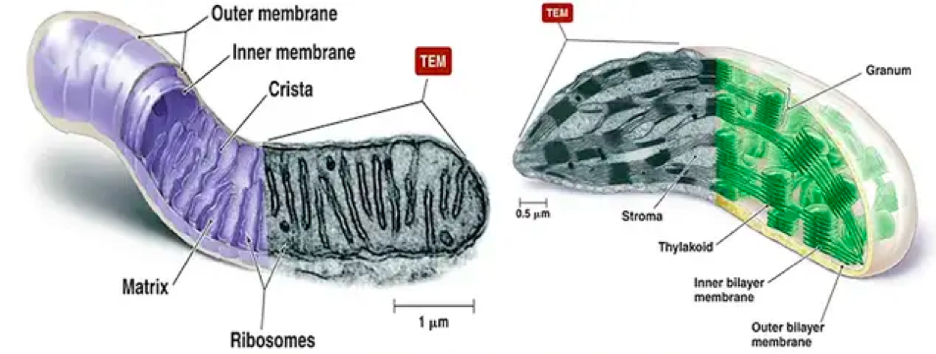
Genetic Incompatibility Lab Introduction

Bio 240 Genetics Lab

**Organelle Genomes and Cytonuclear Adaptation**

The evolution of plant cells was the result of two distinct evolutionary steps that led to the incorporation of two membrane bound organelles with metabolic function, the mitochondria and the chloroplast (Figure 1).



© 2020 Pearson Education, Inc

|  |
| --- |
|  |

**Figure 1**: Double membrane bound organelles, the mitochondria (left) and the chloroplast (right). Black and white images represent transmission electron microscopy images of the organelles.

First, endosymbiosis between an alpha-proteobacterium and an organism with no organelles produced eukaryotic cells with membrane bound mitochondria. The second evolutionary step was the endosymbiosis of photosynthetic cyanobacteria by these initial eukaryotic cells which led to eukaryotic cells with both mitochondria and chloroplasts (Buclar and Fujii, 2012; Figure 2).

|  |
| --- |
| © 2010 Nature Education |

**Figure 2:** The origin of the mitochondria and chloroplasts. Mitochondria resulted from an endosymbiotic relationship between an aerobic prokaryote and an organism with no organelles. Chloroplasts arose from eukaryotic cells that engulfed photosynthetic cyanobacteria.

As a result of arising from prokaryotic endosymbiosis, organelles and prokaryotic species share the following traits:

1) DNA is found in nucleoid regions and is not packaged into chromosomes.

2) DNA is circular and not packaged around histone proteins

3) DNA has few non-coding regions

4) There are usually multiple copies of the genome in the organelle

5) Polycistronic – long mRNA that encode for multiple proteins

Over time, these organisms became integrated into the host cell as organelles, and a large amount of the endosymbiont’s DNA was relocated to the host’s nucleus or lost. Some genes related to organelle metabolism (both cellular respiration and photosynthesis) and regulation (transcription, translation, and proteostasis) were retained by the organelles. These processes thus require close interaction between proteins encoded by genes in both the nuclear and organelle genomes (Fig 3). Consequently, tight cooperation between the nuclear and organelle genomes is necessary for cells to carry out the essential functions of respiration and photosynthesis. Accordingly, if a mutation arises in one genome (organelle or nuclear) that leads to a decrease in function, there will be selection for a compensatory mutation in the other genome to restore function. These cyclical dynamics mean that organelle and nuclear genomes remain closely matched within populations but can become increasingly divergent between populations.

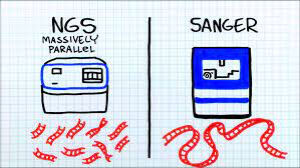


**Figure 3:** Rubisco, the most abundant protein on the planet and critical for photosynthesis is encoded by both a small, nuclear-encoded subunit (rbcs) and a large, chloroplast-encoded subunit (rbcl). A) A ribbon diagram showing how two copies of rbcs (purple and pink, on the outside edges) and two copies of rbcl (orange and green, on the inside) interact to form part of the full rubsico protein. The ribbon diagram is taken from RCSB’s Protein Databank (PBD). B) A diagram showing the same 4 subunits as in A, but now the residues are colored by whether they are chloroplast encoded (green), nuclear encoded (yellow), or nuclear encoded contact residues (red, nuclear encoded residues that directly contact chloroplast encoded residues). Image from Forsythe et al., 2019. CYMIRA: The cytonuclear molecular interactions reference for Arabidopsis. Genome Biology and Evolution. 11(8):2194-2202.

As a side note, we often think of organelle acquired mutations as being deleterious, but this isn’t necessarily true. In fact, often organelle mutations are beneficial to the populations in which they are fixed and allow them to adapt to their environment.

As time passes, the accumulation of mutations in populations of plants separated geographically can lead to reduced fitness in progeny when these populations are crossed, as it can lead to mismatched nuclear and organelle genomes. This negative interaction between proteins encoded by divergent genes is called **genetic incompatibility** and the resulting decrease in fitness is referred to as **reproductive isolation.** As reproductive isolation increases, the divergence between these populations will eventually result in the inability to cross and produce viable/fertile offspring, which results in **speciation** (the formation of new species).

**Sanger vs Next Generation Sequencing**



Thermo Fisher Scientific; Seq It Out

Last week, we sent our ura3 mutants off for Sanger sequencing. Sanger sequences is similar to PCR but creates different lengths of DNA with fluorescent chain-terminating ddNTPs, each nucleotide with a distinct fluorescent tag. Once the ddNTPs are integrated, the amplified DNA fragments are separated by size using capillary gel electrophoresis. This gel is then read by a computer with a laser that is designed to excite the fluorescent tags. The computer then calls each nucleotide in the sequence and forms a chromatograph. With Sanger sequencing, we have to have specific primer pairs and only end up amplifying one gene. Sequencing entire genomes using this method is time consuming and costly. One sanger sequencing read is 500-1500 bp, but the chloroplast genome is 200,000bp, which would require over 130 sets of primer pairs and 260 sequencing reactions to get just a single sequence for each region of the genome. Once completed, we would have to try to piece back together all those single sequences to obtain the full genome sequence.

Using Next Generation Sequencing (NGS) we can sequence the entire chloroplast genome in one sequencing run. The NGS sequencer we will be using for this project is called the MinION. The MinION is a small and portable device that measures the change in the magnitude of electric current that is generated as DNA passes through bio-pores in a flow-cell. Each flowcell has hundreds of bio-pores which dramatically increases the rate in which base-pairs are called. Regardless of which NGS technology you use, you have to computationally assemble the genome from individual reads.

Please check out this video that highlights the difference between Sanger and Next Generation Sequencing

<https://www.youtube.com/watch?v=Wpww8bb63zU>

**Introduction to Coding**

When we looked at our ura3 sequences this week, we were looking at a small data set. When you’re working with only a few sequences, it doesn’t really take much computing power to process. This changes when we start to look at the results from NGS. These sequencers generate huge data sets quickly, and as the size of those data sets increase, so does the need for increased computing power. As the capabilities of our computers likely vary, we will be using a Google Colab notebook in the coming weeks. Personal computers often only possess a single microprocessor resulting which would result in very long processing times with large data sets. Google Colab is a cloud-based tool that can be used when you have large processing needs.

To communicate what we need to these computers; we use what is called a command-line interface. More simply, a command-line interface is a text-based method of instructing your software or computer to complete a task. Each of these text instructions given to the computer is called a command. Commands allow us to more efficiently apply analyses to large data sets without wasting the computing power required for a graphic user interface, where computing power is spent to create visuals and allow the user to manually complete these tasks. The computer completes what to us would be a repetitive and tedious endeavor in a fraction of the time.

Commands follow the same basic structure regardless of the program and analysis being used. First, we call the program by typing in its name. Then we pass the program a series of arguments/options followed by answers. Arguments are preceded by single or double dashes, and all sections of the code (calling the program, arguments, and answers) are separated by spaces. Because of this, it is important to not include any spaces in any individual section of code. For example, you could not call your fastq file in the example below “AR reads.fastq”. If you did that, the program would see it as two different answers to the –fastq argument. Also, it is important to remember you must be precise in command line coding. Names of files and arguments must be entered exactly and are case sensitive.



**Figure 4:** Example of a command given on the command line to call the program NanoPlot and ask it analyze the NGS reads/sequences found in the ARreads.fastq file, giving the prefix “AR” to each output file. Sections of code are labelled according to whether they are the call to the program, arguments, or answers to arguments.