Guide Finder Draft Genome Example Exercise:

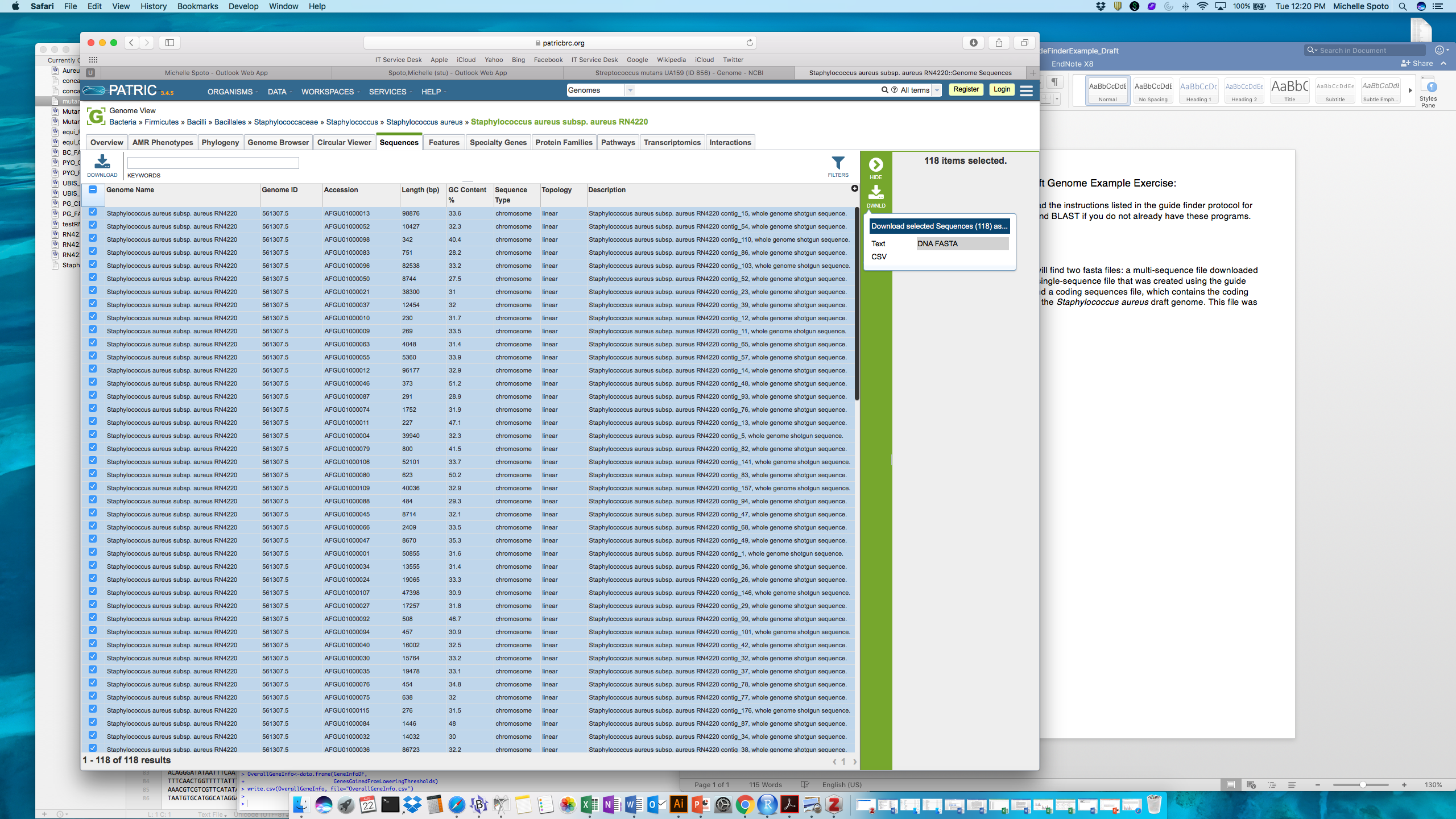
Prior to beginning the example exercise, read the instructions listed in the guide finder protocol for draft genomes. Download the R, RStudio, and BLAST if you do not already have these programs.

1. Identify required files as information

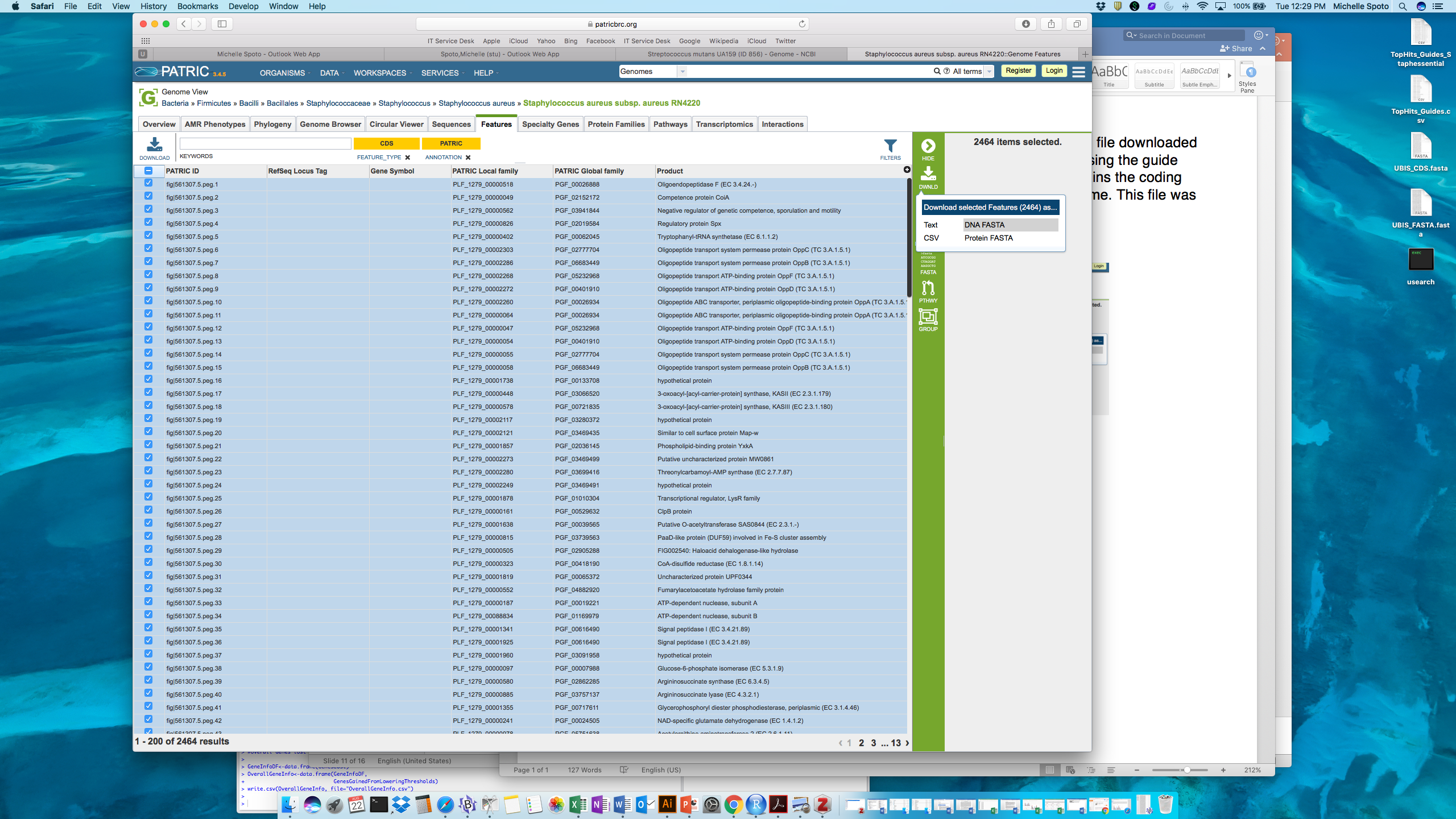
* Multi-sequence fasta file
* Coding sequences file

In the “Draft Genome Example” folder you will find two fasta files: a multi-sequence file downloaded directly from PATRIC and a concatenated, single-sequence file that was created using the guide finder pre-processing script. You will also find a coding sequences file, which contains the coding sequence of each gene (5’🡪 3’ direction) in the *Staphylococcus aureus* draft genome. This file was downloaded directly from PATRIC. The processing for downloading these two files is shown below, for reference.

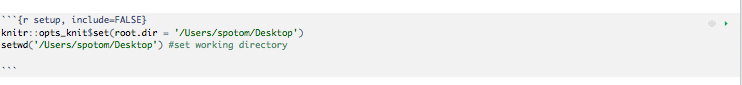
Downloading multi-sequence fasta file from PATRIC:



Downloading coding sequence file from PATRIC:

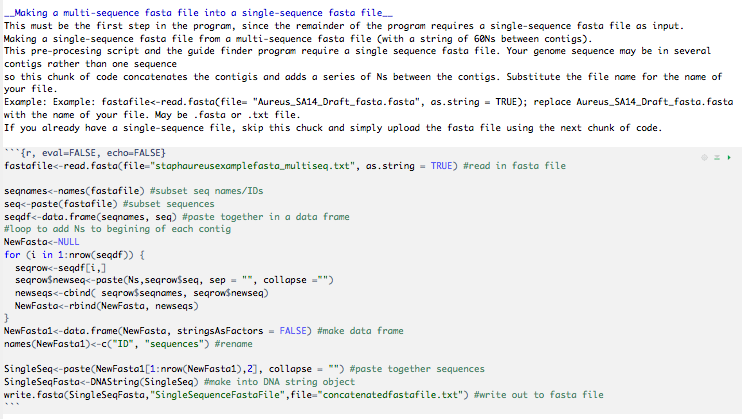


2. Open the pre-processing script with the RStudio program and set the working directory   
The working directory is where all of the files you input into this program should be kept and where all files output by this program will save to. For ease, you can set your working directory to your Desktop, similar to the example shown below, if desired. Set the working directory by identifying the file path to this location. Press the green arrow to run this chunk of code



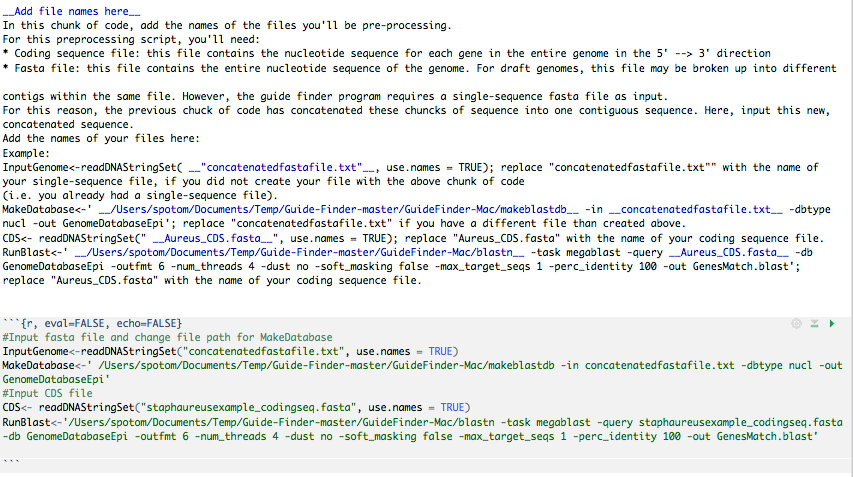
3. In the pre-processing script, fun the next three chunks of code (“Install Packages”, “Load Packages”, and “Functions Created”). You can do this by pressing the green arrow at the top of each chunk. You do not need to make any edits to any of these sections of code. Note that you only need to install the packages once and you only need to load the packages each time you run the pre-processing finder script.

4. In the pre-processing script, Input the mult-sequence fasta file and run the chunk of code to make into a single-sequence file.   
Do this by including the name of your fasta file (staphaureusexamplefasta\_multiseq.txt) in the quotation marks within the read.fasta function (as shown below), if the file name is not already there. Run this chunk of code by pressing the green arrow. Wait until the chunk is entirely done running before proceeding with the next step. The output of this chunk is a file (saved to your working director) called “concatenatedfastafile.txt”, which contains the nucleotide sequence of the input genome concatenated into one sequence, with a series of 60 N’s between each contig. **This single-sequence file will be the working fasta file for the remainder of the pre-processing script and the main guide finder script. It will be saved to your working directory after completion of this step and then re-input into the program in Step 5.**



5. Set paths and input files  
**A**. As shown in the first red box, input your fasta file by including the name of the single-sequence fasta file (concatenatedfastafile.txt) in the quotation marks within the readDNAStringSet function.  
**B.** As shown in the second red box, input your fasta file by including the name of the single-sequence fasta file (concatenatedfastafile.txt) within MakeDatabase  
**C.** As shown in the first blue box, set the file path for the location of the makeblastdb file.  
**D.** As shown in the second blue box, set the file path for the location of the blastn file.   
**E.** As shown in the first green box, input the coding sequences file (staphaureusexample\_codingseq.fasta) in the quotation marks within the readDNAStringSet function  
**F.** As shown in the second green box, input the coding sequences file (staphaureusexample\_codingseq.fasta) within RunBlast

Run this chunk of code by pressing the green arrow



6.Run the rest of the code chunks!   
Run each of the code chunks by pressing the green arrow.

7. Identify the output file  
The output file, DraftGenome\_NewCoordinates.csv, contains a list of each individual gene ID, the start and end coordinates for each gene, and the strand on which the gene is coded (+/-). Check this output file against the file provided (aureusDraftExample\_DraftGenome\_NewCoordinates.csv), to make sure that the preprocessing script was run correctly; they should be the same. Also, check the concatenated file (created in Step 4) and check this against the example concatenated file (EXAMPLEconcatenatedfastafile.txt); they should be the same.

**8. Open the main Guide Finder script in RStudio.**

The remainder of the instructions will guide you through the main Guide Finder script, using the pre-processed files created above.

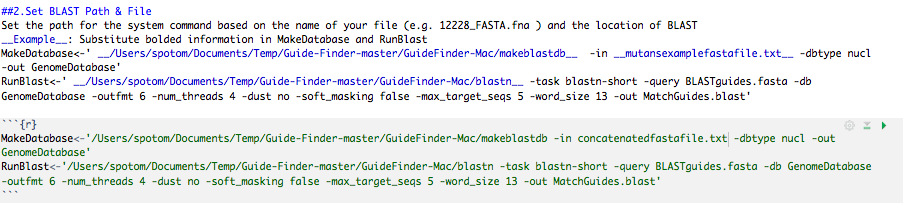
9. Set the working directory.   
The working directory is where all of the files you input into this program (DraftGenome\_NewCoordinates.csv and concatenatedfastafile.txt) should be kept and where all files output by this program will save to. For ease, you can set your working directory to your Desktop, similar to the example shown below, if desired. Set the working directory by identifying the file path to this location. Press the green arrow to run this chunk of code



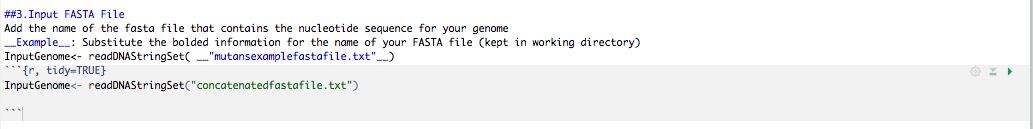
10. Run the next three chunks of code (“Install Packages”, “Loading Packages”, and “Lists Functions”). You can do this by pressing the green arrow at the top of each chunk. You do not need to make any edits to any of these sections of code. Note that you only need to install the packages once and you only need to load the packages each time you run the guide finder script.

11. Set the BLAST file path and the name of the fasta file in the “Set BLAST Path & File” chunk   
**A**. As shown in the red boxes, set the file path for the location of the makeblastdb file and the blastn file.   
*If you do not know how to identify the full path of these files, you can use the “FindPaths-Mac” script to assist you.*

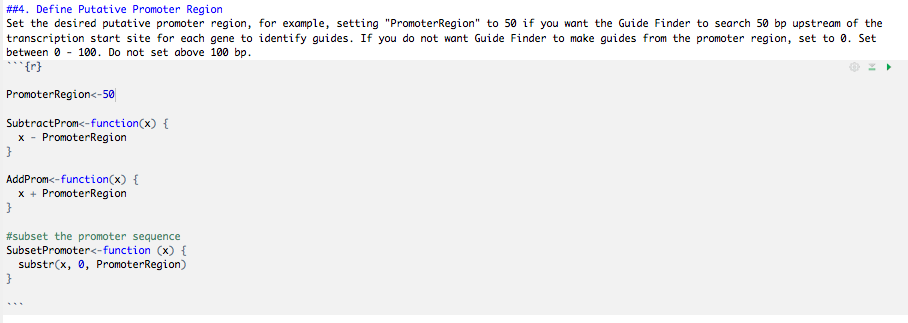
**B.** Set the name of the fasta file here, as shown in the blue box. Change to concatenatedfastafile.txt for this example. Make sure this file is held in the working directory.



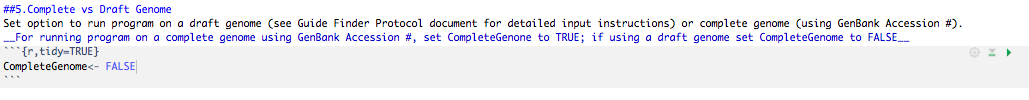
12.Input the FASTA file.   
Do this by including the name of your fasta file in the quotation marks within the readDNAStringSet function. This should be the same file as the last step. Make sure this file is held in the working directory. Run this chunk of code.



13. Define putative promoter region.   
The number you input here will determine how far upstream of a gene’s start site the Guide Finder program will look to identify guides in the promoter region. This can be set anywhere between 0-100. For this exercise, set PromoterRegion to 50 (as shown in the red box).

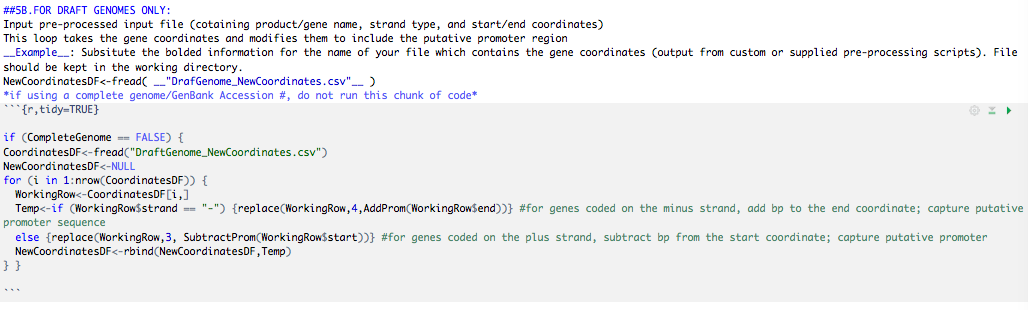


14 .Set CompleteGenome to FALSE.  
In this next chunk of code, set CompleteGenome to FALSE and run this chunk.

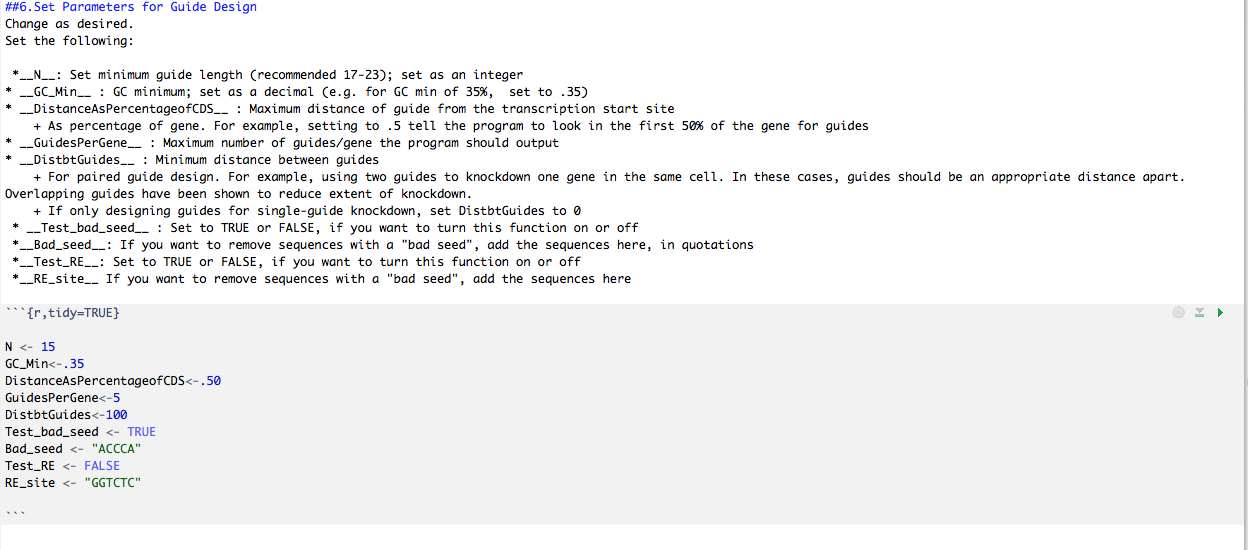


15. Skip the next chunk of code (labeled FOR COMPLETE GENOMES ONLY)  
Do not run this piece.

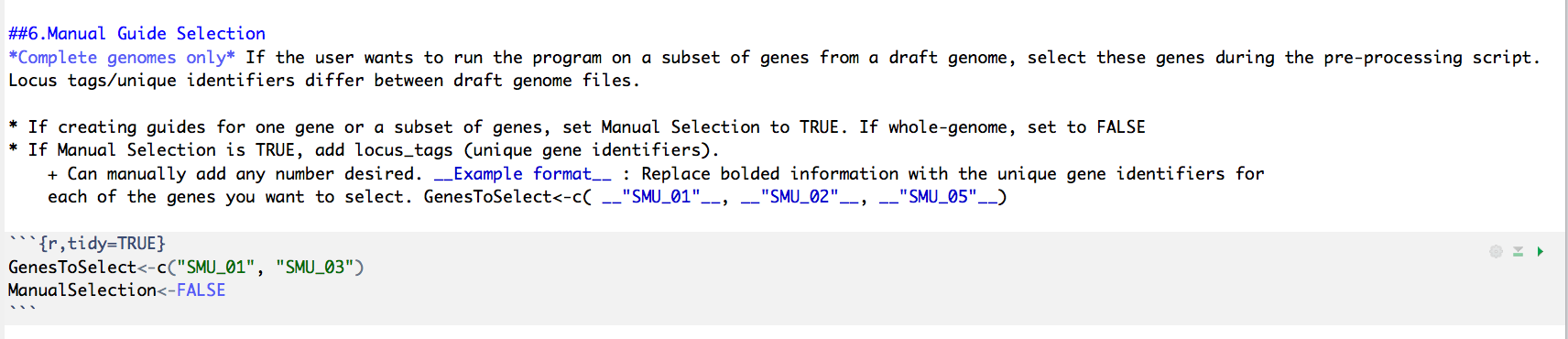
16. Input the DraftGenome\_NewCoordinates.csv file created from the pre-processing script   
Add the name of your output file from the pre-processing script (DraftGenome\_NewCoordinates.csv is default). Add in quotations to parentheses after fread(), as shown in the red box below.



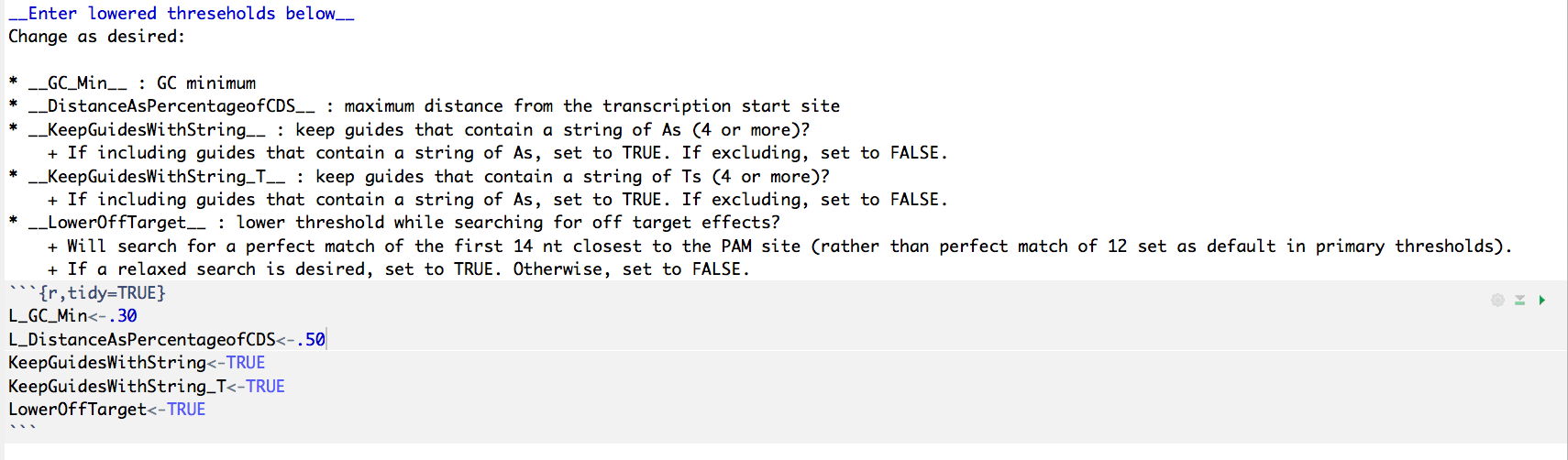
17. Set parameters for guide design   
For this example, we will set the following parameters for guide design. These are based on rational design parameters that have been successfully used to design guides in our lab. The meaning of each parameter is outlined in the R script, as shown below. Press the green arrow to run this chunk of code



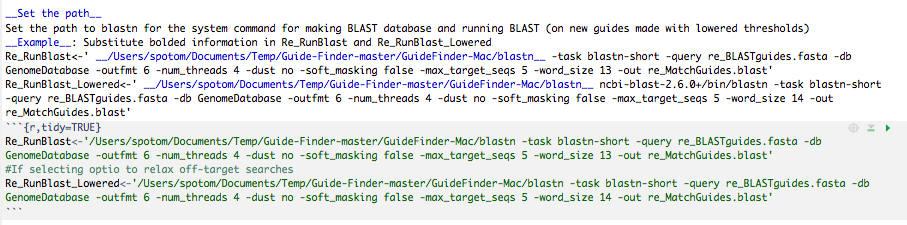
18. Set ManualGuideSelection to FALSE   
This chunk of code allows a user to only use the guide finder program to design guides for a select number of genes (if using a complete genome) The genes are subset using unique locus identification numbers. For this example, we want to run the program on the entire genome so we set ManualSelection to FALSE. There is no need to change or delete the gene identifier numbers in GenesToSelect if ManualSelection is set to FALSE. Press the green arrow to run this chunk of code



19. Set parameters for relaxed thresholds   
The guide finder program has the ability to identify genes that did not produce usable guides under the primary parameters and re-run these genes through the program again with relaxed thresholds in an attempt to recover guides. This allows more guides to be identified without compromising quality for the initial round of guide creation. In this example, set the relaxed parameters as show below. Descriptions of relaxed parameters are described in the script, as shown below. Press the green arrow to run this chunk of code



20. Set the path for BLAST   
Set the path to BLAST again in this chunk. This step is optional and only needs to be run if the user wants to find more guides for genes that did not produce any. In this example, we’ll run this chunk of code because we want to eventually perform this iterative step so we need to set the parameters for relaxed guide design (above, step 11) and set parameters for relaxed off-target searching (step 12). *Set this so the same file path* ***for blastn*** *as you did in Step 11.* Do this for ReRunBlast and ReRunBlast\_Lowered. In this step, there is no need to set the path for makeblastdb anywhere. No need to make any more modifications to the code except to set the file path (seen in red, below). Press the green arrow to run this chunk of code



21. Run the rest of the chunks of code!   
Run the remaining chunks of code. There is no need to edit anywhere below the **DO NOT EDIT BELOW THIS LINE** message. Simply press the green arrow on each chunk to run it. The code is commented so that you may follow along.

22. Look at your output files.   
The program will output files (saved to your working directory):   
1) **CompleteGuidesList:** a list of all possible guides, unfiltered, for reference  
2) **CompleteFilteredGuides:** list of all possible guides, filtered  
3) **TopHits\_Guides:** a list of guides—the desired max number/gene set by the user—closest to the transcription start site  
4) **Pairwise\_Guides:** a list of all possible guide pairs for each gene, for dual gene targeting  
5) **GenesWithoutGuides:** a list of genes that did not produce guides with primary thresholds  
6) **GuidesUsingLoweredThresholds:** a list of top hits guides for genes re-run with relaxed parameters  
7) **OverallGeneInfo:** information about the number of genes that created guides (top hits versus paired guides) from primary thresholds and the number of genes that created guides with the lowered thresholds.

Make sure that these files are the same as those provided in the Draft Genome Example folder to see if the program was run correctly.