conda

SPAdes assembly

Today we are only briefly introducing the assembly with **SPAdes**. SPAdes is already installed in our system (as in any CLIMB VM, I think), but it this wasn't the case we could simply `conda install spades` to have it installed by Miniconda.

Read the manual first

There is a (confusing) online manual for SPAdes, but we are nerd enough to read the instructions from the shell. Since the program is writing its output on a different channel (called standard error) we can't simply pipe it into less, we need to add an extra character (&) to have the redirection of the standard error:

```
spades.py |& less -S
```

(as always we can scroll the text with arrow keys then quit with q to return to the shell prompt).
Here's an extract of the manual:

```
SPAdes genome assembler v3.11.0

Usage: /home/linuxbrew/.linuxbrew/bin/spades.py [options] -o <output_dir>

Basic options:
-o      <output_dir>    directory to store all the resulting files
(required)
--meta                  this flag is required for metagenomic sample data

Input data:
--12    <filename>      file with interlaced forward and reverse paired-end
reads
-1      <filename>      file with forward paired-end reads
-2      <filename>      file with reverse paired-end reads
-s      <filename>      file with unpaired reads
```

Perform the assembly

Default parameters: auto k-mer choice

```
spades.py -1 /bsb/denovo/phage/reads/shotgun1.fq -2
/bsb/denovo/phage/reads/shotgun2.fq -o ~/bsb01/phage_default/
```

If you want to see the output folder, there is an online version, in particular you can see:

1. spades.log - this is the text that SPAdes writes to the terminal during the execution to keep us

updated on the progress. Generally non so useful, but we can discover which k-mer settings have been used!

2. contigs.fasta - usually the output we are mostly interested in: the contigs!

Default parameters: auto k-mer choice

We can perform a second assembly with k-mers set of our choice. We can compare results using different k-mer sets in our group. K-mers have to be odd!

Here an example:

```
spades.py -1 /bsb/denovo/phage/reads/shotgun1.fq -2
/bsb/denovo/phage/reads/shotgun2.fq -o ~/bsb01/phage_29,47,51,59/ -k
29,47,51,59
```

As you can see I specified as output directory, a directory that helps me reminding which k-mers have been used. In this case maybe not elegant, but it's just to stress the concept of choosing useful nonambiguous names.

Pre-made output

If you want to save some time there is a pre made output from the step above here:

```
/bsb/denovo/phage/spades/
```

You can evaluate the assembly metrics with this command:

```
seqkit stats --all /bsb/denovo/phage/spades/contigs.fasta
```

Or if you made more than one assembly in your home directory, using “phage_” as prefix:

```
seqkit stats --all ~/bsb01/phage_*/contigs.fasta
```

This will work if the suggested directory structure has been used. If you made customisations, tune the paths accordingly. Example output:

```
file                             num_seqs  sum_len  min_len    avg_len
max_len  sum_gap      N50
phage_51,65,77,85/contigs.fasta         3  114,163    5,534  38,054.3
62,926        0   62,926
phage_21,29,47,59/contigs.fasta         1  113,939  113,939   113,939
113,939        0  113,939
phage_29,47,51,59/contigs.fasta         1  113,939  113,939   113,939
113,939        0  113,939
phage_default/contigs.fasta             1  113,957  113,957   113,957
113,957        0  113,957
```