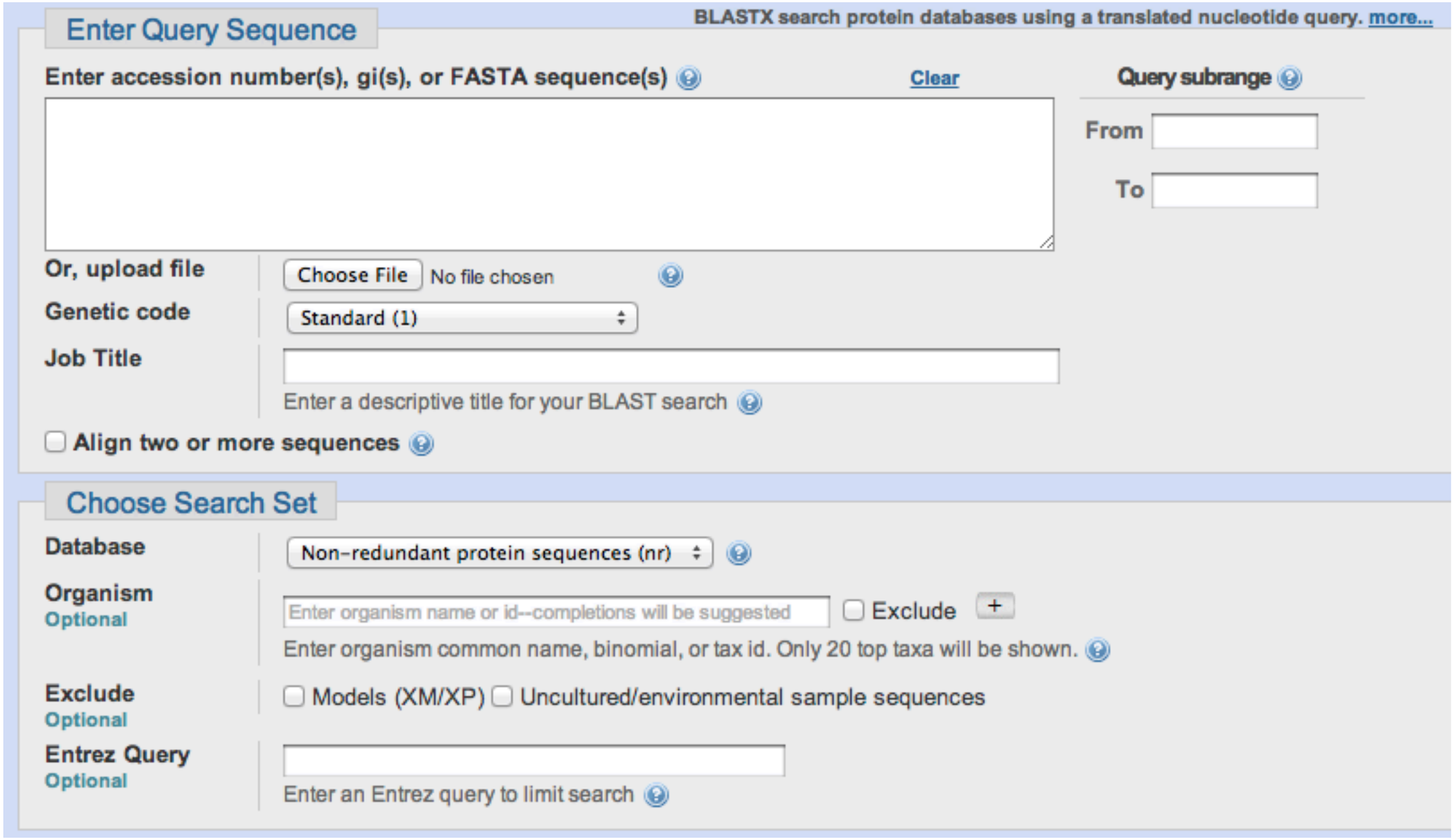
nucleotide blast	Search **nucleotide** database using a **nucleotide** query *Algorithms:* blastn, megablast, discontiguous megablast
protein blast	Search **protein** database using a **protein** query *Algorithms:* blastp, psi-blast, phi-blast, delta-blast
blastx	Search **protein** database using a **translated nucleotide** query
tblastn	Search **translated nucleotide** database using a **protein** query
tblastx	Search **translated nucleotide** database using a **translated nucleotide** query

Web BLAST

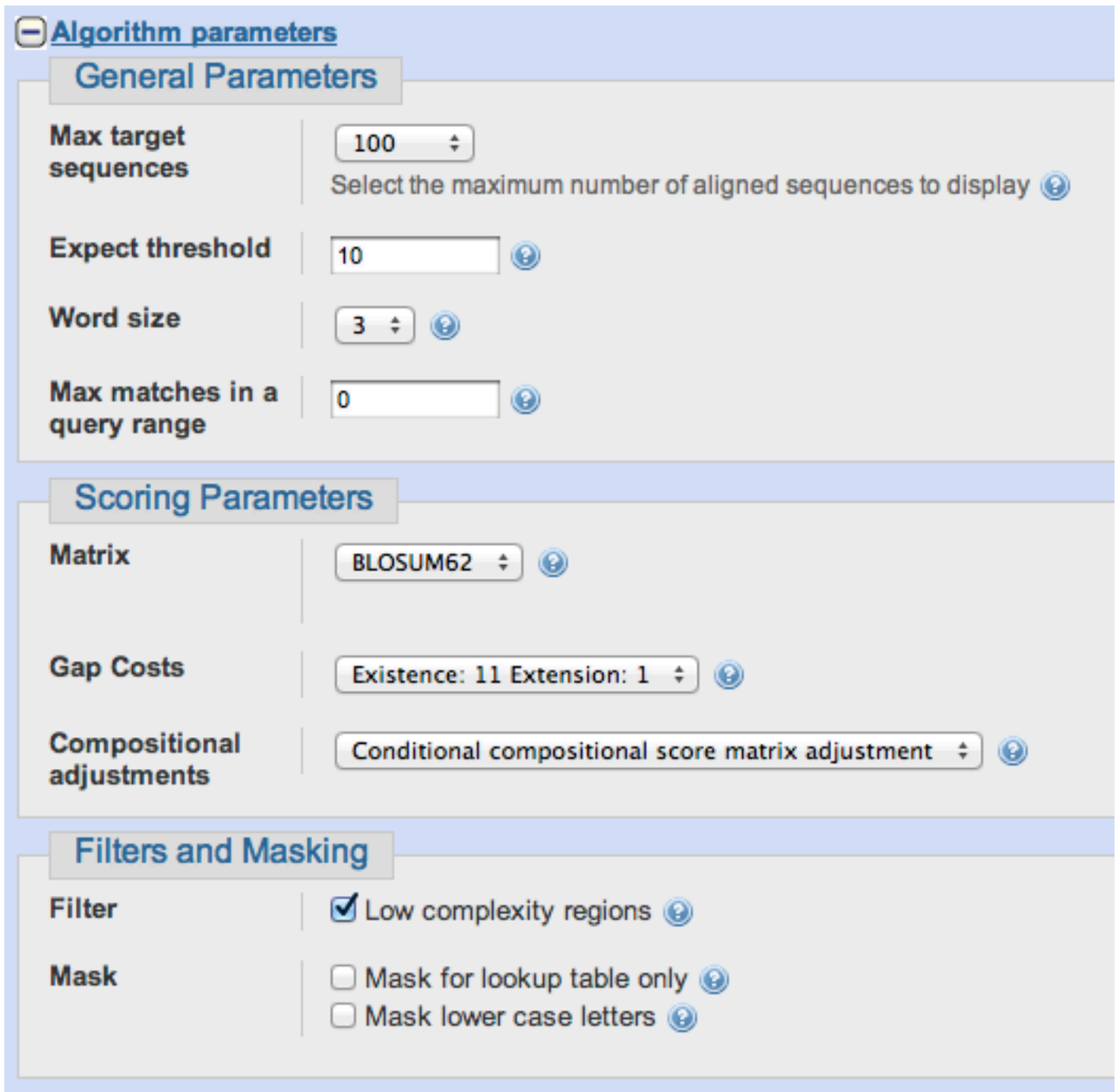

Enter Query Sequence
BLASTX search protein databases using a translated nucleotide query. more...
Enter accession number(s), gi(s), or FASTA sequence(s)
Clear
Query subrange
From
To
Or, upload file
Choose File
No file chosen
Genetic code
Standard (1)
Job Title
Enter a descriptive title for your BLAST search
Align two or more sequences
Choose Search Set
Database
Non-redundant protein sequences (nr)
Organism
Optional
Enter organism name or id--completions will be suggested
Exclude
+
Enter organism common name, binomial, or tax id. Only 20 top taxa will be shown.
Exclude
Optional
Models (XM/XP)
Uncultured/environmental sample sequences
Entrez Query
Optional
Enter an Entrez query to limit search

Web BLAST

- Input our data

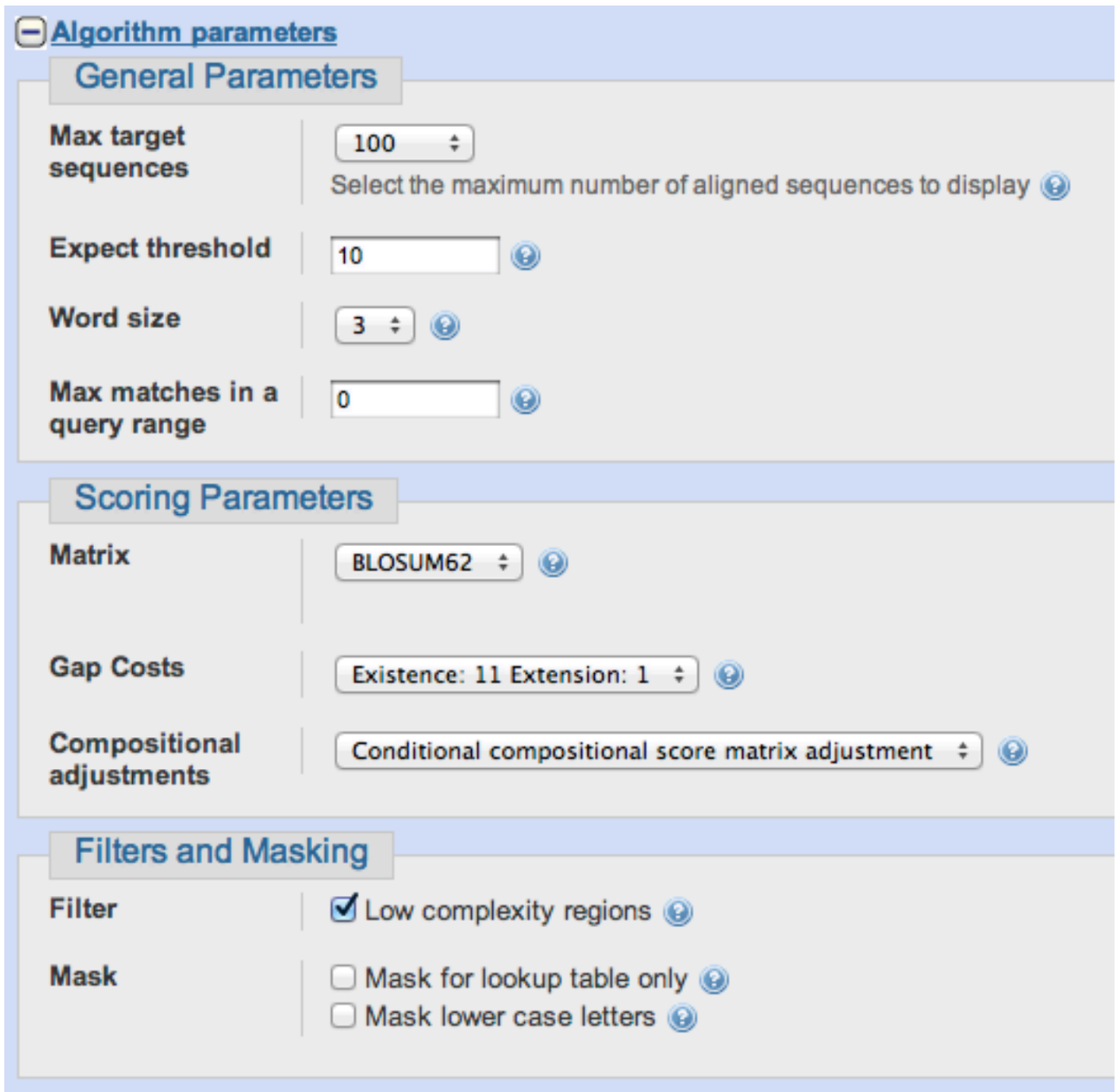


Web BLAST


Algorithm parameters
General Parameters
Max target sequences
100
Select the maximum number of aligned sequences to display
Expect threshold
10
Word size
3
Max matches in a query range
0
Scoring Parameters
Matrix
BLOSUM62
Gap Costs
Existence: 11 Extension: 1
Compositional adjustments
Conditional compositional score matrix adjustment
Filters and Masking
Filter
Low complexity regions
Mask
Mask for lookup table only
Mask lower case letters

Web BLAST

- Then you have to change the settings



DO THIS NO MORE

DO THIS NO MORE

- Command line blast allows you to

DO THIS NO MORE

- Command line blast allows you to
  - Save Settings

DO THIS NO MORE

- Command line blast allows you to
  - Save Settings
  - Never Upload Your Data

DO THIS NO MORE

- Command line blast allows you to
  - Save Settings
  - Never Upload Your Data
  - Better Analyze Your Data

DO THIS NO MORE

- Command line blast allows you to
  - Save Settings
  - Never Upload Your Data
  - Better Analyze Your Data
  - Not Know How Long Till It Starts/Finishes Running

DO THIS NO MORE

- Command line blast allows you to
  - Save Settings
  - Never Upload Your Data
  - Better Analyze Your Data
  - Not Know How Long Till It Starts/Finishes Running
  - Use A Web Browser (or your mouse)

Modules on the HPC

Modules on the HPC

- For organizational purposes, different categories of software are installed in different directories on the UA HPC systems

Modules on the HPC

- For organizational purposes, different categories of software are installed in different directories on the UA HPC systems
- Examples: Java, Matlab, the NCBI blast suite, blat, EMBOSS, R, clustalw, muscle, various compilers
- Modules are preinstalled programs available to everyone by not loaded every time you log in

Modules on the HPC

- For organizational purposes, different categories of software are installed in different directories on the UA HPC systems
- Examples: Java, Matlab, the NCBI blast suite, blat, EMBOSS, R, clustalw, muscle, various compilers Modules are preinstalled programs available to everyone by not loaded every time you log in
- You access them using the `module` program

Modules on the HPC

- For organizational purposes, different categories of software are installed in different directories on the UA HPC systems
- Examples: Java, Matlab, the NCBI blast suite, blat, EMBOSS, R, clustalw, muscle, various compilers
  Modules are preinstalled programs available to everyone by not loaded every time you log in
  - You access them using the `module` program
  - `module avail` shows the available packages

Modules on UPG

```
deblasio@service2(login):~/ParamAdvising/core_column_svm> module avail

------------------------------------------------------ /usr/share/Modules/modulefiles ------------------------------
chkfeature  dot         module-cvs  module-info modules     mpt/2.04    null        perfboost   perfcatcher use.own

------------------------------------------------------------ /uaopt/modulefiles ------------------------------------
R/2.14.1                        exonerate/2.2                   lapack/3.4.0                    quake/0.3.0
abaqus/6.11                     fastqc/0.10.0                   lastz/1.02.00                   raxml/7.2.8
abaqus/6.9                      fftw/2.1.5                      mafft/6.923                     roche454/2.6
abyss/1.3.2                     fftw/3.3                        mathematica/8.0                 ruby/1.9.3-p0
abyss/1.3.4-kmer64              gaussian/g03-E.01               matlab/r2012a                   samtools/0.1.18
abyss/1.3.4-kmer96              gaussian/g09                    mcnp5                           seqtools/4.13.2-2-g0a83
ansys/14.0                      gnuplot/4.6.0                   mcnpx/2.6.0                     soapdenovo/1.05
apbs/1.3                        gotoblas2                       meep/1.1.1                      sparsehash/1.12
augustus/2.6.1                  grace/5.1.22                    mercurial/2.2                   tandemrepeatsfinder/4.04
autodock/4.3.2                  graphviz/2.28.0                 mothur/1.23.1                   tex/2011
bamtools/1.0.2                  gromacs/4.5.5                   mothur/1.26.0                   tophat/2.0.4
blas                            gsl/1.15                        mpich2/1.4.1p1                  trans-abyss/1.3.2
blast/ncbi-2.2.26               hdf5/1.8.8                      mrbayes/3.2.1                   trinity/r2012-03-17
blat/34                         hmmer/3.0                       muscle/3.8.31                   trinity/r2012-05-18
boost/1.48.0                    intel/2012.0.032                ncl/6.0.0                       vasp/4.6
bowtie2/2.0.0-b5                intel-mpi/2012.0.032(default)   netcdf/4.1.3                    vasp/5.2
bwa/0.6.1                       intel-mpi/2012.0.032-ia32       netcdf/4.1.3-intel              velvet/1.2.02
cd-hit/4.5.4                    irods/3.0                       nwchem/6.0                      velvet/1.2.07
clustalw/2.1                    java/1.6.0                      oases/0.2.08                    xtandem/10-12-01-1
cmake/2.8.9                     java/1.7.0                      openmpi/1.4.4
emboss/6.4.0                    lammps/20apr12                  perl/5.14.2
espresso/4.3.2                  lammps/30aug12                  python/2.7.3
```

- `module avail` shows the available packages

Modules on the HPC

```
module load <package>
```

Modules on the HPC

- To load a module and use it, use

```
module load <package>
```

Modules on the HPC

- To load a module and use it, use

```
module load <package>
```

- The programs associated with that package are now available

BLAST command line tools

BLAST command line tools

- With BLAST command lines tools, you can:

BLAST command line tools

- With BLAST command lines tools, you can:
  - Create custom BLAST databases composed of your sequences of interest. A BLAST database is a collection of files that can be created with the `makeblastdb` program.

BLAST command line tools

- With BLAST command lines tools, you can:
  - Create custom BLAST databases composed of your sequences of interest. A BLAST database is a collection of files that can be created with the `makeblastdb` program.
  - Extract FASTA sequences from BLAST databases with the `blastdbcmd` utility.

BLAST command line tools

- With BLAST command lines tools, you can:
  - Create custom BLAST databases composed of your sequences of interest. A BLAST database is a collection of files that can be created with the `makeblastdb` program.
  - Extract FASTA sequences from BLAST databases with the `blastdbcmd` utility.
  - Run `blastn`, `blastp`, `blastx`, etc. searches for large numbers of input sequences

BLAST command line tools

- With BLAST command lines tools, you can:
  - Create custom BLAST databases composed of your sequences of interest. A BLAST database is a collection of files that can be created with the `makeblastdb` program.
  - Extract FASTA sequences from BLAST databases with the `blastdbcmd` utility.
  - Run `blastn`, `blastp`, `blastx`, etc. searches for large numbers of input sequences
  - More info is available at:

BLAST command line tools

- With BLAST command lines tools, you can:
  - Create custom BLAST databases composed of your sequences of interest. A BLAST database is a collection of files that can be created with the `makeblastdb` program.
  - Extract FASTA sequences from BLAST databases with the `blastdbcmd` utility.
  - Run `blastn`, `blastp`, `blastx`, etc. searches for large numbers of input sequences
  - More info is available at:
    - http://www.ncbi.nlm.nih.gov/books/NBK52636/

BLAST command line tools

- With BLAST command lines tools, you can:
  - Create custom BLAST databases composed of your sequences of interest. A BLAST database is a collection of files that can be created with the `makeblastdb` program.
  - Extract FASTA sequences from BLAST databases with the `blastdbcmd` utility.
  - Run `blastn`, `blastp`, `blastx`, etc. searches for large numbers of input sequences
  - More info is available at:
    - http://www.ncbi.nlm.nih.gov/books/NBK52636/
    - http://www.ncbi.nlm.nih.gov/books/NBK1763/

Using Command Line BLAST

Using Command Line BLAST

- 2 Major Steps

Using Command Line BLAST

- 2 Major Steps
  - Generate Database

Using Command Line BLAST

- 2 Major Steps
  - Generate Database
  - Search

Generating a BLAST Database

```
makeblastdb -in <file> -dbtype <type> -out <dbTitle>
```

Generating a BLAST Database

- `makeblastdb` -- will take a file containing the flat sequences and generates a searchable database

```
makeblastdb -in <file> -dbtype <type> -out <dbTitle>
```

Generating a BLAST Database

- `makeblastdb` -- will take a file containing the flat sequences and generates a searchable database

```
makeblastdb -in <file> -dbtype <type> -out <dbTitle>
```

- Here:

Generating a BLAST Database

- `makeblastdb` -- will take a file containing the flat sequences and generates a searchable database

```
makeblastdb -in <file> -dbtype <type> -out <dbTitle>
```

- Here:
  - `<file>` is the fasta file with the database sequences

Generating a BLAST Database

- `makeblastdb` -- will take a file containing the flat sequences and generates a searchable database

```
makeblastdb -in <file> -dbtype <type> -out <dbTitle>
```

- Here:
  - `<file>` is the fasta file with the database sequences
  - `<dbtype>` is one of '`nucl`', or '`prot`'

Generating a BLAST Database

- `makeblastdb` -- will take a file containing the flat sequences and generates a searchable database

```
makeblastdb -in <file> -dbtype <type> -out <dbTitle>
```

- Here:
  - `<file>` is the fasta file with the database sequences
  - `<dbtype>` is one of '`nucl`', or '`prot`'
  - `<dbTitle>` is the file name of the output

BLAST suite of programs: comparison

BLAST suite of programs: comparison

- `blastn`: compares a nucleotide query sequence against a nucleotide sequence database

BLAST suite of programs: comparison

- `blastn`: compares a nucleotide query sequence against a nucleotide sequence database
- `blastp`: compares an amino acid query sequence against a protein sequence database

BLAST suite of programs: comparison

- `blastn`: compares a nucleotide query sequence against a nucleotide sequence database
- `blastp`: compares an amino acid query sequence against a protein sequence database
- `blastx`: compares a nucleotide query sequence translated in all reading frames against a protein sequence database

BLAST suite of programs: comparison

- `blastn`: compares a nucleotide query sequence against a nucleotide sequence database
- `blastp`: compares an amino acid query sequence against a protein sequence database
- `blastx`: compares a nucleotide query sequence translated in all reading frames against a protein sequence database
- `tblastn`: compares a protein query sequence against a nucleotide sequence database dynamically translated in all reading frames

BLAST suite of programs: comparison

- `blastn`: compares a nucleotide query sequence against a nucleotide sequence database
- `blastp`: compares an amino acid query sequence against a protein sequence database
- `blastx`: compares a nucleotide query sequence translated in all reading frames against a protein sequence database
- `tblastn`: compares a protein query sequence against a nucleotide sequence database dynamically translated in all reading frames
- `tblastx`: compares the six-frame translations of a nucleotide query sequence against the six-frame translations of a nucleotide sequence database. Note that `tblastx` program cannot be used with the nr database on the BLAST Web page.

Searching Your New Database

```
[t]blast[xnp] -db <dbTitle> -query <input_file>
```

Searching Your New Database

```
[t]blast[xnp] -db <dbTitle> -query <input_file>
```

- here `[xnp]` is one of those three, and `[t]` is optional should be replaced with the BLAST program you want to use

Searching Your New Database

```
[t]blast[xnp] -db <dbTitle> -query <input_file>
```

- here `[xnp]` is one of those three, and `[t]` is optional should be replaced with the BLAST program you want to use
  - i.e. blastn, blastx, blastp, tblastx, ...

Searching Your New Database

`[t]blast[`*`xnp`*`] -db <`*`dbTitle`*`> -query <`*`input_file`*`>`

- here `[xnp]` is one of those three, and `[t]` is optional should be replaced with the BLAST program you want to use
  - i.e. blastn, blastx, blastp, tblastx, ...
- input file is in FASTA format

HPC Batch Jobs

HPC Batch Jobs

- Jobs run interactively (from the command line) on UA HPC systems are limited to 1Gb of memory and 30 minutes of runtime

HPC Batch Jobs

- Jobs run interactively (from the command line) on UA HPC systems are limited to 1Gb of memory and 30 minutes of runtime
- To run longer jobs or programs that require more than 1Gb of memory, a PBS (Portable Batch Scheduler) job script file is required. These can be `.csh`, `.bash`, or `perl` scripts

HPC Batch Jobs

- Jobs run interactively (from the command line) on UA HPC systems are limited to 1Gb of memory and 30 minutes of runtime
- To run longer jobs or programs that require more than 1Gb of memory, a PBS (Portable Batch Scheduler) job script file is required. These can be `.csh`, `.bash`, or `perl` scripts
- Every PI group gets 30,000 cpu hours per month free of charge on the HPC, after these are used (default queue) you may use the windfall queue

HPC Batch Jobs

- Jobs run interactively (from the command line) on UA HPC systems are limited to 1Gb of memory and 30 minutes of runtime
- To run longer jobs or programs that require more than 1Gb of memory, a PBS (Portable Batch Scheduler) job script file is required. These can be `.csh`, `.bash`, or `perl` scripts
- Every PI group gets 30,000 cpu hours per month free of charge on the HPC, after these are used (default queue) you may use the windfall queue
- The top of the script contains PBS directives

HPC Batch Jobs

- Jobs run interactively (from the command line) on UA HPC systems are limited to 1Gb of memory and 30 minutes of runtime
- To run longer jobs or programs that require more than 1Gb of memory, a PBS (Portable Batch Scheduler) job script file is required. These can be `.csh`, `.bash`, or `perl` scripts
- Every PI group gets 30,000 cpu hours per month free of charge on the HPC, after these are used (default queue) you may use the windfall queue
- The top of the script contains PBS directives
- At the bottom are the commands and programs

HPC Batch Jobs

- Jobs run interactively (from the command line) on UA HPC systems are limited to 1Gb of memory and 30 minutes of runtime
- To run longer jobs or programs that require more than 1Gb of memory, a PBS (Portable Batch Scheduler) job script file is required. These can be `.csh`, `.bash`, or `perl` scripts
- Every PI group gets 30,000 cpu hours per month free of charge on the HPC, after these are used (default queue) you may use the windfall queue
- The top of the script contains PBS directives
- At the bottom are the commands and programs
- Jobs first go into a queue and are run as resources become available

Information needed for PBS Batch Submissions

Information needed for PBS Batch Submissions

- You will need to know your PI group. You can find this with the `va` command.

Information needed for PBS Batch Submissions

- You will need to know your PI group. You can find this with the `va` command.
- You may also need to specify the full paths for input and output files. To determine this, use the `pwd` command

Information needed for PBS Batch Submissions

- You will need to know your PI group. You can find this with the `va` command.
- You may also need to specify the full paths for input and output files. To determine this, use the `pwd` command
- The PBS directives at the beginning of the script tell the system where to send email when your job begins, ends, or aborts, and what resources you are requesting (number of CPUs, time to run, etc.)

Information needed for PBS Batch Submissions

- You will need to know your PI group. You can find this with the `va` command.
- You may also need to specify the full paths for input and output files. To determine this, use the `pwd` command
- The PBS directives at the beginning of the script tell the system where to send email when your job begins, ends, or aborts, and what resources you are requesting (number of CPUs, time to run, etc.)
- If command lines are very long, such as `blastall` with many options, you can continue across several lines by placing a backslash (`\`) at the end of each line

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -W group_list=myGroup
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0

module load blast

cd ~deblasio/blast_dir
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -W group_list=myGroup
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0

module load blast

cd ~deblasio/blast_dir
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@myInstitution.edu
#PBS -W group_list=myGroup
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0

module load blast

cd ~deblasio/blast_dir
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Where my group is your PIs netid

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0

module load blast

cd ~deblasio/blast_dir
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -l
select=1:ncpus=12:mem=24Gb
#PBS -l walltime=24:0:0

module load blast

cd ~deblasio/blast_dir
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -W group_list=myGroup
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0
module load blast

cd ~deblasio/blast_dir
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -W group_list=myGroup
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0
```

```
module load blast
```

```
cd ~deblasio/blast_dir
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -W group_list=myGroup
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0

module load blast

cd ~deblasio/blast_dir
time blastn -db /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Sample PBS Batch Submit File to run BLAST

```
#!/bin/sh
#PBS -N test_blast
#PBS -m bea
#PBS -M myEmail@email.arizona.edu
#PBS -W group_list=myGroup
#PBS -q standard
#PBS -l select=1:ncpus=12:mem=24Gb
#PBS -l cput=96:0:0
#PBS -l walltime=24:0:0

module load blast
```

Notice that the number of processors requested (8) in one more than the num_threads option. Also, the `nt` database requires a lot of memory!

```
time blastn –db  /genome/nt -query seqs.fasta -out seqs.bln \
-evalue 1e-30 -num_descriptions 20 -num_alignments 20 \
-num_threads 11
```

Submitting and Monitoring PBS Jobs

```
qsub <run_blast.sh>
```

```
qstat -a
```

```
qstat -f <JobNumber> | more
```

Submitting and Monitoring PBS Jobs

- To submit a PBS job, use the qsub command with your PBS script as the argument

```
qsub <run_blast.sh>
```

```
qstat -a
```

```
qstat -f <JobNumber> | more
```

Submitting and Monitoring PBS Jobs

- To submit a PBS job, use the qsub command with your PBS script as the argument
  - PBS will return a job number

```
qsub <run_blast.sh>
```

```
qstat -a
```

```
qstat -f <JobNumber> | more
```

Submitting and Monitoring PBS Jobs

- To submit a PBS job, use the qsub command with your PBS script as the argument
  - PBS will return a job number

```
qsub <run_blast.sh>
```

- To view the queue of PBS Jobs, run the command

```
qstat -a
```

```
qstat -f <JobNumber> | more
```

Submitting and Monitoring PBS Jobs

- To submit a PBS job, use the qsub command with your PBS script as the argument
  - PBS will return a job number

```
qsub <run_blast.sh>
```

- To view the queue of PBS Jobs, run the command

```
qstat -a
```

- To see details about a running job, use:

```
qstat -f <JobNumber> | more
```

Files Produced by PBS

Files Produced by PBS

- When your job finishes, you will see 2 files from `PBS`. The files are named according to the *<JobName>* you specify with the `-N` option in the `PBS` submit file and the *<JobID>* Number assigned by `PBS`.

Files Produced by PBS

- When your job finishes, you will see 2 files from `PBS`. The files are named according to the *<JobName>* you specify with the `-N` option in the `PBS` submit file and the *<JobID>* Number assigned by `PBS`.
  - *<JobName>*`.e`*<JobNumber>*

Files Produced by PBS

- When your job finishes, you will see 2 files from `PBS`. The files are named according to the *<JobName>* you specify with the `-N` option in the `PBS` submit file and the *<JobID>* Number assigned by `PBS`.
  - *<JobName>*.e*<JobNumber>*
    - This file contains Error messages. Look here first if your job did not run as expected. If the size of this file is zero, there were no error messages.

Files Produced by PBS

- When your job finishes, you will see 2 files from `PBS`. The files are named according to the *<JobName>* you specify with the `-N` option in the `PBS` submit file and the *<JobID>* Number assigned by `PBS`.
  - *<JobName>*.e*<JobNumber>*
    - This file contains Error messages. Look here first if your job did not run as expected. If the size of this file is zero, there were no error messages.
  - *<JobName>*.o*<JobNumber>*

Files Produced by PBS

- When your job finishes, you will see 2 files from `PBS`. The files are named according to the *<JobName>* you specify with the `-N` option in the `PBS` submit file and the *<JobID>* Number assigned by `PBS`.
  - *<JobName>*.e*<JobNumber>*
    - This file contains Error messages. Look here first if your job did not run as expected. If the size of this file is zero, there were no error messages.
  - *<JobName>*.o*<JobNumber>*
    - This file contains Standard Output messages.

Files Produced by PBS

- When your job finishes, you will see 2 files from `PBS`. The files are named according to the *<JobName>* you specify with the `-N` option in the `PBS` submit file and the *<JobID>* Number assigned by `PBS`.
  - *<JobName>*`.e`*<JobNumber>*
    - This file contains Error messages. Look here first if your job did not run as expected. If the size of this file is zero, there were no error messages.
  - *<JobName>*`.o`*<JobNumber>*
    - This file contains Standard Output messages.
    - The following two lines are at the beginning of the .o file and can be ignored:

```
Warning: no access to tty (Bad file descriptor).
```

Files Produced by PBS

- When your job finishes, you will see 2 files from `PBS`. The files are named according to the *<JobName>* you specify with the `-N` option in the `PBS` submit file and the *<JobID>* Number assigned by `PBS`.
  - *<JobName>*`.e`*<JobNumber>*
    - This file contains Error messages. Look here first if your job did not run as expected. If the size of this file is zero, there were no error messages.
  - *<JobName>*`.o`*<JobNumber>*
    - This file contains Standard Output messages.
    - The following two lines are at the beginning of the .o file and can be ignored:

```
Warning: no access to tty (Bad file descriptor).
```

- Look at the .o file to see how much CPU time your PI's group was charged for the run and how much time remains in your group's monthly allocation.