Hands-on Instruction in Sequence Assembly

Part 1. Download sequence files in fastq format from GenBank Sequence Read Archive.

1. Go to GenBank Sequence Read Archive - http://www.ncbi.nlm.nih.gov/sra.
2. In the search field, type in the following: Pinus thunbergii
3. Click on record SRX014012 (this should be the second record).
4. On the right side of the screen, click on "Download data for this experiment SRX014012".
5. Download the file SRR030730.fastq.gz to your computer
6. Go back to the Sequence Read Archive using the back button/arrow on your browser, and search for Pinus lambertiana.
7. Click on record SRX014023 (this should be the first record).
8. On the right side of the screen, click on "Download data for this experiment SRX014023".
9. Download the file SRR027096.fastq.gz to your computer.
10. The downloaded files now need to be unzipped. If you are using a Mac, simply double-click on each zipped file and it should unzip. If you are using a PC/Windows, you will probably have to download a utility to unzip the files. We use and recommend the simple program 7-zip, available free for download at: http://www.7-zip.org/download.html.

Part 2. Download the annotated reference sequence from GenBank.

1. From GenBank, search CoreNucleotide for Pinus thunbergii chloroplast, complete genome.
2. Scroll down and click on the record for the Pinus thunbergii chloroplast genome (NC_001631.1 GI:7524593).
3. In the upper right of your screen, click on the highlighted 'Send', and select Complete Record: File: Format GenBank. Then 'Create File' and download to your computer. The file should save as sequences.gb.

Part 3. Uploading the read and reference files into CLC Genomics Workbench.

1. Open CLC Genomics Workbench.
2. In the navigation area (upper left), right-click (windows) or ctrl+click (Mac) on the folder CLC_Data and create a new folder called Reads and Reference.

3. Repeat the previous step twice to make two more folders called De Novo Assembly and Reference-Guided Assembly.

4. From the dropdown menu, select File: Import High-Throughput Sequencing Data: Illumina.

5. In the import window, select one of your unzipped fastq files (or you can select both at the same time if they are in the same location) and click the Next button (we will leave the General Options and Illumina Options at their defaults).

6. Click the Save option for Result Handling and leave the Make Log box unchecked, then click Next again.

7. Make sure the Reads and Reference folder is selected and click on the Finish button. It should take a couple of minutes to load the fastq files, as they are very large. When you have fully imported both of your read files, proceed to the next step.

8. Again make sure the Reads and Reference folder is highlighted in the Navigation area, then return to the File dropdown menu, and select Import.

9. In the file selection window, select the Pinus thunbergii reference file you downloaded from GenBank, and select Files of Type GenBank (.gbk/.gb/.gp). Click Select, and the file should upload quickly. The file will be called NC_001631, you can rename this to something like thunbergii_ref if you wish, by clicking on the file name once, waiting a second, then clicking on it again and typing in a new name.

CHECKPOINT #1: At this point, you should have created a Reads and Reference Folder (subfolder of the directory CLC_Data) containing the two fastq files and your annotated reference genome, and two empty folders (De Novo Assembly and Reference-Guided Assembly). Take a moment to view the contents of the three files you have loaded, by double-clicking on each one and scrolling briefly through them in the workspace window. Note that the fastq format is similar to fasta format, except that quality scores for each position are included with the positional base calls. Also note that the *P*. *thunbergii* reference sequence contains all of the annotations present in the original GenBank reference – cool!

Part 4. Parsing the read files based on barcode.

In this exercise, you will parse two accessions of interest by their identifying 3-nucleotide barcodes/tags. Each of the read files you will parse contains reads from a single lane of sequencing with barcoded reads from four different accessions.

1. Select the SRR027096 read file in the Navigation Area.

2. From either the dropdown or the Toolbox window (lower left corner), select High-Throughput Sequencing: Multiplexing: Process Tagged Sequences.

3. Select the read file SRR027096 and make sure it is in the Selected Elements box by itself, then click Next.

4. Click on the "+ Add" button, select Linker Type: Barcode, then specify ATT, AAT, CCT, GGT for the Barcode sequences. Click OK.

5. Click on the "+ Add" button again, select Linker Type: Sequence, then change the Min and Max Length to 33. Click OK.

NOTE: *The order of the previous two steps is important. For example, if you did step 5 followed by step 4, the program would search for all sequences of the specified length ending in ATT, rather than starting in ATT.*

6. Click Next. On the following screen, check Remove faulty barcodes and wait for the Preview calculations to finish. Answer the pertinent parts of **Question 1** on the final page of this handout, then click Next.

7. Choose Result Handling: Save, and click Next.

8. Select the Reads and References folder, then click Finish.

9. This should generate the four barcode files. Change the name of the file Barcode: (ATT) to Barcode: (ATT): P_lambertiana, then delete the other three barcode files generated (we will not use these).

10. Repeat steps 1-9 for read file SRR030730, with the following alterations:
    Step 4, Barcode sequences: AAT, CCT, CGT, GGT
    Step 9, Output file name change: Barcode: (AAT) changed to
    Barcode: (AAT):P_thunbergii

CHECKPOINT #2: At this point, you should have sorted eight barcodes out of your read files and renamed two of them as Barcode: (ATT):P_lambertiana and Barcode: (AAT):P_thunbergii. In addition, you should have attempted to answer **Question 1** at the end of this handout. Note that the reads in the sorted files are 33 base pairs long, and have had their barcodes removed. You can view the reads and verify this by opening the files (double-click on each one).

NOTE: As you move through the exercises, we recommend closing any files open in the workspace when you finish each section. This will keep your workspace uncluttered, bigger, and make it simpler to keep track of what file you are viewing at any given time.

Part 5. Filtering a read file based on quality scores.

In this exercise, you will manipulate filtering parameters based on the positional quality scores of the reads, to see how the read pools are affected. CLC uses a Phred scale (or scores converted to a Phred scale) to determine the quality score (Q) for each position of each read. The quality score is then converted into an error probability, such that $Prob_{error} = 10^{Q/-10}$ (meaning that low quality scores have a higher probability of error). CLC's algorithm keeps a running tally of error probabilities along a read and uses this information to trim reads based on quality and user-specified limits. For more information, you can search 'Quality trimming' in the help menu and see the handout on fastq file format and quality scores.

1. Select the file Barcode: (ATT):P_lambertiana in the navigation area.

2. From either the dropdown or the Toolbox window, select High-Throughput Sequencing:Trim Sequences.

3. Make sure the file Barcode: (ATT):P_lambertiana is in the Selected Elements box by itself, then click Next.

4. We will first use the CLC Bio default parameters for Trim using quality scores (0.05), but *unselect* the box for Trim ambiguous nucleotides. Click Next.

5. Since our sequences do not have any of the specified adapters on the following page, click Next again.

6. Check the box for Discard reads below length, and set the value to 25. Click Next.

7. Change Result Handling to Save. Click Next.

8. Make sure the Reads and References folder is highlighted. Click Finish.

9. Double-click on the report to open and examine the report. Fill in / Answer the pertinent parts of **Question 2**.

10. Repeat steps 1-9, but switch the value for Trim using quality scores in step 4 to 0.005 (a higher stringency).

11. Repeat steps 1-9, but switch the value for Trim using quality scores in step 4 to 0.7 (a lower stringency).

Part 6. De novo assembly.

In this exercise, you will perform two de novo assemblies on your *P. lambertiana* read file and briefly explore a nifty freeware short-read assembly viewer called Tablet. De novo assemblies, as the name suggests, do not utilize or depend on a reference genome. Rather, the assembly algorithm is used to map segments of identical or nearly identical sequence shared by multiple reads, allowing them to overlap and form larger contigs. Manipulating

the length and identity requirements of the matching segments (which you will do below) can result in different assembly results in terms of the number, length, and quality of resulting contigs.

1. Select your short read sequence file Barcode: (ATT):P_lambertiana, then from the Toolbox window or dropdown menu select High-Throughput Sequencing: De Novo Assembly.

2. Make sure the file Barcode: (ATT):P_lambertiana is alone in the Selected Elements window and click Next.

3. On the following screen, unselect the Fast ungapped alignment box and the global alignment box and set the mapping parameters as follows, then click Next:

Mismatch Cost	3	(the cost of a mismatch between two overlapping reads)
Limit	7	(total score of alignment must be greater than read length minus this value)
Insertion cost	3	(cost of an insertion in overlapping reads)
Deletion cost	3	(cost of a deletion in overlapping reads)

4. On the following screen, select the following parameters:
   Options: Add conflict annotations
   Conflict resolution: Vote (A,C,G,T)
   Non-specific matches: Random
   Set the minimum contig length to 200, then click Next.

5. On the following screen, select only the following options, then click Next:
   Output options: Map reads back to contigs (slow)
   Result handling: Save
   Log handling: Make log

6. On the final screen, make sure the folder De Novo Assemblies is selected as the output file destination, then click Finish.

7. A log file will open allowing you to monitor the program's progress. The time to complete the assembly will vary depending on your computer's processing capability, but the assembly should finish in ten minutes or less.

8. When the assembly has finished, double-click on the newly created assembly file to open and view the contigs in your assembly. This will give you an idea of the number and sizes of contigs assembled, as well as the number of reads and average coverage for each one.

9. Next you should export your contigs in the ACE file format. To do this, select File and then Export. Select where you would like to save the file and choose ACE files (.ace). You will use this file in step 11 after starting your next de novo assembly run.

10. Now repeat steps 1-8, but change the settings in step 3 to the following:

    Mismatch Cost 2
    Limit 11
    Insertion cost 1
    Deletion cost 1

11. While your second de novo assembly is running, open the Tablet program that you installed prior to the workshop. In the top left corner of the screen that opens, choose Open Assembly. Browse for the ACE file you created in step 9 and open it.

12. Use the Tablet viewer to explore the assemblies of several contigs of varying length.

13. Next you will search among your contigs for a specific gene, *atp*B. At the top of the panel above the list of contigs, choose the magnifying glass in order to open the search feature. Type in the first 18 bp of the *P. lambertiana atp*B sequence (ATGAGAACCAATCCTCTT) as your search string. Select Search for subsequences and Search All Contigs from the menus below the search box. Then click the magnifying glass to search.

14. Tablet will find all of the reads that contain the subsequence you entered and will output a list of contigs where those reads have been mapped.

    How many reads contained your search string? _____
    Which contig contained the *atp*B sequence? _____

15. Click on one of the reads in your results list to see where it is located in the contig. Your search string will be highlighted for a few seconds. Mouse over the highlighted search string to see the full read. Explore several of the reads mapped to this contig.

16. Tablet also shows translations of the consensus sequence in the middle pane of the viewer window. Note the Met corresponding to the start codon (ATG) of *atp*B. You can view translations in all six reading frames by selecting the Advanced tab near the top of your screen and selecting multiple reading frames in the Protein pane.

17. Next you will search among your contigs for the end of the *atp*B gene. Repeat the search steps using a search string of AACATGGAAAATTAA.

    How many reads contained this search string? _____
    Is the entire *atp*B gene contained in a single contig? _____

18. Your second de novo assembly should now be complete. Return to the CLC Genomics Workbench and select your first de novo assembly (from Barcode: (ATT):P_lambertiana), and from the Toolbox window or dropdown menu choose High-Throughput Sequencing: Create Detailed Mapping Report. Make sure this file is alone in the Selected Element window, and click Next.

19. On the following screen, set the Long contigs threshold to 2000 and the Short contigs threshold to 300. You should immediately see the counts for how many of each contig types are present in your assembly. If you're interested, try changing these numbers to get an idea of the size distribution of contigs in your assembly. When you're satisfied, click Next.

20. On the following screen, choose the Save option under Result handling, but leave the other options unchecked. Click Next.

21. Make sure your De Novo Assemblies folder is selected for the output destination, then click Finish.

22. Repeat steps 18-21 for your second de novo assembly.

23. Using the output from your Detailed Mapping Reports, answer all parts of **Question 3**.

CHECKPOINT 3. At this point, you should have explored the effect of quality filtering on your read pools, and completed two de novo assemblies with varying parameter settings on your *P. lambertiana* read file. We will not use material from **Part 5** (Quality Filtering) again, however a de novo assembly of *Pinus lambertiana* (either of the two is fine) will be used in **Part 8** below. Also, all parts of **Question 2** and **Question 3** should have been answered.

Part 7. Reference-guided assembly.

In this exercise, you will perform four reference-guided assemblies using your short read sequence files and *P. thunbergii* as a reference. As with de novo assembly, manipulating mapping parameters (mismatch costs, insertion/deletion costs, etc.) can affect the resulting quality of your assembly. However, the choice of reference can also have an effect, as it is often difficult to map short read sequences from a species to a distantly related (and/or highly divergent) reference.

1. Select the reads file Barcode: (AAT):P_thunbergii, and then select High-Throughput Sequencing: Map Reads to Reference from the Toolbox window or dropdown menu.

2. Make sure your file Barcode: (AAT):P_thunbergii is alone in the Selected Elements window, then click Next.

3. On the following screen, select your *P. thunbergii* reference (NC_001631 or whatever you have renamed it). Click Next.

4. On the following screen, unselect the Fast ungapped alignment box and the global alignment box and set the mapping parameters as follows, then click Next:

Mismatch cost	3
Limit	7
Insertion cost	3

Deletion cost 3

5. On the following screen, select the following parameters, then click Next:
   Options: Add conflict annotations
   Conflict resolution: Vote (A,C,G,T)
   Non-specific matches: Random

6. On the following screen, select only the following options, then click Next:
   Result handling: Save
   Log handling: Make log

7. On the final screen, make sure the folder Reference-Guided Assemblies is selected as the output file destination, then click Finish.

8. A log file will open allowing you to monitor the program's progress on the de novo assembly. The time to complete the assembly will vary depending on your computer's processing capability, but these assemblies should in general be much faster than then de novo assemblies.

9. When the assembly has finished, repeat steps 1-8, but change the settings in step 4 to the following:
   Mismatch cost 2
   Limit 11
   Insertion cost 1
   Deletion cost 1

10. Repeat steps 1-9 using the file Barcode: (ATT):P_lambertiana.

11. Take a few minutes to explore one or more of your reference-guided assemblies by opening them (double-click on them) in the workspace window.

12. Select your first reference-guided assembly (from Barcode: (AAT):P_thunbergii), and from the Toolbox window or dropdown menu choose High-Throughput Sequencing: Create Detailed Mapping Report. Make sure the correct file is in the Selected Element window, and click Next.

13. On the following screen, no parameters are available to choose, so simply Click Next.

14. On the following screen, choose the Save option under Result handling, but leave the other options unchecked. Click Next.

15. Make sure your Reference-Guided Assemblies folder is selected for the output destination, then click Finish.

16. Repeat steps 12-15 for your other three reference-guided assemblies.

17. Using the output from your Detailed Mapping Reports, answer all parts of **Question 3**.

CHECKPOINT 4. At this point, you should have completed four reference-guided assemblies with varying parameter settings. The second (less stringent) reference-guided assembly of *Pinus lambertiana* will be used in **Part 8** below, while the first (more stringent) reference-guided assembly of *P. thunbergii* will be used in **Part 9** below. Also, all parts of **Question 4** should have been answered.

Part 8. De novo vs. reference-guided assembly.

In this exercise, you will contrast de novo and reference-guided strategies in a region of high divergence. You will first map a large contig from your either of your de novo assemblies of *P. lambertiana* from **Part 6** to the reference genome. Then you will compare this assembly/mapping to your second (less stringent) reference-guided assembly of *P. lambertiana* from **Part 7**.

1. Open (double-click) your de novo assembly of Barcode: (ATT):P_lambertiana so that you see the table of contigs. Now sort the table based from largest to smallest length by clicking twice on the Length of consensus sequence tab at the top of the table.

2. There should be a large contig of size 4458 or 4459 base pairs within the top three longest contigs. Select this contig, then click on the Open Consensus button to open its sequence. Note also the fairly high value for average coverage (~300X).

3. Check the sequence at the beginning of the contig. Starting from the first or second position of the contig, it should read: TATGGGGAAGTATGGT… If you see this sequence, you have the correct contig and can proceed to the next step. If you don't, please let one of the instructors know and they will help you find the correct contig before moving ahead.

4. From the dropdown menu, select File: Export. Export this file to your desktop in FASTA format. It should save as the file Consensus from Contig 2.fa.

5. Select your Reads and Reference folder in the Navigation Area. From the dropdown menu, select File: Import, and import the file Consensus from Contig 2.fa back into the Workbench from your desktop.

6. Select both your reference sequence (NC_001631 or whatever you renamed it) and the file ConsensusfromContig2 in your Reads and Reference folder. You can select both at the same time by selecting one of the files, then holding down the Ctrl button and clicking on the other file.

7. From the Toolbox dropdown menu or window, select Alignments and Trees: Create Alignment.

8. Make sure your reference sequence and contig file are both in your Selected Elements window, and click Next.

9. Set your Gap open cost at 5, and leave the other parameters at their default settings. Click Next.

10. Select Save for Result handling and click Next.

11. Select your De Novo Assemblies folder for output and click Finish.

12. After a few minutes when the alignment process is complete, you should have the file ConsensusfromContig2_alignment in your De Novo Assemblies folder. Double-click on this file to open it.

13. Take a moment to understand the format of the alignment. You should see groups of five rows containing (from top to bottom): the alignment of your de novo contig, the reference sequence, the consensus of the two sequences, a conservation plot, and a sequence logo graphic.

14. Now, scroll down to alignment position 113750 where alignment of the contig starts. Alternatively, you can also use the Find function in the Read Mapping Settings window on the right instead of scrolling to the start of the alignment. Now scroll briefly to the end of the contig's alignment at alignment position 118379. Note that there are several fairly large indels, for example those at alignment positions 15101-15164 and 15787-15832.

15. Return to alignment position 115100, where the first large indel begins. What nucleotides are called in your aligned contig between alignment positions 115120 and 115160?

    _____

    What nucleotides are called in your aligned contig between alignment positions 115118 and 115202?

    _____

16. Now compare this to your *second* reference-guided assembly of *P*. *lambertiana*. Specifically look at alignment positions 115080-115100 and 115180-115200. From an eyeball estimate, what would you say is the consensus sequence of the nucleotide sequences that are *masked* in your short reads between the following positions? (You can see these by scrolling vertically in the assembly)

    115080-115100 _____

    115180-115200 _____

17. Now answer all parts of **Question 5**.

Part 9. SNPs and DIPs.

In this exercise, you will run SNP (single-nucleotide polymorphism) and DIP (deletion-insertion polymorphism) analyses on one of your reference-guided assemblies of *P. thunbergii*. This will allow you to see general patterns in SNPs and DIPs occurrence, and what factors might affect your confidence in these calls.

1. Select the first reference-guided assembly of *P. thunbergii* from **Part 7** (file Barcode: (AAT):P_thunbergii mapping) in the Navigation Area, and then select High-Throughput Sequencing: SNP Detection from the Toolbox window or dropdown menu.

2. Make sure your file Barcode: (AAT):P_thunbergii mapping is alone in the Selected Elements window, then click Next.

3. On the following screen, leave the parameters at their default values, but set Significance: Minimum variant frequency (%) to 51 and Ploidy: Maximum expected variations to 1. Click Next.

4. On the following screen, leave the default parameters, but change Output options: Genetic code to 11 Bacterial and Plant Plastid and change the Result handling to Save. Click Next.

5. Make sure your folder Reference-Guided Assemblies is selected for the output location, then click Finish.

6. When the SNP Detection process is finished, double click on the SNP table to open it. The SNP output file should be called Barcode: (AAT):P_thunbergii mapping SNP Detection Table.

   How many SNP calls in total were made? _____
   Assuming unassigned ORFs (PithCP017, 026 and 032) are not real coding loci, how many missense mutations were called? _____
   How many nonsense mutations were called? _____

7. Sort your SNP table by frequency so that the 100.0 frequencies are at the top. Find the highest count for a100.0 frequency SNP call. This should be 139 counts for a Ser826Ala change in *rpo*B. Open (by double-clicking on) your read mapping assembly (file Barcode: (AAT):P_thunbergii mapping). Using the Find function under the Read Mapping Settings window on the right, search for the position of this SNP (position 23486). Take a moment to scroll down through the mapped reads to see the depth of this SNP call.

8. Now go back to your SNP table and find the lowest count for a 100.0 frequency SNP call. This should be six counts for a Tyr83Asn change in a putative coding locus PithCP032. Again using the Find function in the Read Mapping Settings window, search for the position of this SNP (29208) in your assembly. Take a moment to scroll down through the mapped reads to see the mapping of this SNP call. Now answer the pertinent parts of **Question 6**.

9. Now you will generate a DIP summary. Select the same reference-guided assembly you just worked with to generate your SNP report (Barcode: (AAT):P_thunbergii mapping), then select DIP Detection from the Toolbox window or dropdown menu.

10. Make sure that the file Barcode: (AAT):P_thunbergii is alone in the selected elements window, then click Next.

11. On the following screen, set your Minimum coverage at 4, the Minimum variant frequency (%) at 51, and the Maximum expected variations at 1. Leave the Advanced box unchecked. Click Next.

12. On the following screen, leave the options at their default, except change the Result handling to Save. Click Next.

13. Make sure your Reference-Guided Assemblies folder is selected for the output location, then click Finish.

14. When the DIP Detection process is finished, double click on the DIP table to open it. The DIP output file should be called Barcode: (AAT):P_thunbergii mapping DIP Detection Table.

    How many DIP calls in total were made? _____
    Assuming unassigned ORFs (PithCP004 and 031) are not real coding loci, how many frameshift mutations were called? (read the annotation descriptions carefully!)
    _____

15. Again in your read mapping assembly (file Barcode: (AAT):P_thunbergii mapping) use the Find function under the Read Mapping Settings window on the right, and search for the position of the frameshift mutation in *atp*A. Take a moment to scroll down through the mapped reads to see the depth of this DIP call.

16. Now answer the remaining parts of **Question 6**.

CHECKPOINT 5. At this point, you should have completed all of the exercises in the handout, and answered fully all of the questions at the end of the handout. Congratulations, you are now a Next-Gen sequencing expert! Now, get out there and start sequencing!

Hands-on Instruction in Sequence Assembly: Questions

Question 1. Read sorting by barcode.

How many total reads are in your read files?

SR027096 _____

SR030730 _____

How many reads in each file have failed barcodes (were not assigned to a barcode)?

SR027096 _____

SR030730 _____

Since all of the barcodes are 3 base pairs long, can you use these numbers to estimate error rate in the first three positions of your sequence reads?

SR027096 _____

SR030730 _____

Would you expect this error rate to increase or decrease further along the read? Why?

Question 2. Filtering read pools with quality scores.

For each value of the 'Trim using quality score' parameter, answer how your read pool was affected as follows:

Parameter value	Total reads before trimming	# reads trimmed	Total reads after trimming	Avg. read length before trimming	Avg. read length after trimming
0.05					
0.005					
0.7					

Would you guess that more stringent quality filtering will give a better assembly, or would it be better to throw as much data as possible at the assembly and use relaxed (or no) quality filtering? Ask one of the instructors and see what they think.

Question 3. De novo assemblies.

For the following questions, each de novo assembly is designated as follows:

A. *P. lambertiana* – more stringent assembly parameters (first de novo assembly)
B. *P. lambertiana* – less stringent assembly parameters (second de novo assembly)

How many contigs were created from your read pool?
A._____ B._____

How many reads were successfully used in contig assembly (i.e. matched)?
A._____ B. _____

What percentage of total reads included in the analysis was this?
A._____ B._____

What was the average length of contigs in the assembly?
A._____ B._____

How long was the longest contig assembled?
A._____ B._____

Which of the above summary statistics was most affected by making the assembly parameters less stringent in the *P. lambertiana* de novo assembly? Why do you think this is?

The N50 metric is a commonly used statistic to gauge de novo assembly quality. There are several differing definitions of N50. Many groups, including CLC Bio, determine the N50 by sorting a set of contigs from longest to shortest and sequentially summing their lengths until the total is at least 50% of the sum of all contigs. The length of the shortest contig in this set is the N50 value for the assembly.

What is the N50 for each of your assemblies?
A._____ B._____

Based on all of the above summary statistics, which de novo assembly appears to be the most successful? Why?

Question 4. Reference-guided assemblies.

For the following questions, each reference-guided assembly is designated as follows:

A. *P. thunbergii* – more stringent assembly parameters (first reference-guided assembly)
B. *P. thunbergii* – less stringent assembly parameters (second reference-guided assembly
C. *P. lambertiana* – more stringent assembly parameters
D. *P. lambertiana* – less stringent assembly parameters

How many reads were mapped to the *P. thunbergii* reference sequence?
A._____ B._____ C._____ D._____

What was the length of the consensus sequence for the mapped reads?
A._____ B._____ C._____ D._____

What fraction of the reference was covered?
A._____ B._____ C._____ D._____

What is the average coverage depth?
A._____ B._____ C._____ D._____

How many non-specific matches (reads aligning equally well to multiple places in the reference) were found?
A._____ B._____ C._____ D._____

How many non-perfect matches (reads with at least one mismatch or gap) were found?
A._____ B._____ C._____ D._____

Of the four reference-guided assemblies, which looks the most and which the least successful?

P. thunbergii and *P. lambertiana* are separated by perhaps 80 million years of evolution. Do you see any evidence of this in the effectiveness of reference-guided assembly using *P. thunbergii* as a reference for both read pools? What regions of the genome would you predict to have more or less success in assembly using a reference-guided approach in the case of *P. lambertiana*?

Question 5. De novo vs. reference-guided assemblies.

What pieces of evidence suggest the gap in reference-guided assembly between alignment positions 115096 and 115189 is incorrect?

Where might you expect to find the majority of assembly errors in de novo contigs – in the middle or at the ends? Why?

Where might you expect to find the majority of assembly errors in reference-guided assembly? Why?

When would you choose de novo over reference-guided assembly, or vice-versa? Or, would you always use both?

Question 6. SNPs and DIPs.

Of the two SNP calls you investigated - Ser826Ala in *rpo*B (depth 197) and Tyr83Asn (depth 6) in PithCP032 – which do you think is more believable? Why?

Do you think that call frequency or depth is more important in calling SNPs with confidence from short read sequence data?

What possibilities might account for SNPs called (or nearly called) with near 50% frequency in a chloroplast assembly? For an example, see the SNP call in your table at Reference position 29271.

In the vicinity of the SNP call in *rpo*B, you should notice that there are a number of gaps in the consensus sequence of the mapped assembly. Are any of these apparent indels also present in your DIP report? Why or why not?

In the single DIP that leads to a frameshift mutation (reference position 10102, atpA), where in the coding region does the frameshift fall?

Do the coverage and frequency of the call make this DIP more or less believable to you?

Challenge! Assume the DIP is real. What effect does this ultimately have? Hint: continue working your way along the consensus sequence in the direction of translation, one codon at a time.

Essential Linux

Compiled by Aaron Liston and Todd Mockler, Oregon State University

File and Directory Management

ls – List directory contents. **ll** – List directory contents and file details.

***** – Match any character in a directory or file name. For example, **ls *.fsa** lists all files that end with fsa.

mkdir – Make a directory.

cd – Change the current directory. Use **cd ..** to move up one directory.

cp – Copy files and directories. **mv** – Move (and rename) files and directories.

rm – Remove files and directories. By default, you must type **y** to approve each. You can turn this off with **rm –f**, but be very careful, as there is no recycle bin!

ln – Creates a link to a file. Used to access a single file from multiple locations.

Tab Key – Autocomplete names of files in a directory.

Up Arro **w** – Go back to previous commands.

Ctrl – u – Erase a command line.

dos2unix, mac2unix – Converts line return characters from Windows and Mac files to Linux.

chmod – By default, files can by read by all users, but only modified by their creator (you). To allow all other users to modify a file, type **chmod a+w filename** (**g+w** for only members of your group). To make a file executable (e.g. a shell script) type **chmod +x filename** (**g+x** for your group, and **a+x** for all users).

df -h – Reports how much disk space is available on your drive. **du -h** – Disk space available in a folder.

File Viewing

cat – Concatenate files and prints files to the screen (cat –A shows hidden characters, e.g. tabs).

less – The best way to view file contents (unless the file is very large).

head – Outputs the first part of files, by default the last ten lines (useful for large files).

tail – Outputs the last part of files, by default the last ten lines (use on output files to view progress).

wc – Prints the number of bytes, words, and lines in files.

q – In many programs, e.g. less, hitting 'q' will exit the program.

File Manipulation

& – Appending the & to the end of the command line will make the job run in the background.

> – Direct the output of a command to a file.

>> – Appends to an existing file instead of creating a new file.

| – Pipe the outout of a command to another command.

grep – Searches a file (or standard input) for lines containing a match to a pattern. Examples:

grep ^ACGT filename > output will move all lines in "filename" that begin with ACGT to "output".

grep –c ^ACGT will count the number of lines that begin with ACGT.

sed – Finds and replaces text in a file. Example:

sed s/A/T/g filename > output will replace all occurences of A with T in "filename".

sort - Sorts the lines of text files. A numeric sort for field (key) 3 is specified with **sort –n –k3**
uniq – Finds (and optionally counts) the unique lines in a sorted file. For example:
sort filename | uniq –c | sort –n –r > output
will sort and count the data in a file, then sort by reverse numeric order of the count.
cut – Copies one or more fields (by default, delimited by tabs) in a file.

File Compression and Compilation

gzip – To compress a single file. The compressed file(s) will be given the suffix **.gz**
gunzip – To extract a gzip compressed file.
tar – To combine and compress multiple files, or to separate and uncompress files. Used for installing programs such as Velvet: **tar -zxvf velvet-latest.tgz**
make – To compile a set of programs specified in a file named **makefile.**

Miscellaneous

history - Displays the command you have used in a session (very useful for documenting your work).
users – Displays other users on this computer.
top – Displays all jobs running on this computer.
ctrl+c – Terminate a program or job running in the foreground.
man – Displays a brief manual page for any command.
For example: **man cat** - shows the usage and options for the command 'cat':
info – Displays detailed manual pages for most commands.

Regular Expressions

Used to denote search terms that are not alphanumeric. Useful in commands such as **grep** and **sed**.
For example, in the **grep** command above, the **^** denotes "beginning of a line". Likewise, **$** denotes "end of a line". For a good introduction to regular expressions, see
http://www.zytrax.com/tech/web/regex.htm