Sequence Analysis Pipeline

SARS-CoV2 Virus

Jessie Bologna

5/18/20

NYU Bioinformatics — Problem Solving for Bioinformatics

INTRODUCTION :

ORF1ab Spike Protein:

- The novel virus SARS-CoV-2 which is the virus responsible for the Covid-19 outbreak is closely related to its predecessor SARS-CoV with both viruses sharing 79% sequence identity.
- Both viruses also use angiotensin converting enzyme-2 (ACE2) as its cellular receptor.
- However the SARS-CoV-2 spike-protein has nearly a 10-20-fold higher affinity for ACE2 than the corresponding spike protein of SARS-CoV which accounts for its higher virulence capacity
- (Kober et al., 2020)
- The ORF1ab spike protein mediates binding and entry into the host cells which allows the virus to infect the individual.
- This protein is often a good target for vaccine development, therefore it is important to understand any changes or mutations that affect this protein.
- (Wrap et al, 2020).

The following sequence analysis pipeline will outline and detail the steps of analyzing the genomic sequence of the SARS-CoV-2 OFR1ab spike protein.

Image source:Thinkstock

Analysis Pipeline:

Step 1: Mapping the SNPS (mutations)

Step 2: Parsing the files to obtain the coding sequences for each gene and strain in the whole genome

Step 3 : Multiple Sequence Alignment: Using BLAST+

- Compare the coding sequences from all the various strains of ORF1ab to the mutated reference genome file.

Step 4: Parse and interpret output files

Step 1: Mapping the SNPs

- Map mutations to reference genome NC_045512.fasta, by using the SNP table.
  - Input file: The virus's full genome fasta file(obtained from NCBI)
- Create a new genome file with the incorporated SNPs
  - Output file: mutated reference genome titled NC_045512_Mutated.fasta.
- *Perform this operation for each genome strain.*

```
#Get full genome sequence from file and turn into a list for parsing
import os
def Map_Mutations (input_file):
    with open (input_file, 'r') as input_file:
        #open fasta file to get the genome for reading
        #remove the first line
        header = input_file.readline()
        #turn the sequence into a list
        wild_type_list = [x for line in input_file.read().split('\n') for x in line]
        #print the first mutation location to check that it is the wildtype file
        print(wild_type_list[240])

    #Then open the SNP location file for reading to get the mutated genome from the locations
    with open ('countResult.txt', 'r') as input_file2:
        #remove the first line
        input_file2.readline()
        #open a file to write the final mutated genome
        out = open('NC_045512_mutated.txt', 'w')
        #loop through the snp location file and split at the :
        #then get insert the mutated snp at the index location in the wild type genome list
        for i in input_file2.readlines():
            coord, mutate = i.split(':')
            wild_type_list[int(coord)-1] = mutate[4]
        print(wild_type_list[240])
        mutation = "".join(wild_type_list)

        output_dir = os.getcwd()
        if not os.path.isdir(output_dir + '/Mutated_fasta/' ):
            os.mkdir(output_dir + '/Mutated_fasta/')
        with open(output_dir + '\\Mutated_fasta\\' + 'NC_045512_Mutated.fasta', "w") as f:
            f.write(str('>2019-nCoV|WH01|NC_045512|2020-01-05') + '\n'+ mutation)
        print( '\n'+ "Files Saved in Directory" + '\n')
```

Step 1: Mapping Snp's cont.

- ☑ *Map changes to reference genome NC_045512.fasta, by using the SNP table.*
- ☑ *Create a new genome file with the incorporated SNPs, called NC_045512_Mutated.fasta.*
- Perform the previous operation for each genome strain –
  - Here we create a new file containing the mutated coding sequences's for each gene by obtaining the genes coordinates from the virus's Genbank file

```
def Mutated_CDS_per_Genes(input_file):
    with open (input_file, 'r') as f:   #Just make sure to change this file to the mutated file
        data = f.read()
        #get the mutated cds for each gene
        #slide at the locations of each gene from the genbank file
        ORFlab = (data[265:21555])
        #write the sliced data to the outfile as the mutated cds for each gene
        out.write('Mutated CDS for Gene ORF1ab' + '\n' + str(ORFlab) + '\n')
        S = (data[21562:25384])
        out.write('Mutated CDS for Gene S' + '\n' + str(S) + '\n')
        ORF3a = (data[25392:26220])
        out.write('Mutated CDS for Gene ORF3a' + '\n' + str(ORF3a) + '\n')
        E = (data[26244:26472])
        out.write('Mutated CDS for Gene E' + '\n' + str(E) + '\n')
        M = (data[26522:27191])
        out.write('Mutated CDS for Gene M' + '\n' + str(M) + '\n')
        ORF6 = (data[27201:27387])
        out.write('Mutated CDS for Gene ORF6' + '\n' + str(ORF6) + '\n')
        ORF7a = (data[27393:27759])
        out.write('Mutated CDS for Gene ORF7a' + '\n' + str(ORF7a) + '\n')
        ORF7b = (data[27755:27887])
        out.write('Mutated CDS for Gene ORF7b' + '\n' + str(ORF7b) + '\n')
        ORF8 = (data[27893:28259])
        out.write('Mutated CDS for Gene ORF8' + '\n' + str(ORF8) + '\n')
        N = (data[28273:29533])
        out.write('Mutated CDS for Gene N' + '\n' + str(N) + '\n')
        ORF10 = (data[29557:29674])

        out.write('Mutated CDS for Gene ORF10' + '\n' + str(ORF10) + '\n')
        out.close()
```

Step 2: Parsing the Coding Sequences

- Parse the coding sequences for all genes and strains. (Your file should look like below image)
  - Input file: SARS-CoV-2-CdsFastaResults.fasta (provided by professor)
  - Output files: A separate file for each gene which will contain all of the genes various strains coding sequences.
- *Then create files of the coding sequences for each ORF1ab strain*
- *Translate each coding sequence for ORF1ab into its corresponding protein sequence.*

```
11 def parse_fasta_per_gene (input_file):
12     with open (input_file, 'r') as input_file:
13         output_dir = os.getcwd()
14         if not os.path.isdir(output_dir + '/SARS-CoV2_CDS_Outputfile/' ):
15             os.mkdir(output_dir + '/SARS-CoV2_CDS_Outputfile/')
16 
17         Genes = ['E','M','N', 'NS3','NS6','NS7a','NS7b','NS8','orf1a','orf1ab','S','ORF10','ORF3a','ORF6
18         list = []
19         #loop through each element to split
20         for element in input_file:
21             list.append(element)
22         line = ''.join(list)
23         x = line.split('>')
24         for i in Genes:
25             symbol = 'Gene Symbol:'+ i
26             filename = symbol + '-' + '.fasta'
27             filename = filename.replace('Gene Symbol:', 'Gene_and_Strains_')
28             #open outfile for writing
29             out_file = open(output_dir + '\\SARS-CoV2_CDS_Outputfile\\' + filename, 'w')
30             for line in x:
31                 if symbol in line:
32                     out_file.write(str('>') + line)
33             out_file.close()
```

```
1  >gb:MN988668:26244-26471|Organism:Severe acute respiratory syndrome coronavirus 2|Strain Name:2019-nCoV WHU01|Protein Name:envelope
   protein|Gene Symbol:E
2  ATGTACTCATTCGTTTCGGAAGAGACAGGTACGTTAATAGTTAATAGCGTACTTCTTTTTCTTGCTTTCG
3  TGGTATTCTTGCTAGTTACACTAGCCATCCTTACTGCGCTTCGATTGTGTGCGTACTGCTGCAATATTGT
4  TAACGTGAGTCTTGTAAAACCTTCTTTTTACGTTTACTCTCGTGTTAAAAATCTGAATTCTTCTAGAGTT
5  CCTGATCTTCTGGTCTAA
6
7  >gb:MN988669:26244-26471|Organism:Severe acute respiratory syndrome coronavirus 2|Strain Name:2019-nCoV WHU02|Protein Name:envelope
   protein|Gene Symbol:E
8  ATGTACTCATTCGTTTCGGAAGAGACAGGTACGTTAATAGTTAATAGCGTACTTCTTTTTCTTGCTTTCG
9  TGGTATTCTTGCTAGTTACACTAGCCATCCTTACTGCGCTTCGATTGTGTGCGTACTGCTGCAATATTGT
10 TAACGTGAGTCTTGTAAAACCTTCTTTTTACGTTTACTCTCGTGTTAAAAATCTGAATTCTTCTAGAGTT
11 CCTGATCTTCTGGTCTAA
12
```

Step 2: Parsing Coding Sequences for ORF1ab

- ☑ *Parse the coding sequence for ORF1ab from all genome strains.*
- Translate each coding sequence for ORF1ab into its corresponding protein sequences.
  - Input file: File created from previous step for the ORD1ab gene (this contains all the CDS's for each of ORF1ab's strains.
  - Output file: The output file will be very similar to the input file, except the coding sequences for each strain will now be translated into its amino acid sequence. We do this by using the below codon table in our code.
  - *Note: I used the Biopython modules: SeqIO, Alphabet, and IUPAC.

```
from Bio import SeqIO
from Bio.Alphabet import IUPAC
def parse_fasta_protein (input_file):
    input_file = SeqIO.parse("SARS-CoV-2-CdsFastaResults.fasta", 'fasta')
    #open outfile for writing
    with open('Gene_orflab_Protein.txt', 'w') as out_file;
        # DNA codon table
        protein = {"TTT" : "F", "CTT" : "L", "ATT" : "I", "GTT" : "V",
                   "TTC" : "F", "CTC" : "L", "ATC" : "I", "GTC" : "V",
                   "TTA" : "L", "CTA" : "L", "ATA" : "I", "GTA" : "V",
                   "TTG" : "L", "CTG" : "L", "ATG" : "M", "GTG" : "V",
                   "TCT" : "S", "CCT" : "P", "ACT" : "T", "GCT" : "A",
                   "TCC" : "S", "CCC" : "P", "ACC" : "T", "GCC" : "A",
                    "TCA" : "S", "CCA" : "P", "ACA" : "T", "GCA" : "A",
                   "TCG" : "S", "CCG" : "P", "ACG" : "T", "GCG" : "A",
                   "TAT" : "Y", "CAT" : "H", "AAT" : "N", "GAT" : "D",
                   "TAC" : "Y", "CAC" : "H", "AAC" : "N", "GAC" : "D",
                   "TAA" : "STOP", "CAA" : "Q", "AAA" : "K", "GAA" : "E",
                   "TAG" : "STOP", "CAG" : "Q", "AAG" : "K", "GAG" : "E",
                   "TGT" : "C", "CGT" : "R", "AGT" : "S", "GGT" : "G",
                   "TGC" : "C", "CGC" : "R", "AGC" : "S", "GGC" : "G",
                   "TGA" : "STOP", "CGA" : "R", "AGA" : "R", "GGA" : "G",
                   "TGG" : "W", "CGG" : "R", "AGG" : "R", "GGG" : "G"
                   }
        protein_sequence = ""
        for line in input_file:
            if 'orflab' in line.description:
                Seq1 = line.seq
        for i in range(0, len(Seq1)-(3+len(Seq1)%3), 3):
            if protein[Seq1[i:i+3]] == "STOP" :
                break
            protein_sequence += protein[Seq1[i:i+3]]
```

Step 3: Multiple Sequence Alignment

- Here I will run the multiple sequence alignment in BLAST+ on my local machine
  - To do this make sure to install BLAST+ on your computer by following the install instructions from NCBI
- First set up our BLAST databases by running *makeblastdb:*
  - One db with the reference genome file
  - One db with the mutated genome file that we created by mapping the SNP's

For the mutated ref database I used the mutated fasta file as the -in file

For the non-mutated ref database I used the reference genome file as the -in file

```
admin@admin-PC /cygdrive/c/users/admin/desktop/NCBI/blast-2.10.0+/db
$ makeblastdb -in NC_045512.fasta -dbtype nucl -title SARSCOV2_ref -out NC_045512.fasta_ref


Building a new DB, current time: 05/18/2020 19:00:23
New DB name:   C:\users\admin\desktop\NCBI\blast-2.10.0+\db\NC_045512.fasta_ref
New DB title:  SARSCOV2_ref
Sequence type: Nucleotide
Deleted existing Nucleotide BLAST database named C:\users\admin\desktop\NCBI\blast-2.10.0+\db\NC_045512.fasta_ref
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 1 sequences in 0.000860156 seconds.


admin@admin-PC /cygdrive/c/users/admin/desktop/NCBI/blast-2.10.0+/db
$ makeblastdb -in NC_045512_Mutated.fasta -dbtype nucl -title SARSCOV2_Mutated_ref -out NC_045512_Mutated.fasta_ref


Building a new DB, current time: 05/18/2020 19:15:19
New DB name:   C:\users\admin\desktop\NCBI\blast-2.10.0+\db\NC_045512_Mutated.fasta_ref
New DB title:  SARSCOV2_Mutated_ref
Sequence type: Nucleotide
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 1 sequences in 0.00121503 seconds.
```

STEP 3: MULTIPLE SEQUENCE ALIGNMENT

The next step is to run the sequence alignment :

- We are going to run a series of local alignments on the virus's key gene ORF1ab, which forms the spike protein structure that attaches itself to the ACE2 receptor enabling it to enters the host cell.
- We are going to do our alignment using *blastn*.
  - Run all of ORF1ab's strains against the mutated database (I also ran against the non-mutated db just as a reference)
  - Run based on different perc_identity: 95, 90, and 85 (also you can change various other parameters)
  - Note: I obtained two different file formats; the default format and an xml format. The xml format will be used in the next step to parse the results for easier analysis.

```
admin@admin-PC /cygdrive/c/users/admin/desktop/NCBI/blast-2.10.0+/db
$ blastn -query Gene_and_Strains_orf1ab.fasta -db NC_045512_Mutated.fasta_ref -perc_identity 90 -outfmt 5 -out MSA_Results_90

admin@admin-PC /cygdrive/c/users/admin/desktop/NCBI/blast-2.10.0+/db
$ blastn -query Gene_and_Strains_orf1ab.fasta -db NC_045512_Mutated.fasta_ref -perc_identity 90 -out MSA_Results_90_txt
```

Here is a snippet of what the summary results look like- located on the top and bottom of each alignment in the output file

```
Database: SARSCOV2_Mutated_ref
           1 sequences; 29,903 total letters



Query= gb:MN988668:265-21554|Organism:Severe acute respiratory syndrome
coronavirus 2|Strain Name:2019-nCoV WHU01|Protein Name:orf1ab
polyprotein|Gene Symbol:orf1ab

Length=21291

                                                                  Score     E
Sequences producing significant alignments:                      (Bits)  Value

2019-nCoV|WH01|NC_045512|2020-01-05                               37331   0.0


>2019-nCoV|WH01|NC_045512|2020-01-05
Length=29903

 Score = 37331 bits (20215),  Expect = 0.0
 Identities = 20925/21291 (98%), Gaps = 1/21291 (0%)
```

```
Lambda      K        H
   1.33    0.621     1.12

Gapped
Lambda      K        H
    1.28    0.460    0.850

Effective search space used: 635641293


  Database: SARSCOV2_Mutated_ref
    Posted date:  May 18, 2020  7:15 PM
  Number of letters in database: 29,903
  Number of sequences in database:  1



Matrix: blastn matrix 1 -2
Gap Penalties: Existence: 0, Extension: 2.5
```

STEP 4: PARSE THE RESULTS

- Parse the output file to check for variations of the of1ab gene per strains:
  - Use NCBIXML parser to parse the output file (make sure to format the output file as XML)
  - The output file (default format) produces an alignment for each strain - note on the below images that not all strains have the same scores
  - By parsing the file (XML format) we can check the alignments to see where the strains vary
  - (* Here I used the Biopython modules Blast and Bio.Blast NCBIXML)

```
from Bio import Blast
from Bio.Blast import NCBIXML
import os

def Parse_MSA_Results (input_file):
    #use SeqIO parser to parse the file
    with open(input_file, 'r') as f:
        results = NCBIXML.parse(f)
        out = open('MSA_Results.txt','w')
        for result in results:
            for alignment in result.alignments:
                for hsp in alignment.hsps:
                    if hsp.expect <= 0.00:
                        out.write('\n' + '****Alignment****' + '\n')
                        out.write ('sequence:' + str(alignment.title) + '\n')
                        out.write('ID:' + str(result.query_id) + "\n")
                        out.write('ID:' + str(result.query[:170])+ "\n")
                        out.write('score:' + str(hsp.score) + '\n')
                        out.write('gaps:' + str(hsp.gaps) + '\n')
                        out.write(hsp.query[0:75] + '...' + '\n')
                        out.write (hsp.match[0:75] + '...' + '\n')
                        out.write(hsp.sbjct[0:75] + '...' + '\n')
        out.close()
```

```
Query= gb:MN908947:266-21555|Organism:Severe acute respiratory syndrome
coronavirus 2|Strain Name:Wuhan-Hu-1|Protein Name:orf1ab
polyprotein|Gene Symbol:orf1ab

Length=21291

                                                                  Score     E
Sequences producing significant alignments:                      (Bits)  Value

2019-nCoV|WH01|NC_045512|2020-01-05                               37331   0.0


>2019-nCoV|WH01|NC_045512|2020-01-05
Length=29903

 Score = 37331 bits (20215),  Expect = 0.0
 Identities = 20925/21291 (98%), Gaps = 1/21291 (0%)
 Strand=Plus/Plus
```

```
Query= gb:MT246473:192-21478|Organism:Severe acute respiratory syndrome
coronavirus 2|Strain Name:SARS-CoV-2/human/USA/WA-UW216/2020|Protein
Name:orf1ab polyprotein|Gene Symbol:orf1ab

Length=21288

                                                                  Score     E
Sequences producing significant alignments:                      (Bits)  Value

2019-nCoV|WH01|NC_045512|2020-01-05                               37296   0.0


>2019-nCoV|WH01|NC_045512|2020-01-05
Length=29903

 Score = 37296 bits (20196),  Expect = 0.0
 Identities = 20916/21291 (98%), Gaps = 4/21291 (0%)
 Strand=Plus/Plus
```

Step 4: Parse the Results

- Variations between strains:
  - After parsing the output file it is easier to read and to see where the alignments vary
  - In the below sample of the parsed output file you can see that some strains have larger variations in scores and gaps.

```
****Alignment****
sequence:gnl|BL_ORD_ID|0 2019-nCoV|WH01|NC_045512|2020-01-05
ID:Query_1
ID:gb:MN988668:265-21554|Organism:Severe acute respiratory syndrome coronavirus 2|Strain Name:2019-nCoV WHU01|Protein Name:orf1ab
polyprotein|Gene Symbol:orf1ab
score:20215.0
gaps:1
ATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGTCCAACTCAGTTTGCCTGTTTTACAGGTTCGCGAC...
|||||||||||||||||||||||||||||||||||||||||||||||||| |||||||||||||||||||| |||||...
ATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGTCCAACTTAGTTTGCCTGTTTTACAGGTTTGCGAC...

****Alignment****
sequence:gnl|BL_ORD_ID|0 2019-nCoV|WH01|NC_045512|2020-01-05
ID:Query_2
ID:gb:MN988669:265-21554|Organism:Severe acute respiratory syndrome coronavirus 2|Strain Name:2019-nCoV WHU02|Protein Name:orf1ab
polyprotein|Gene Symbol:orf1ab
score:20215.0
gaps:1
ATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGTCCAACTCAGTTTGCCTGTTTTACAGGTTCGCGAC...
|||||||||||||||||||||||||||||||||||||||||||||||||| |||||||||||||||||||| |||||...
ATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGTCCAACTTAGTTTGCCTGTTTTACAGGTTTGCGAC...

****Alignment****
sequence:gnl|BL_ORD_ID|0 2019-nCoV|WH01|NC_045512|2020-01-05
ID:Query_3
ID:gb:LC521925:263-21528|Organism:Severe acute respiratory syndrome coronavirus 2|Strain Name:2019-nCoV/Japan/AI/I-004/2020|Protein
Name:orf1ab polyprotein|Gene Symbol:orf1a
score:20140.0
gaps:25
ATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGTCCAACTCAGTTTGCCTGTTTTACAGGTTCGCGAC...
|||||||||||||||||||||||||||||||||||||||||||||||||| |||||||||||||||||||| |||||...
ATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGTCCAACTTAGTTTGCCTGTTTTACAGGTTTGCGAC...
```

Step 4: Parse the Results

Variations between strains –

- You can change the parameters when parsing the output file to show lower scoring hits, or hits with high number of gaps etc.
- See below example of some strains from orf1ab that have a higher variation from the mutated genome file
- From these files you can note important changes and variations between strains in the ORF1ab region of the genome to reach conclusions about the viruses ability to bind to the receptor site –
  - Such changes can cause either a higher ability to bind and enter the cell causing it to be more virulent
  - Or some mutations can work against the virus by limiting its ability to bind to the receptor site making it more difficult to enter the cell.
- A further step is to run an alignment using the ORF1ab's translated sequence to look for specific amino acid changes which will help to analyze these variations and mutations.

```
from Bio import Blast
from Bio.Blast import NCBIXML
import os

def Parse_MSA_Results (input_file):
    #use SeqIO parser to parse the file
    with open(input_file, 'r') as f:
        results = NCBIXML.parse(f)
        out = open('MSA_Results_low_score.txt','w')
        for result in results:
            for alignment in result.alignments:
                for hsp in alignment.hsps:
                    if hsp.score <= 20150:
                        out.write('\n' + '****Alignment****' + '\n')
                        out.write ('sequence:' + str(alignment.title) + '\n')
                        out.write('ID:' + str(result.query_id) + "\n")
                        out.write('ID:' + str(result.query[:170])+ "\n")
                        out.write('score:' + str(hsp.score) + '\n')
                        out.write('gaps:' + str(hsp.gaps) + '\n')
                        out.write(hsp.query[0:75] + '...' + '\n')
                        out.write (hsp.match[0:75] + '...' + '\n')
                        out.write(hsp.sbjct[0:75] + '...' + '\n')
        out.close()
```

****Alignment****
sequence:gnl|BL_ORD_ID|0 2019-nCoV|WH01|NC_045512|2020-01-05
ID:gb:LC521925:263-21528|Organism:Severe acute respiratory syndrome coronavirus 2| Strain Name:2019-nCoV/Japan/AI/I-004/2020 |
score:20140.0
gaps:25
****Alignment****
sequence:gnl|BL_ORD_ID|0 2019-nCoV|WH01|NC_045512|2020-01-05
ID:gb:MT044258:266-21531|Organism:Severe acute respiratory syndrome coronavirus 2 |Strain Name:2019-nCoV/USA-CA6/2020
score:20134.0
gaps:25
****Alignment****
sequence:gnl|BL_ORD_ID|0 2019-nCoV|WH01|NC_045512|2020-01-05
ID:gb:MT159716:266-21540|Organism:Severe acute respiratory syndrome coronavirus 2| Strain Name:2019-nCoV/USA-CruiseA-18/2020 |
score:20166.0
gaps:16
****Alignment****
sequence:gnl|BL_ORD_ID|0 2019-nCoV|WH01|NC_045512|2020-01-05
ID:gb:MT039887:266-21555|Organism:Severe acute respiratory syndrome coronavirus 2| Strain Name:2019-nCoV/USA-WI1/2020 |
score:20208.0
gaps:4

References

- Altschul, S.F., Gish, W., Miller, W., Myers, E.W. & Lipman, D.J. (1990) "Basic local alignment search tool." J. Mol. Biol. 215:403-410. PubMed
- Camacho C., Coulouris G., Avagyan V., Ma N., Papadopoulos J., Bealer K., & Madden T.L. (2008) "BLAST+: architecture and applications." BMC Bioinformatics 10:421. PubMed
- Pachetti, M., Marini, B., Benedetti, F. *et al*.(2020) Emerging SARS-CoV-2 mutation hot spots include a novel RNA-dependent-RNA polymerase variant. *J Transl Med* **18,** 179 https://doi.org/10.1186/s12967-020-02344-6
- Wrapp, D., Wang, N., Corbett, K., Goldsmith, J., Hsieh, C., Abiona, O., Graham, B., McLellan, J. (March, 24 2020) Cryo-EM strucutre of the 2019-nCoV spike in the prefusion conformation. *Science367 (6483), 1260-1263.* DOI: 10.1126/science.abb2507originally published online February 19, 2020
- Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.