Task 2: Annotation using blast

- Use blast to identify viral and bacterial RNAs

- Basic blast tools
  - blastn (nucleotide against nucleotide), close relationship
  - blastx (nucleotide against protein), distant relationship
  - tblastn (protein against translated nucleotide)
  - tblastx (translated nucleotide against translated nucleotide)
  - rpsblast (blast against protein structural database)

- Blast can be performed at local computer
  - Download/make the blast database
  - Use your fasta file as query and perform blast

ftp://ftp.ncbi.nlm.nih.gov/blast/db/

Some useful databases from the NCBI ftp

Non-redundant nucleotide (nt)

Non-redundant protein (nr)

Bacterial and Archaeal 16S rRNA

(16SMicrobial)

Conserved domain database (cdd_delta)

Transcriptome Shotgun Assembly (tsa)

Whole Genome Shortgun Assembly (wgs)

Taxonomy Information (taxdb)

And many others….

Task 2.1: Blastn against the nt database

- First we will blast against the NCBI nucleotide (nt) database using blastn
  - ✓ Aligns contig nt sequences with all nt sequences in NCBI database

- In order to have the taxonomy information in the blast results, we first obtain taxonomy information database (taxdb) – need in working dir

```
wget ftp://ftp.ncbi.nlm.nih.gov/blast/db/taxdb.tar.gz
```

- Once the downloading is completed, unzip the file:

```
tar -zxvf taxdb.tar.gz
```

Task 2.1: Blastn against the nt database

- Blast parameters:

```
blastn -query "$Name".contigs.fa -db $dbnt/nt -out "$Name".contigs.fa.nt -evalue 1E-10 -
max_target_seqs 1 -num_threads 2 -outfmt "6 qseqid sacc salltitles pident length evalue
sskingdoms"
```

-query query file name

-db BLAST database name (here it is nt database located in a shared databases folder)

-out output file name (you check this file for results)

-evalue e-value cutoff (minimum acceptable e-value for saving hits)

-max_target_seqs Number of hits to keep for each query – *Will discuss further*

-num_threads number of threads used for blast analyses

-outfmt the output format, here we use the tabular format "6" with custom specifiers

- The job should take a few mins (normally 1 day)
- Output file microplus.contigs.fa.nt should be created

Variables
Name=microplus
dbnt= /course/data/databases

Format of Blast Output

- Default output format

```
Query= TR654_c0_g2_i1_len=1355

Length=1355

                                                                    Score     E
Sequences producing significant alignments:                        (Bits)  Value

  gi|635546944|ref|YP_009029996.1| putative capsid protein [Jingm...    455   7e-160
  gi|635546945|ref|YP_009029997.1| putative membrane protein [Jin...    333   3e-108


> gi|635546944|ref|YP_009029996.1| putative capsid protein [Jingmen
tick virus]
Length=254

 Score =   455 bits (1171),  Expect = 7e-160, Method: Compositional matrix adjust.
 Identities = 254/254 (100%), Positives = 254/254 (100%), Gaps = 0/254 (0%)
 Frame = +3

Query  135  MPNAKtalllvtlvglalgGNDKPDWEKVKLLGQQLQSGIATKNAVKVKYYAELIISEMP  314
            MPNAKTALLLVTLVGLALGGNDKPDWEKVKLLGQQLQSGIATKNAVKVKYYAELIISEMP
Sbjct  1    MPNAKTALLLVTLVGLALGGNDKPDWEKVKLLGQQLQSGIATKNAVKVKYYAELIISEMP  60
```

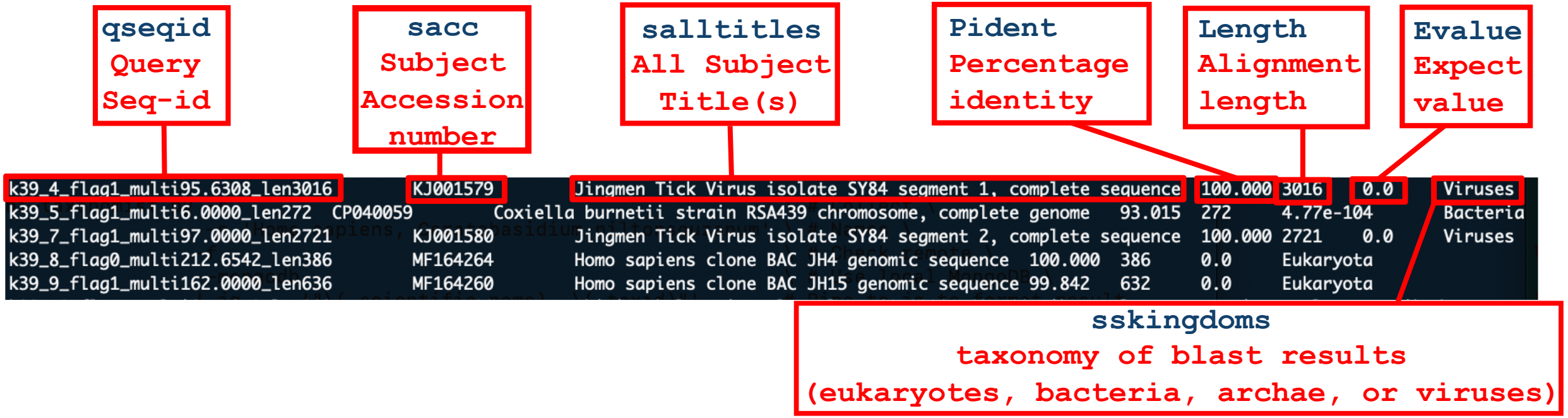
Format of Blast Output

- To simplify, we can change the format of blast output to tabular by adding:

-outfmt "6"

- To adjust the specifiers, and design your own output tables:

-outfmt "6 qseqid sacc salltitles pident length evalue sskingdoms"



Blast: max target sequences

- Blast parameters:

```
blastn -query "$Name".contigs.fa -db $dbnt/nt -out "$Name".contigs.fa.nt -evalue 1E-10 -
max_target_seqs 1 -num_threads 2 -outfmt "6 qseqid sacc salltitles pident length evalue
sskingdoms"
```

- For this workshop, to save time, we set the maximum no. of target sequences as 1
- Easiest to view just one result per contig HOWEVER if you use –max_target_seqs, blast does not necessarily return the top hit
- Default is 500 – should be pretty safe that the top hit is among 500
- Therefore, on your own data, leave out the max_target_seqs altogether

Variables
Name=microplus
dbnt= /course/data/databases

Task 2.1: Blastn against the nt database

- Because in a normal pipeline, you would get more than one blast hit, you would need to extract the top hit
- To keep only the top hit

```
awk -F$'\t' '!seen[$1]++' "$Name".contigs.fa.nt >
"$Name".contigs.fa.nt.topHit
```

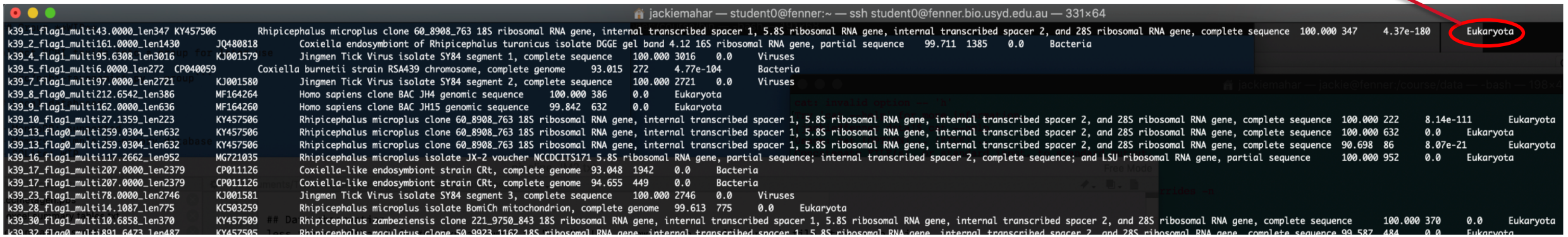
Variables
Name=microplus

Task 2.1: Blastn against the nt database

- Check your results

```
less "$Name".contigs.fa.nt.topHit
```

Taxonomy: very useful




```
k39_1_flag1_multi43.0000_len347 KY457506       Rhipicephalus microplus clone 60_8908_763 18S ribosomal RNA gene, internal transcribed spacer 1, 5.8S ribosomal RNA gene, internal transcribed spacer 2, and 28S ribosomal RNA gene, complete sequence  100.000 347     4.37e-180       Eukaryota
k39_2_flag1_multi161.0000_len1430      JQ480818        Coxiella endosymbiont of Rhipicephalus turanicus isolate DGGE gel band 4.12 16S ribosomal RNA gene, partial sequence    99.711  1385    0.0     Bacteria
k39_4_flag1_multi95.6308_len3016       KJ001579        Jingmen Tick Virus isolate SY84 segment 1, complete sequence    100.000 3016    0.0     Viruses
k39_5_flag1_multi6.0000_len272  CP040059        Coxiella burnetii strain RSA439 chromosome, complete genome     93.015  272     4.77e-104       Bacteria
k39_7_flag1_multi97.0000_len2721       KJ001580        Jingmen Tick Virus isolate SY84 segment 2, complete sequence    100.000 2721    0.0     Viruses
k39_8_flag0_multi212.6542_len386       MF164264        Homo sapiens clone BAC JH4 genomic sequence     100.000 386     0.0     Eukaryota
k39_9_flag1_multi162.0000_len636       MF164260        Homo sapiens clone BAC JH15 genomic sequence    99.842  632     0.0     Eukaryota
k39_10_flag1_multi27.1359_len223       KY457506        Rhipicephalus microplus clone 60_8908_763 18S ribosomal RNA gene, internal transcribed spacer 1, 5.8S ribosomal RNA gene, internal transcribed spacer 2, and 28S ribosomal RNA gene, complete sequence  100.000 222     8.14e-111       Eukaryota
k39_13_flag0_multi259.0304_len632      KY457506        Rhipicephalus microplus clone 60_8908_763 18S ribosomal RNA gene, internal transcribed spacer 1, 5.8S ribosomal RNA gene, internal transcribed spacer 2, and 28S ribosomal RNA gene, complete sequence  100.000 632     0.0     Eukaryota
k39_13_flag0_multi259.0304_len632      KY457506        Rhipicephalus microplus clone 60_8908_763 18S ribosomal RNA gene, internal transcribed spacer 1, 5.8S ribosomal RNA gene, internal transcribed spacer 2, and 28S ribosomal RNA gene, complete sequence  90.698  86      8.07e-21        Eukaryota
k39_16_flag1_multi117.2662_len952      MG721035        Rhipicephalus microplus isolate JX-2 voucher NCCDCITS171 5.8S ribosomal RNA gene, partial sequence; internal transcribed spacer 2, complete sequence; and LSU ribosomal RNA gene, partial sequence     100.000 952     0.0     Eukaryota
k39_17_flag1_multi207.0000_len2379      CP011126        Coxiella-like endosymbiont strain CRt, complete genome  93.048  1942    0.0     Bacteria
k39_17_flag1_multi207.0000_len2379      CP011126        Coxiella-like endosymbiont strain CRt, complete genome  94.655  449     0.0     Bacteria
k39_23_flag1_multi78.0000_len2746       KJ001581        Jingmen Tick Virus isolate SY84 segment 3, complete sequence    100.000 2746    0.0     Viruses
k39_28_flag1_multi14.1087_len775       KC503259        Rhipicephalus microplus isolate BomiCh mitochondrion, complete genome   99.613  775     0.0     Eukaryota
k39_30_flag1_multi10.6858_len370       KY457509        Rhipicephalus zambeziensis clone 221_9750_843 18S ribosomal RNA gene, internal transcribed spacer 1, 5.8S ribosomal RNA gene, internal transcribed spacer 2, and 28S ribosomal RNA gene, complete sequence      100.000 370     0.0     Eukaryota
k39_32_flag0_multi891.6473_len487      KY457505        Rhipicephalus maculatus clone 50_9923_1162 18S ribosomal RNA gene, internal transcribed spacer 1, 5.8S ribosomal RNA gene, internal transcribed spacer 2, and 28S ribosomal RNA gene, complete sequence 99.587  484     0.0     Eukaryota
```

Variables
Name=microplus

Task 2.1: Blastn against the nt database

- Extract blast hit to viruses

```
grep "Viruses" "$Name".contigs.fa.nt.topHit
```

Viruses Only

```
[student0@fenner ~]$ grep "Viruses" "$Name".contigs.fa.nt.topHit
k39_3_flag1_multi78.0000_len2746      KJ001581      Jingmen Tick Virus isolate SY84 segment 3, complete sequence   100.000 2746   0.0   Viruses
k39_4_flag1_multi95.6308_len3016      KJ001579      Jingmen Tick Virus isolate SY84 segment 1, complete sequence   100.000 3016   0.0   Viruses
k39_7_flag1_multi97.0000_len2721      KJ001580      Jingmen Tick Virus isolate SY84 segment 2, complete sequence   100.000 2721   0.0   Viruses
k39_92_flag1_multi48.0000_len2733     KJ001582      Jingmen Tick Virus isolate SY84 segment 4, complete sequence   100.000 2733   0.0   Viruses
```

A utility that searches plain txt datasets for lines that match regular expression, using syntax:

grep "search term" file_to_search

- Extract blast hit to bacteria

```
grep "Bacteria" "$Name".contigs.fa.nt.topHit
```

```
[student0@fenner ~]$ grep "Bacteria" "$Name".contigs.fa.nt.topHit
k39_102_flag1_multi7.0000_len330      CP015012      Rickettsia amblyommatis isolate An13, complete genome   100.000 330    1.16e-170      Bacteria
k39_17_flag1_multi207.0000_len2379    CP011126      Coxiella-like endosymbiont strain CRt, complete genome  93.048  1942   0.0      Bacteria
k39_2_flag1_multi161.0000_len1430     JQ480818      Coxiella endosymbiont of Rhipicephalus turanicus isolate DGGE gel band 4.12 16S ribosomal RNA gene, partial sequence   99.711  1385   0.0      Bacteria
k39_5_flag1_multi6.0000_len272  CP040059      Coxiella burnetii strain RSA439 chromosome, complete genome     93.015  272    4.77e-104      Bacteria
k39_77_flag1_multi7.0000_len264 CP032049      Rickettsia japonica strain LA4/2015 chromosome, complete genome 99.621  264    2.06e-132      Bacteria
```

Variables

Name=microplus

Task 2.2: Blastx against the protein database

- To detect more diverse sequences → blast against the NBCI non-redundant (nr) protein database

- Use blastx to blast nt sequences against a protein database
  - Blastx translates nt seqs in all 6 frames and aligns with protein database
  - Slower than nt blast
  - Using DIAMOND blast software as this is quicker

Task 2.2: Blastx against the protein database

- Blast parameters:

```
diamond blastx -q "$Name".contigs.fa -d $dbnr/nr -o "$Name".contigs.fa.nr -e 1E-10 -k 1 -p
2 -f 6 qseqid qlen sseqid stitle pident length evalue
```

-q query file name

-d BLAST database name (here it is nr database located in a shared databases folder)

-o output file name (you check this file for results)

-e e-value cutoff (minimum acceptable e-value for saving hits)

-k Number of hits to keep for each query - same applies as for blastn -> don't use -k normally

-p number of processors/threads used for blast analyses

-f the output format, here we use the tabular format "6" with custom specifiers

- The job should take <2 mins (normally 1-2 days)
- Output file microplus.contigs.fa.nr should be created

Variables
Name=microplus
dbnr=/course/data/databases/diamond

Why use diamond blast?

- "Diamond": a much needed replacement for **BLASTX,** 4 orders of magnitude faster

- Three modifications:

  (1) Uses an optimized subset (seed) of the query and reference sequences

  (2) Improved method for storing information

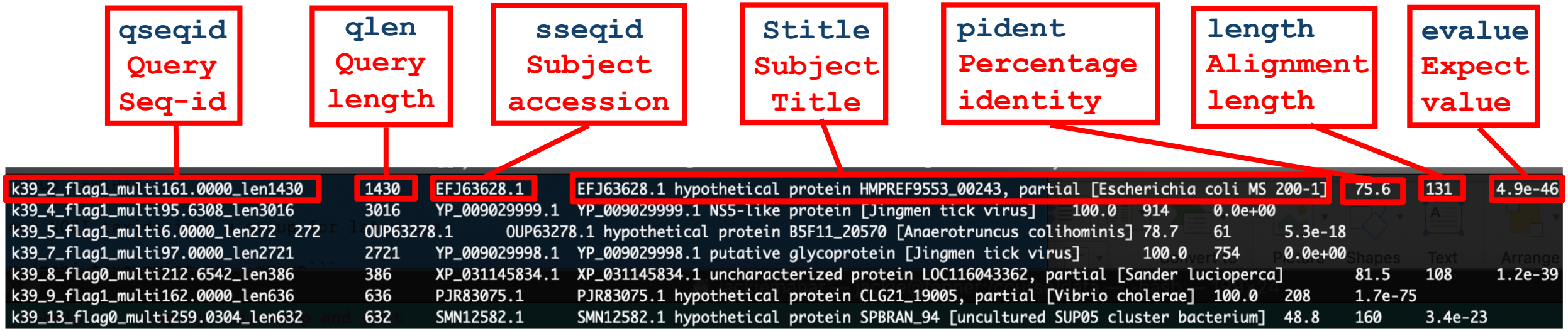
  (3) Reduced amino acid alphabet (only 11 amino acids)

- Not so much improvement when the database or query size is small.

Task 2.2: Blastx against the protein database

- Check your results

```
less "$Name".contigs.fa.nr
```



Variables
Name=microplus

Task 2.2: Blastx against the protein database

- Find virus hits

```
grep "\[.*virus" "$Name".contigs.fa.nr
```

```
[student0@fenner ~]$ grep -i "\[.*virus" "$Name".contigs.fa.nr.tophit
k39_23_flag1_multi78.0000_len2746   2746   YP_009030000.1  YP_009030000.1 NS3-like protein [Jingmen tick virus]   100.0   808   0.0e+00
k39_4_flag1_multi95.6308_len3016    3016   YP_009029999.1  YP_009029999.1 NS5-like protein [Jingmen tick virus]   100.0   914   0.0e+00
k39_7_flag1_multi97.0000_len2721    2721   YP_009029998.1  YP_009029998.1 putative glycoprotein [Jingmen tick virus]   100.0   754   0.0e+00
k39_92_flag1_multi48.0000_len2733   2733   YP_009029997.1  YP_009029997.1 putative membrane protein [Jingmen tick virus]   100.0   538   1.3e-305
```

- Regex search term: `\[.*virus`

- Searches for lines with the word “virus” appearing AFTER an open square bracket
- In nr database, organism names are enclosed in square brackets

Variables
Name=microplus

Task 2.3: Taxonomy lineage annotation

Original blast table

Identifier	Length	Accession	Blast hit	Identity	Align length	Evalue
k39_3	2721	YP_009029998.1	YP_009029998.1 putative glycoprotein [Jingmen tick virus]	100	754	0.00E+00

Adding taxonomy information to blast table

Identifier	Length	Accession	Blast hit	Identity	Align length	Evalue	Taxonomy lineage
k39_3	2721	YP_009029998.1	YP_009029998.1 putative glycoprotein [Jingmen tick virus]	100	754	0.00E+00	Viruses|NA|NA|Flaviviridae|NA|NA|Mogiana tick virus

Task 2.3: Taxonomy lineage annotation

Blast Table

Identifier	Length	Accession	Blast hit	Identity	Align length	Evalue
k39_3	2721	YP_009029998.1	YP_009029998.1 putative glycoprotein [Jingmen tick virus]	100	754	0.00E+00

Taxonomy ID Table

Aceesion	Accession	Taxonomy ID	GI number
ABI99685	ABI99685.1	405955	115511611
ADX22475	ADX22475.1	32630	323090655
APY21410	APY21410.1	6941	1135519473
CAJ30045	CAJ30045.1	431944	78033430

Taxonomy Lineage Table

Taxonomy ID	Taxonomy Lineage
1311	Bacteria|Firmicutes|Bacilli|Lactobacillales|Streptococcaceae|Streptococcus|Streptococcus agalactiae
1491393	Viruses|NA|NA|Flaviviridae|NA|NA|Mogiana tick virus

Task 2.3: Taxonomy lineage annotation

- Get a list of accession numbers from blastx top hits

```
cat "$Name".contigs.fa.nr.tophit | cut -f3 | sort -u | grep -v "^[0-9]" |
grep -v -e '^$' > "$Name".accession_list.txt
```

Variables
Name=microplus

- Get a **taxonomy lineage** for each accession number

```
grep -F -f "$Name".accession_list.txt $access2taxid/prot.accession2taxid |
tee "$Name".taxid_table.txt  | cut -f3 -d$'\t' | sort -u | tee
"$Name".taxid_list.txt | python3 $taxonomist/ncbi.taxonomist.py --sep "|" -
d | sed "s/|/\t/" > "$Name".lineage_table.txt
```

Get these two scripts running and then we can go through them

Variables
Name=microplus
access2taxid=/course/data/databases
taxonomist=/course/bin/simbiont/ncbi/tools

Task 2.3: Taxonomy lineage annotation

- Get **Accession List** from blastx table

Identifier	Length	Accession	Blast hit	Identity	Align length	Evalue
k39_3	2721	YP_009029998.1	YP_009029998.1 putative glycoprotein [Jingmen tick virus]	100	754	0.00E+00

The cut utility lets you cut out specific columns. –f3 extracts the 3rd column (contains contig names)

```
cat "$Name".contigs.fa.nr.tophit | cut -f3 | sort -u
| grep -v "^[0-9]" | grep -v -e '^$' >
"$Name".accession_list.txt
```

Variables
Name=microplus

Task 2.3: Taxonomy lineage annotation

- From Accession List, use prot.accession2taxid database to make a **Taxonomy ID Table**

```
grep -F -f "$Name".accession_list.txt
$access2taxid/prot.accession2taxid | tee
"$Name".taxid_table.txt  | cut -f3 -d$'\t' | sort -u |
tee "$Name".taxid_list.txt | python3
$taxonomist/ncbi.taxonomist.py --sep "|" -d | sed
"s/|/\t/" > "$Name".lineage_table.txt
```

Aceesion	Accession	Taxonomy ID	GI number
ABI99685	ABI99685.1	405955	115511611
ADX22475	ADX22475.1	32630	323090655
APY21410	APY21410.1	6941	1135519473
CAJ30045	CAJ30045.1	431944	78033430

Variables
Name=microplus
access2taxid=/course/data/databases
taxonomist=/course/bin/simbiont/ncbi/tools

Task 2.3: Taxonomy lineage annotation

- From Taxonomy ID table, cut **Taxonomy ID list**

```
grep -F -f "$Name".accession_list.txt
$access2taxid/prot.accession2taxid | tee
"$Name".taxid_table.txt  | cut -f3 -d$'\t' | sort -u |
tee "$Name".taxid_list.txt | python3
$taxonomist/ncbi.taxonomist.py --sep "|" -d | sed
"s/|/\t/" > "$Name".lineage_table.txt
```

Aceesion	Accession	Taxonomy ID	GI number
ABI99685	ABI99685.1	405955	115511611
ADX22475	ADX22475.1	32630	323090655
APY21410	APY21410.1	6941	1135519473
CAJ30045	CAJ30045.1	431944	78033430

Variables
Name=microplus
access2taxid=/course/data/databases
taxonomist=/course/bin/simbiont/ncbi/tools

Task 2.3: Taxonomy lineage annotation

- Based on taxonomy ID list, generate **Lineage Table**

```
grep -F -f "$Name".accession_list.txt
$access2taxid/prot.accession2taxid | tee
"$Name".taxid_table.txt  | cut -f3 -d$'\t' | sort -u | tee
"$Name".taxid_list.txt | python3
$taxonomist/ncbi.taxonomist.py --sep "|" -d | sed "s/|/\t/" >
"$Name".lineage_table.txt
```

Variables
Name=microplus
access2taxid=/course/data/databases
taxonomist=/course/bin/simbiont/ncbi/tools

Taxonomy ID	Taxonomy Lineage
1311	Bacteria|Firmicutes|Bacilli|Lactobacillales|Streptococcaceae|Streptococcus|Streptococcus agalactiae
1491393	Viruses|NA|NA|Flaviviridae|NA|NA|Mogiana tick virus

Task 2.3: Taxonomy lineage annotation

- Checking the taxonomy lineage table

```
head "$Name".lineage_table.txt
```

```
[student0@fenner ~]$ head "$Name".lineage_table.txt
1172189 Eukaryota|NA|Ciliophora|Intramacronucleata|Spirotrichea|NA|Sporadotrichida|NA|Oxytrichidae|Oxytrichinae|Oxytricha|NA|Oxytricha trifallax
1310558 Bacteria|Proteobacteria|Gammaproteobacteria|Pseudomonadales|Moraxellaceae|Acinetobacter|Acinetobacter baumannii
1491393 Viruses|NA|NA|Flaviviridae|NA|NA|Mogiana tick virus
151549  Eukaryota|Metazoa|Arthropoda|Hexapoda|Insecta|Amphiesmenoptera|Lepidoptera|Glossata|Psychidae|Oiketicinae|Eumeta|NA|Eumeta japonica
1630406 Bacteria|Bacteroidetes|Flavobacteriia|Flavobacteriales|Flavobacteriaceae|Tamlana|Tamlana sp. s12
169435  Bacteria|Firmicutes|Clostridia|Clostridiales|Ruminococcaceae|Anaerotruncus|Anaerotruncus colihominis
181606  Eukaryota|Metazoa|Nematoda|NA|Enoplea|NA|Trichinellida|NA|Trichinellidae|NA|Trichinella|NA|Trichinella sp. T9
2200888 Archaea|Crenarchaeota|Thermoprotei|Sulfolobales|Sulfolobaceae|Sulfolobus|Sulfolobus sp. B1
2528593 Bacteria|Firmicutes|Clostridia|NA|NA|NA|Clostridia bacterium k32
283035  Eukaryota|Metazoa|Chordata|Craniata|Actinopteri|NA|Perciformes|Percoidei|Percidae|Luciopercinae|Sander|NA|Sander lucioperca
```

Variables
Name=microplus

Task 2.3: Taxonomy lineage annotation

Blast Table

Identifier	Length	Accession	Blast hit	Identity	Align length	Evalue
k39_3	2721	YP_009029998.1	YP_009029998.1 putative glycoprotein [Jingmen tick virus]	100	754	0.00E+00

Taxonomy ID Table

Aceesion	Accession	Taxonomy ID	GI number
ABI99685	ABI99685.1	405955	115511611
ADX22475	ADX22475.1	32630	323090655
APY21410	APY21410.1	6941	1135519473
CAJ30045	CAJ30045.1	431944	78033430

Taxonomy Lineage Table

Taxonomy ID	Taxonomy Lineage
1311	Bacteria|Firmicutes|Bacilli|Lactobacillales|Streptococcaceae|Streptococcus|Streptococcus agalactiae
1491393	Viruses|NA|NA|Flaviviridae|NA|NA|Mogiana tick virus

Task 2.3: Taxonomy lineage annotation

Taxonomy Lineage Table

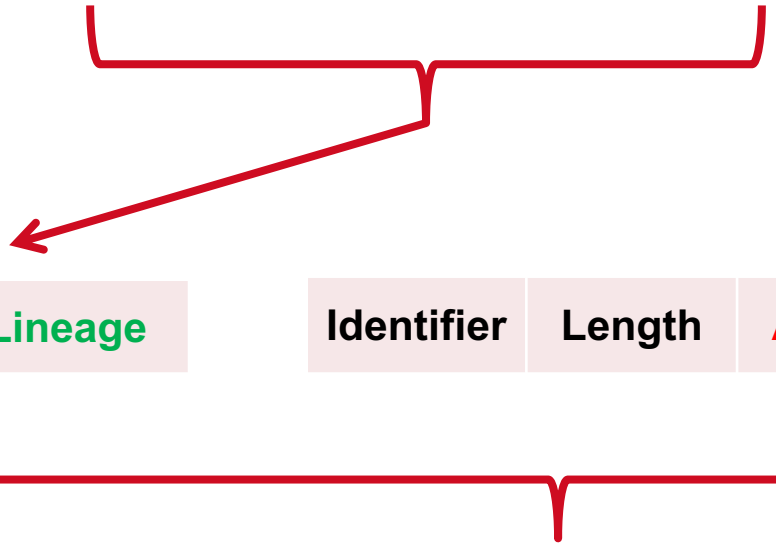
Taxonomy ID	Taxonomy Lineage

Taxonomy ID Table

Aceesion	Accession	Taxonomy ID	GI number

Blast Table

Identifier	Length	Accession	Blast hit	Identity	Align length	Evalue



Accession	Taxonomy Lineage

Identifier	Length	Accession	Blast hit	Identity	Align length	Evalue	Taxonomy Lineage

Task 2.3: Taxonomy lineage annotation

- Add taxonomy lineage to blastx results using SQLite (Join the three tables)
- SQLite is an SQL database engine – essentially can use it to make databases & join tables

#Note - can't use variables in sqlite3, therefore use complete file name within sqlite3

```
sqlite3 "$Name".sql

### Create hypothetical tables

CREATE TABLE lineage (taxonomyid INT, taxlineage CHAR);

CREATE TABLE taxid (accession CHAR, accession2 CHAR, taxonomyid INT, GI INT);

CREATE TABLE blast (reads CHAR, length INT, accession2 CHAR, blasthit CHAR, pident_nt FLOAT, hitlength INT,
evalue FLOAT);

### Import the tables

.mode tabs

.import ./microplus.lineage_table.txt lineage

.import ./microplus.taxid_table.txt taxid

.import ./microplus.contigs.fa.nr.tophit blast
```

Variables
Name=microplus

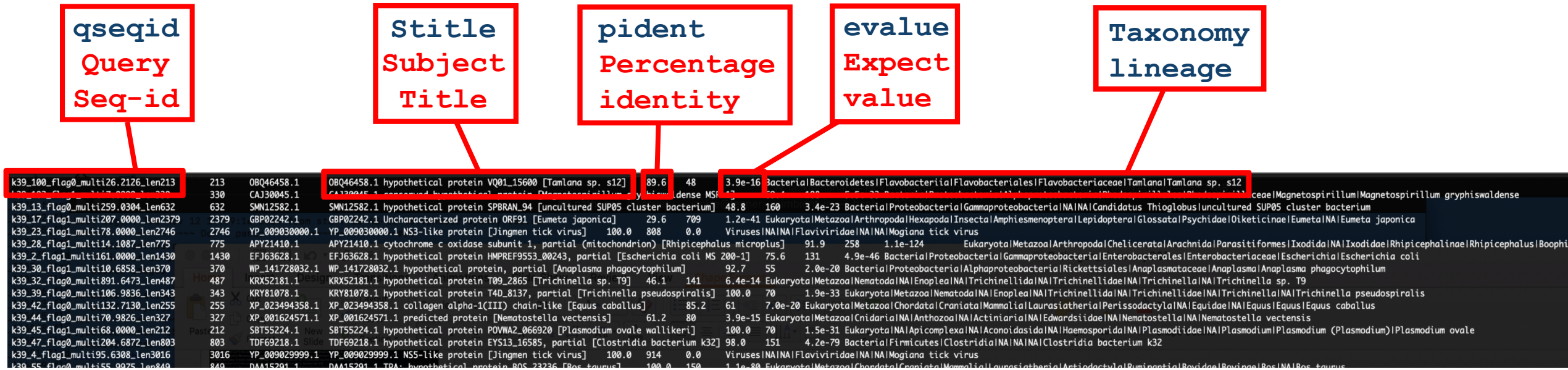
Task 2.3: Taxonomy lineage annotation

```
### Merge taxid table and lineage table, take only the accession and taxonomy lineage column

CREATE TABLE taxid_lineage (accession2 CHAR, taxlineage CHAR);

INSERT INTO taxid_lineage SELECT taxid.accession2, lineage.taxlineage FROM taxid LEFT JOIN lineage ON
taxid.taxonomyid=lineage.taxonomyid;

### Merge taxid_lineage and blast table: take the whole blast table plus the taxonomy lineage column

CREATE TABLE blast_taxid_lineage (reads CHAR, length INT, accession2 CHAR, blasthit CHAR, pident_nt FLOAT, hitlength INT,
evalue FLOAT, taxlineage CHAR);

INSERT INTO blast_taxid_lineage SELECT blast.*, taxid_lineage.taxlineage FROM blast LEFT JOIN taxid_lineage ON
blast.accession2=taxid_lineage.accession2;

### Write out the table

.output microplus.contigs.fa.nr.tophit.edited

SELECT rowid, blast_taxid_lineage.* FROM blast_taxid_lineage;

.output stdout

.exit
```

Task 2.3: Taxonomy lineage annotation

- View table with taxonomy info added to blastx results

```
less "$Name".contigs.fa.nr.tophit.edited
```

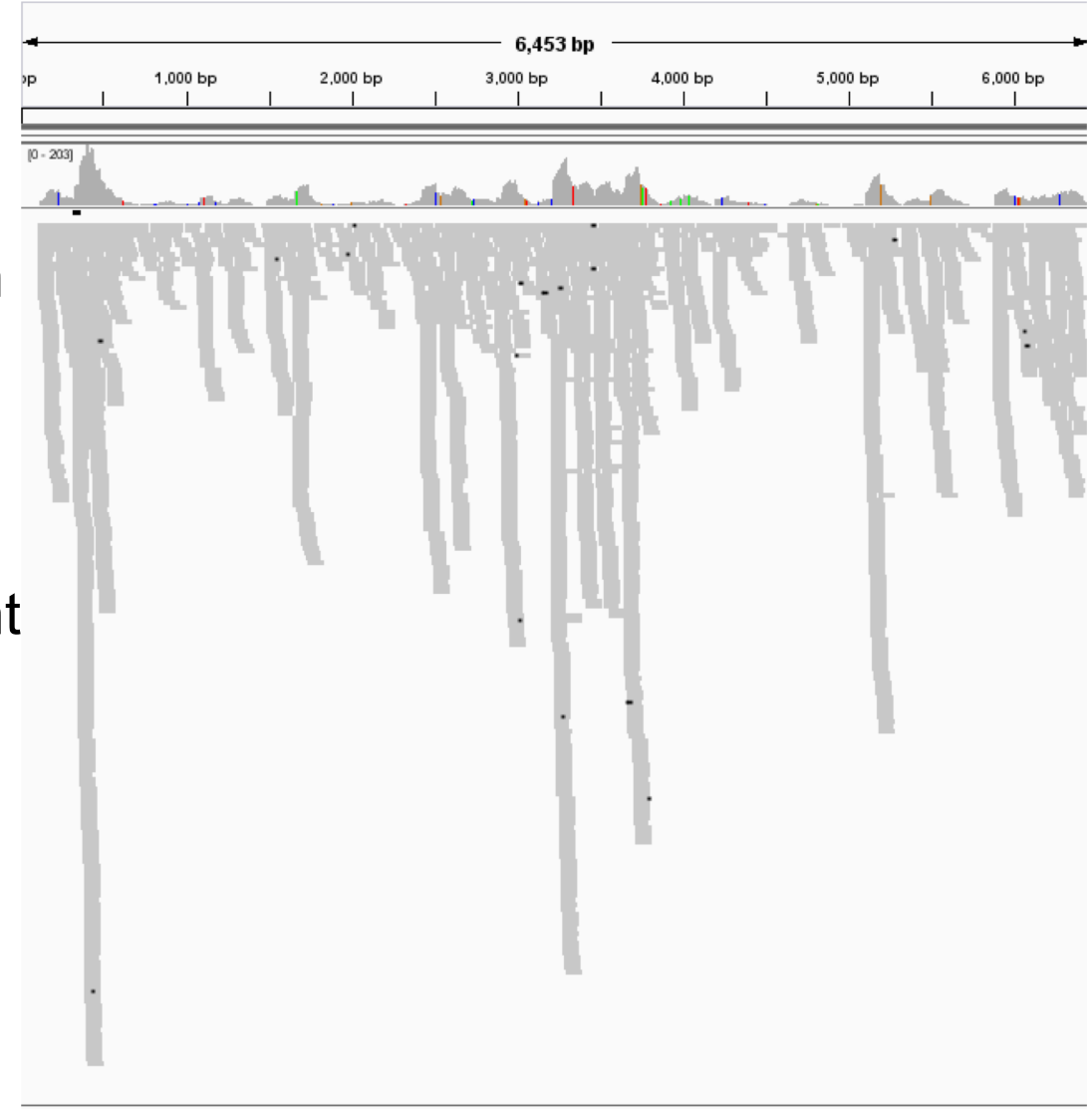


Variables
Name=microplus

Task 3: Estimate abundance by mapping

- Using Bowtie2 to map the reads back to the assembled contigs
  - Estimate the abundance (if there is no bias in sample processing)

- Also useful for (not covered today):
  - Determining major and minor mutation variant
  - Correcting errors from assembly
  - Extending the contigs at both ends



Task 3: Estimate abundance by mapping

- In interest of time, will demonstrate read mapping only for the ”virus” contigs

  - Get list of virus contigs with keyword "Virus" in taxonomy lineage

```
grep "Viruses" "$Name".contigs.fa.nr.tophit.edited | cut -f2 | sort -u
> "$Name"_VirusContigs_list.txt
```

  - Get fasta sequences of contigs with "Virus" in taxonomy lineage

```
seqtk subseq "$Name".contigs.fa "$Name"_VirusContigs_list.txt >
"$Name".contigs.fa.virusHit.fa
```

The seqtk program with the subseq option allows you to extract specific fasta sequences using a list of sequence names

Variables
Name=microplus

Task 3: Estimate abundance by mapping

- The simplest approach to estimate abundance
  - Step 1. Bowtie, map the reads to the assembled contigs
  - Step 2. Samtools, determine the amount of reads mapped to each contigs

Task 3: Estimate abundance by mapping

- Bowtie and samtools commands:

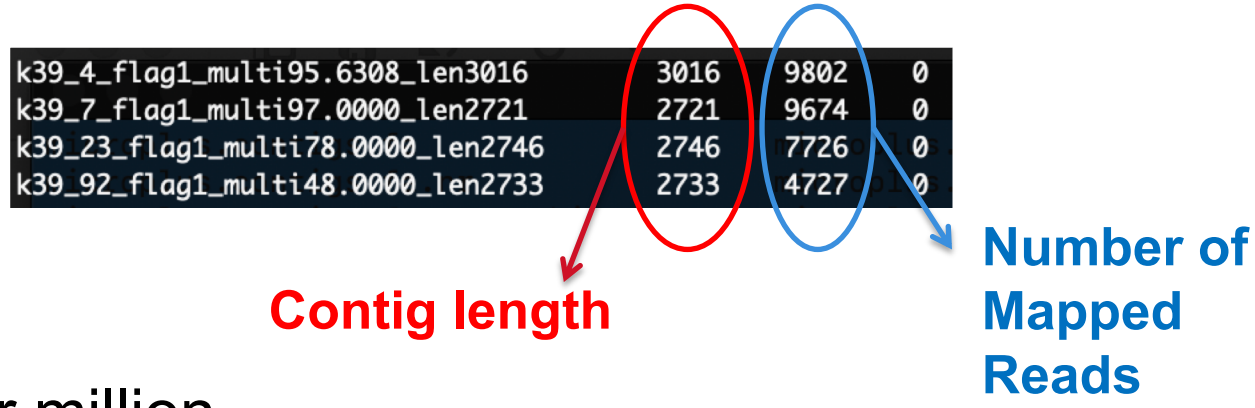
```
# Indexing
bowtie2-build "$Name".contigs.fa.virusHit.fa reference

# Mapping
bowtie2 --local --threads 2 -x reference -U $inpath/"$Name".fq -S "$Name".sam

# Convert sam to bam
samtools view -bSF4 "$Name".sam > "$Name".bam

# Sorting
samtools sort -@ 20 "$Name".bam > "$Name".sorted.bam

# Indexing
samtools index "$Name".sorted.bam

# Read count for each contigs
samtools idxstats "$Name".sorted.bam > "$Name"_mapping.txt
```

Variables
Name=microplus
inpath=/course/data

Task 3: Estimate abundance by mapping

- Check the results

```
less "$Name"_mapping.txt
```



- Calculate abundance: RPM, reads per million

formula:

RPM = (mapped reads/total reads) x 1,000,000

Variables
Name=microplus

Software required to run pipeline

- For set-up on personal machines after this workshop, software required are
  - megahit: https://github.com/voutcn/megahit
  - blast+: https://blast.ncbi.nlm.nih.gov/Blast.cgi?PAGE_TYPE=BlastDocs&DOC_TYPE=Download
  - diamond: https://github.com/bbuchfink/diamond
  - python3: https://www.python.org/download/releases/3.0/
  - sqlite3: https://sqlite.org/index.html
  - seqtk: https://github.com/lh3/seqtk
  - ncbi-taxonomist: https://gitlab.com/janpb/simbiont →See next slide for instructions
  - bowtie2: http://bowtie-bio.sourceforge.net/bowtie2/index.shtml
  - samtools: http://samtools.sourceforge.net
  - To view mapping: Integrative Genomics Viewer (IGV), Geneious, or similar

Set-up ncbi-taxonomy

- ncbi.taxonomy.py is a work in progress by Jan Buchmann
- To download the version used in this workshop that will work with the commands used here, follow the below instructions:

```
### Get ncbi.taxonomist

git clone --recurse-submodules -b course   https://gitlab.com/janpb/simbiont.git


### Define variable for path to ncbi.taxonomist.py

#$SIMBIONT indicates the path where you cloned the repository above eg if you cloned the repository into /course/bin, then SIMBIONT=/course/bin

taxonomist=$SIMBIONT/simbiont/ncbi/tools
```

Databases required to run the pipeline

- For set-up on personal machines after workshop, databases required are:
  - nt (blast format)

    wget ftp://ftp.ncbi.nlm.nih.gov/blast/db/nt*gz
  - nr (fasta format → needs to be formatted for diamond)

    wget ftp://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/nr.gz
  - prot.accession2taxid.gz

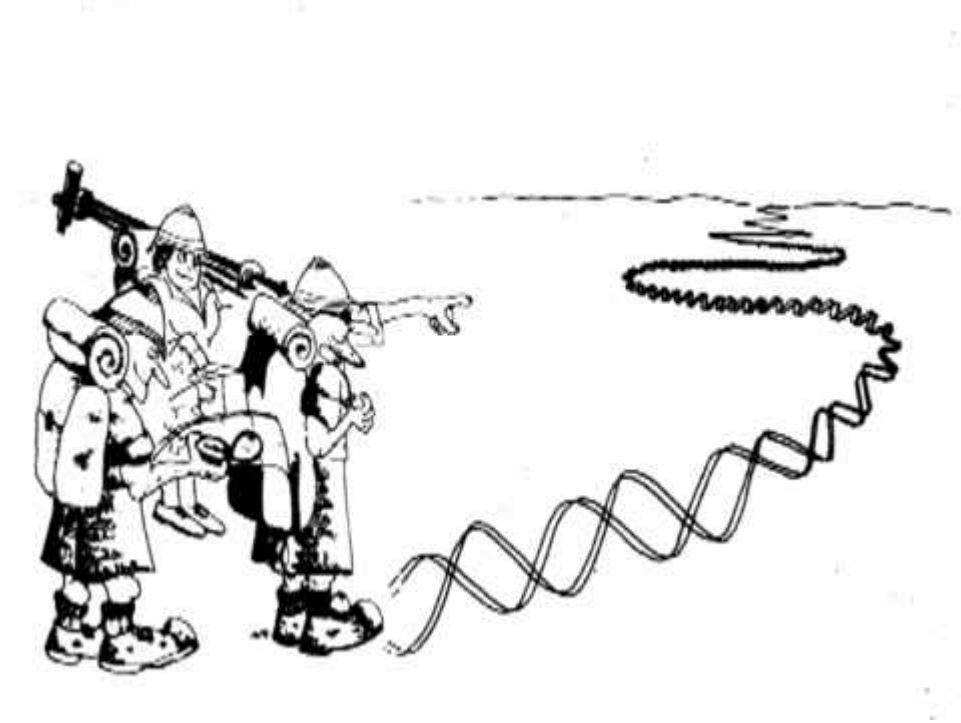
    wget ftp://ftp.ncbi.nih.gov/pub/taxonomy/accession2taxid/prot.accession2taxid.gz

Thank You!

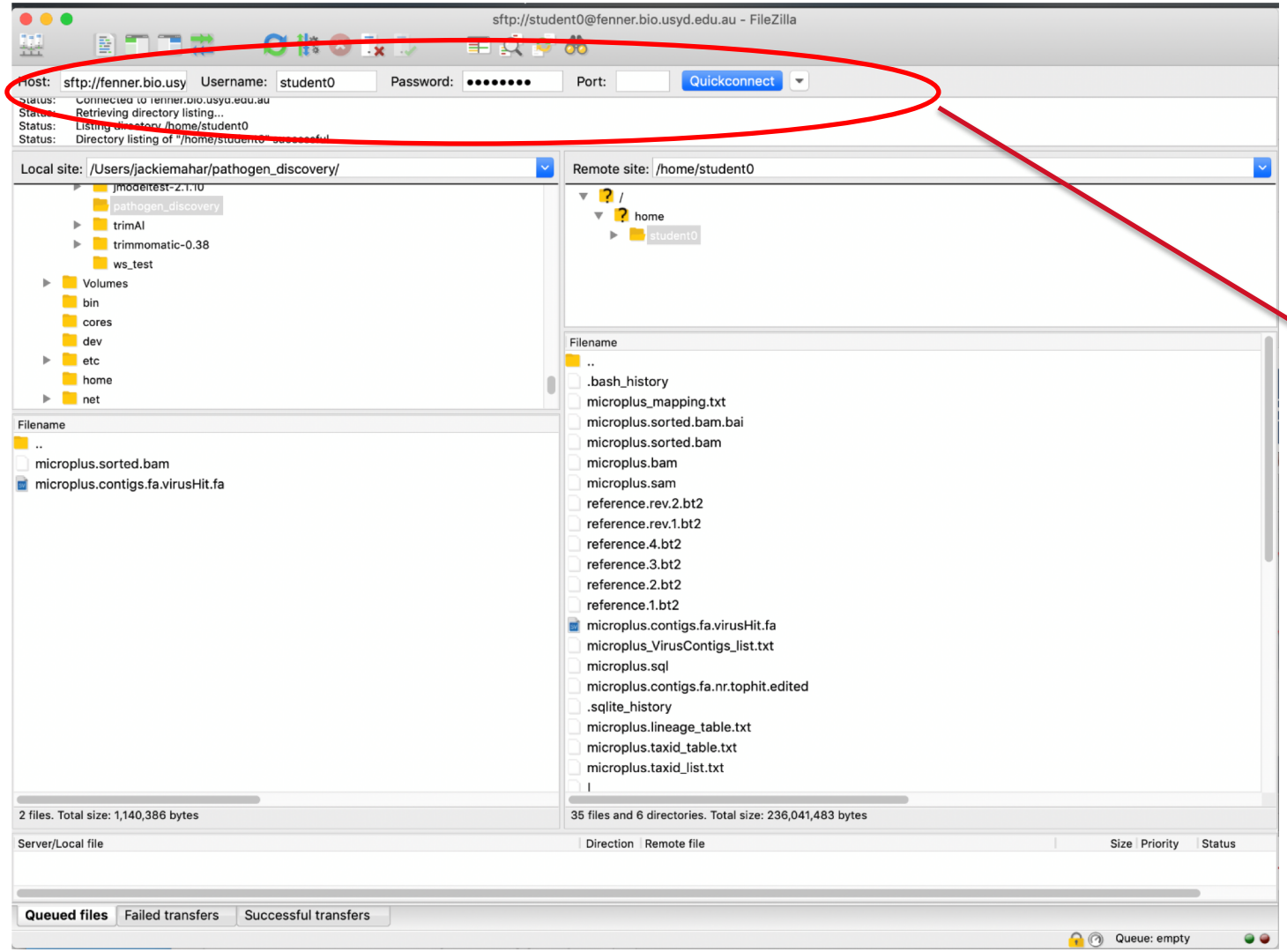
- That completes the workshop
- Questions and comments are welcome

mang.shi@sydney.edu.au

jackie.mahar@sydney.edu.au



Setup file transfer system



If you want to transfer your results files to your own laptop for further practice:

We will use FileZilla for file transfer

Fill in the host: fenner.bio.usyd.edu.au
Username: student<number>
Password: student<number>
Port: 22
Click "Quickconnect"