Viral genome notebook

Dec 03, 2020

Contents

A repo for keeping notebooks for viral genome analysis and data visualization

CHAPTER 1

List of example notebooks

1.1 Building consensus genome from raw fastq reads

- **Notebook version**: `v0.0.1`
- **Created by:** Dr. Hiren Ghosh, `Imperial BRC Genomics Facility`
- **Maintained by:** `Imperial BRC Genomics Facility`
- **Docker image:** imperialgenomicsfacility/viral-genome-analysis-notebooks
- **Github repository:** imperial-genomics-facility/viral-genome-notebook-image
- **Created on:** `2020-April-21 14:42`
- **Contact us:** Imperial BRC Genomics Facility
- **License:** Apache License 2.0

1.1.1 Configure notebook for run

[1]:
```
## Number of CPU to use
CPU_THREADS = 1

## Default value for runningthe notebook in binder
MEM_LIMIT_GB = 2
MEM_LIMIT_BYTES = MEM_LIMIT_GB * 1000000000

## Download raw fastq instead of SRA format, faster on binder
FETCH_RAW_FASTQ = True

## subsample reads to 1M to run in binder, set it to zero to disable
SUBSAMPLE_READ = 1000000
```

(continues on next page)

(continued from previous page)

```
## A toggle for running assembly on binder, set it to 0 to disable
RUN_ASSEMBLY = 1

## List of k-mers to use for de-novo assembly
ASSEMBLY_KMERS = '27,31'

## Accession id of the reference genome
REFERENCE_fasta = 'NC_045512.2'
```

1.1.2 Prepare sample list

[2]:
```
## we only have one sample in the list along with the fastq files for the sample

list_of_samples_data = \
  [{'sample_name':'SRR10971381',
    'fastq_files' : [
      '/tmp/SRR10971381_1.fastq.gz',
      '/tmp/SRR10971381_2.fastq.gz']}
  ]
```

1.1.3 Load required python libraries

[3]:
```
import os, requests
```

1.1.4 Fetch fastq files from SRA

[4]:
```
%%time
if FETCH_RAW_FASTQ:
  ## Download raw fastq from SRA (fast)
  !wget -O /tmp/SRR10971381_1.fastq.gz https://sra-pub-src-1.s3.amazonaws.com/
↪SRR10971381/WH_R1.fastq.gz.1
  !wget -O /tmp/SRR10971381_2.fastq.gz https://sra-pub-src-1.s3.amazonaws.com/
↪SRR10971381/WH_R2.fastq.gz.1
else:
  ## Download reads in SRA format and then convert it to fastq (slow)
  !wget -O /tmp/SRR10971381 https://sra-download.ncbi.nlm.nih.gov/traces/sra46/SRR/
↪010714/SRR10971381
  ## Convert SRA format data to fastq format
  !fastq-dump --split-files --gzip -outdir /tmp /tmp/SRR10971381
```

```
--2020-04-21 12:12:18--  https://sra-pub-src-1.s3.amazonaws.com/SRR10971381/WH_R1.
↪fastq.gz.1
Resolving sra-pub-src-1.s3.amazonaws.com (sra-pub-src-1.s3.amazonaws.com)... 52.216.
↪78.124
Connecting to sra-pub-src-1.s3.amazonaws.com (sra-pub-src-1.s3.amazonaws.com)|52.216.
↪78.124|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2739477612 (2.6G) [application/x-troff-man]
Saving to: '/tmp/SRR10971381_1.fastq.gz'
```

(continues on next page)

(continued from previous page)

```
/tmp/SRR10971381_1. 100%[===================>]   2.55G  29.4MB/s    in 80s

2020-04-21 12:13:38 (32.5 MB/s) - '/tmp/SRR10971381_1.fastq.gz' saved [2739477612/
↪2739477612]

--2020-04-21 12:13:41--  https://sra-pub-src-1.s3.amazonaws.com/SRR10971381/WH_R2.
↪fastq.gz.1
Resolving sra-pub-src-1.s3.amazonaws.com (sra-pub-src-1.s3.amazonaws.com)... 52.216.
↪147.140
Connecting to sra-pub-src-1.s3.amazonaws.com (sra-pub-src-1.s3.amazonaws.com)|52.216.
↪147.140|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2838458153 (2.6G) [application/x-troff-man]
Saving to: '/tmp/SRR10971381_2.fastq.gz'

/tmp/SRR10971381_2. 100%[===================>]   2.64G  30.1MB/s    in 82s

2020-04-21 12:15:03 (32.8 MB/s) - '/tmp/SRR10971381_2.fastq.gz' saved [2838458153/
↪2838458153]

CPU times: user 3.05 s, sys: 721 ms, total: 3.77 s
Wall time: 2min 46s
```

Sub-sample reads for binder run

This is an optional step. We are sub-sampling reads from the raw fastq files to the value specified in the variable **SUBSAMPLE_READ** to make it work in the Binder

[5]:
```
%%time
## following step may take some time to run

for entry in list_of_samples_data:
  if SUBSAMPLE_READ > 0:
    sample_name = entry.get('sample_name')
    fastq_files = entry.get('fastq_files')
    R1_fastq = fastq_files[0]
    R2_fastq = fastq_files[1]
    R1_sub_fastq = '/tmp/{0}_sub_1.fastq'.format(sample_name)
    R2_sub_fastq = '/tmp/{0}_sub_2.fastq'.format(sample_name)
    ## running seqtk to subsample files
    print('subsampling reads for sample {0} R1'.format(sample_name))
    !seqtk sample -2 -s100 $R1_fastq $SUBSAMPLE_READ  > $R1_sub_fastq
    print('subsampling reads for sample {0} R2'.format(sample_name))
    !seqtk sample -2 -s100 $R2_fastq $SUBSAMPLE_READ > $R2_sub_fastq
    entry.update({'subsample_fastq_files':[R1_sub_fastq,R2_sub_fastq]})
```

```
subsampling reads for sample SRR10971381 R1
subsampling reads for sample SRR10971381 R2
CPU times: user 9.85 s, sys: 1.7 s, total: 11.5 s
Wall time: 10min 29s
```

1.1.5 Check for viral DNA contamination using Fastv

Fetch required reference genomes and list of unique k-mers

[6]:
```
## fetch coronavirus genomes

!wget -O /tmp/SARS2_153_complete_genomes_20200329.fasta \
  https://storage.googleapis.com/sars-cov-2/SARS2_153_complete_genomes_20200329.fasta

## fetch unique kmers for coronavirus

!wget -O /tmp/SARS-CoV-2.kmer.fa \
  http://opengene.org/fastv/data/SARS-CoV-2.kmer.fa
```

```
--2020-04-21 12:25:34--  https://storage.googleapis.com/sars-cov-2/SARS2_153_complete_
↪genomes_20200329.fasta
Resolving storage.googleapis.com (storage.googleapis.com)... 108.177.112.128, 2607:
↪f8b0:4001:c07::80
Connecting to storage.googleapis.com (storage.googleapis.com)|108.177.112.128|:443...␣
↪connected.
HTTP request sent, awaiting response... 200 OK
Length: 4662317 (4.4M) [application/octet-stream]
Saving to: '/tmp/SARS2_153_complete_genomes_20200329.fasta'

/tmp/SARS2_153_comp 100%[===================>]   4.45M  --.-KB/s    in 0.07s

2020-04-21 12:25:34 (62.2 MB/s) - '/tmp/SARS2_153_complete_genomes_20200329.fasta'␣
↪saved [4662317/4662317]

--2020-04-21 12:25:35--  http://opengene.org/fastv/data/SARS-CoV-2.kmer.fa
Resolving opengene.org (opengene.org)... 47.90.42.109
Connecting to opengene.org (opengene.org)|47.90.42.109|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 7632 (7.5K) [application/octet-stream]
Saving to: '/tmp/SARS-CoV-2.kmer.fa'

/tmp/SARS-CoV-2.kme 100%[===================>]   7.45K  --.-KB/s    in 0s

2020-04-21 12:25:35 (886 MB/s) - '/tmp/SARS-CoV-2.kmer.fa' saved [7632/7632]
```

Prepare function for fastv run

[7]:
```
## run fastv function

def run_fastv(sample_name,ref_genome,ref_kmers,R1_fastq,R2_fastq,
              output_path='fastv_output'):
  '''
  A function for running fastv tool for a paired-end fastq data

  :param sample_name: Sample name
  :param ref_genome: Reference genome fasta file
  :param ref_kmers: Reference k-mers fasta file
  :param R1_fastq: Path for R1 fastq file
  :param R2_fastq: Path for R2 fastq file
  :param output_path: Output dir path, default fastv_output in current dir
  :returns: fastv_html_output,fastv_json_output,fastv_log_output
  '''
```

(continues on next page)

(continued from previous page)

```
  try:
    output_path = os.path.abspath(output_path)
    !mkdir -p $output_path
    print('running fastv for sample {0}'.format(sample_name))
    fastv_html_output = os.path.join(output_path,'{0}.fastv.html'.format(sample_name))
    fastv_json_output = os.path.join(output_path,'{0}.fastv.json'.format(sample_name))
    fastv_log_output = os.path.join(output_path,'{0}.fastv.log'.format(sample_name))
    !~/bin/fastv \
    -i $R1_fastq \
    -I $R2_fastq \
    -k $ref_kmers \
    -g $ref_genome \
    -h $fastv_html_output \
    -j $fastv_json_output \
    --thread $CPU_THREADS 2> $fastv_log_output
    return fastv_html_output,fastv_json_output,fastv_log_output
  except Exception as e:
    raise ValueError(
            'Failed to run fastv for sample {0}, error: {1}'.format(sample_name,e))
```

Run fastv for all samples

[8]:
```
%%time

for entry in list_of_samples_data:
  sample_name = entry.get('sample_name')
  if SUBSAMPLE_READ > 0:
    fastq_files = entry.get('subsample_fastq_files')
  else:
    fastq_files = entry.get('fastq_files')

  R1_fastq = fastq_files[0]
  R2_fastq = fastq_files[1]
  fastv_html_output,fastv_json_output,fastv_log_output =\
    run_fastv(
      sample_name=sample_name,
      ref_genome='/tmp/SARS2_153_complete_genomes_20200329.fasta',
      ref_kmers='/tmp/SARS-CoV-2.kmer.fa',
      R1_fastq=R1_fastq,
      R2_fastq=R2_fastq)
  entry.update(
    {'fastv_files':[fastv_html_output,fastv_json_output,fastv_log_output]})
```

```
running fastv for sample SRR10971381
CPU times: user 2 s, sys: 334 ms, total: 2.33 s
Wall time: 2min 19s
```

[9]:
```
## now we have the list of output files appended in the sample list

list_of_samples_data
```

[9]:
```
[{'sample_name': 'SRR10971381',
  'fastq_files': ['/tmp/SRR10971381_1.fastq.gz',
```

(continues on next page)

(continued from previous page)

```
   '/tmp/SRR10971381_2.fastq.gz'],
  'subsample_fastq_files': ['/tmp/SRR10971381_sub_1.fastq',
   '/tmp/SRR10971381_sub_2.fastq'],
  'fastv_files': ['/home/vmuser/examples/fastv_output/SRR10971381.fastv.html',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.json',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.log']}]
```

1.1.6 De-novo assembly of viral genome using Megahit

Prepare function for Megahit assembly run

[10]:
```
def run_megahit_assembly(sample_name,R1_fastq,R2_fastq,output_path='megahit_output'):
  '''
  A function for running megahit de-novo assembly for a paired-end fastq data

  :param sample_name: Sample name
  :param R1_fastq: Path for R1 fastq file
  :param R2_fastq: Path for R2 fastq file
  :param output_path: Output dir path, default megahit_output in current dir
  :returns: megahit_assembly,fastg_output
  '''
  try:
    output_path = os.path.abspath(output_path)
    !mkdir -p $output_path
    output_dir = os.path.join(output_path,'megahit_assembly_{0}'.format(sample_name))
    print('running de-novo assembly using megahit for sample {0}'.format(sample_name))
    !megahit \
      -1 $R1_fastq \
      -2 $R2_fastq \
      -o $output_dir \
      --k-list $ASSEMBLY_KMERS \
      --num-cpu-threads $CPU_THREADS \
      --memory $MEM_LIMIT_BYTES \
      --tmp-dir /tmp
    max_kmer = ASSEMBLY_KMERS.split(',')[-1]
    fastg_input = \
      os.path.join(
        output_dir,
        'intermediate_contigs',
        'k{0}.contigs.fa'.format(max_kmer))
    fastg_output = os.path.join(output_dir,'{0}_k{1}.fastg'.format(sample_name,max_
↪kmer))
    print('converting de-novo assembly to fastg for sample {0}'.format(sample_name))
    !megahit_toolkit contig2fastg $max_kmer $fastg_input > $fastg_output
    return output_dir,fastg_output
  except Exception as e:
    raise ValueError(
            'Failed to run fastv for sample {0}, error: {1}'.format(sample_name,e))
```

Run de-novo assembly for all the samples

[11]:
```
%%time

## following step may take upto 20 min

if RUN_ASSEMBLY > 0:
  for entry in list_of_samples_data:
    sample_name = entry.get('sample_name')
    if SUBSAMPLE_READ > 0:
      fastq_files = entry.get('subsample_fastq_files')
    else:
      fastq_files = entry.get('fastq_files')

    R1_fastq = fastq_files[0]
    R2_fastq = fastq_files[1]
    megahit_output_dir,fastg_output = \
      run_megahit_assembly(
        sample_name=sample_name,
        R1_fastq=R1_fastq,
        R2_fastq=R2_fastq)
    entry.update({'megahit_output_dir':megahit_output_dir,'megahit_fastg':fastg_
↪output})
```

```
running de-novo assembly using megahit for sample SRR10971381
2020-04-21 12:27:56 - MEGAHIT v1.2.9
2020-04-21 12:27:56 - Using megahit_core with POPCNT and BMI2 support
2020-04-21 12:27:56 - Convert reads to binary library
2020-04-21 12:28:00 - b'INFO  sequence/io/sequence_lib.cpp  :   77 - Lib 0 (/tmp/
↪SRR10971381_sub_1.fastq,/tmp/SRR10971381_sub_2.fastq): pe, 2000000 reads, 151 max␣
↪length'
2020-04-21 12:28:00 - b'INFO  utils/utils.h                :  152 - Real: 3.
↪4744\tuser: 2.2256\tsys: 1.1238\tmaxrss: 164380'
2020-04-21 12:28:00 - Start assembly. Number of CPU threads 1
2020-04-21 12:28:00 - k list: 27,31
2020-04-21 12:28:00 - Memory used: 2000000000
2020-04-21 12:28:00 - Extract solid (k+1)-mers for k = 27
2020-04-21 12:30:02 - Build graph for k = 27
2020-04-21 12:31:32 - Assemble contigs from SdBG for k = 27
2020-04-21 12:37:34 - Local assembly for k = 27
2020-04-21 12:37:55 - Extract iterative edges from k = 27 to 31
2020-04-21 12:38:22 - Build graph for k = 31
2020-04-21 12:38:49 - Assemble contigs from SdBG for k = 31
2020-04-21 12:42:07 - Merging to output final contigs
2020-04-21 12:42:07 - 9056 contigs, total 2427140 bp, min 200 bp, max 8614 bp, avg␣
↪268 bp, N50 253 bp
2020-04-21 12:42:08 - ALL DONE. Time elapsed: 851.511862 seconds
converting de-novo assembly to fastg for sample SRR10971381
CPU times: user 13.2 s, sys: 2 s, total: 15.3 s
Wall time: 14min 17s
```

[12]:
```
## now we have the list of assembly output files present in the sample list

list_of_samples_data
```

[12]:
```
[{'sample_name': 'SRR10971381',
  'fastq_files': ['/tmp/SRR10971381_1.fastq.gz',
   '/tmp/SRR10971381_2.fastq.gz'],
  'subsample_fastq_files': ['/tmp/SRR10971381_sub_1.fastq',
   '/tmp/SRR10971381_sub_2.fastq'],
```

(continues on next page)

(continued from previous page)

```
  'fastv_files': ['/home/vmuser/examples/fastv_output/SRR10971381.fastv.html',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.json',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.log'],
  'megahit_output_dir': '/home/vmuser/examples/megahit_output/megahit_assembly_
↪SRR10971381',
  'megahit_fastg': '/home/vmuser/examples/megahit_output/megahit_assembly_SRR10971381/
↪SRR10971381_k31.fastg'}]
```

1.1.7 Map raw reads on reference genome

Prepare function for reference genome fetching

[13]:
```
def fetch_genome_fasta_from_ncbi(refseq_id,output_path='.',file_format='fasta'):
  '''
  A function for fetching the genome fasta sequences from NCBI

  :param refseq_id: NCBI genome id
  :param output_path: Path to dump genome files, default '.'
  :param file_format: Output file format, default fasta, supported formats are 'fasta
↪' and 'gb'
  :returns: output_file
  '''
  try:
    output_path = os.path.abspath(output_path)
    !mkdir -p $output_path
    url = \
      'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nucleotide&id={0}&
↪rettype={1}'.\
        format(refseq_id,file_format)
    r = requests.get(url)
    if r.status_code != 200:
      raise ValueError('Failed to download file for {0}, http status code {1}'.
↪format(refseq_id,r.status_code))
    data = r.content.decode('utf-8')
    output_file = \
      os.path.join(
        os.path.abspath(output_path),
        '{0}.{1}'.format(refseq_id,file_format))
    with open(output_file,'w') as fp:
      fp.write(data)
    print('Downloaded genome seq for {0}'.format(refseq_id))
    return output_file
  except Exception as e:
    raise ValueError('Failed to download data for {0} from NCBI, error: {1}'.
↪format(refseq_id,e))
```

[14]:
```
reference_fastq = fetch_genome_fasta_from_ncbi(REFERENCE_fasta,output_path='ref_genome
↪')
reference_fastq
```

```
Downloaded genome seq for NC_045512.2
```

[14]:
```
'/home/vmuser/examples/ref_genome/NC_045512.2.fasta'
```

Build Bowtie2 index for reference genome

[15]:
```
## create reference index dir

!mkdir -p bowtie2_ref

bowtie2_ref = os.path.abspath('bowtie2_ref/NC_045512.2')
## Build Bowtie2 index for reference genome

!bowtie2-build \
  $reference_fastq \
  $bowtie2_ref > bowtie2_build.log
```

```
Building a SMALL index
```

Prepare function for bowtie2 mapping

[16]:
```
def run_bowtie2_mapping(sample_name,bowtie2_index,R1_fastq,R2_fastq,output_path=
↪'bowtie2_output'):
  '''
  A function for running bowtie2 mapping for a paired-end fastq data

  :param sample_name: Sample name
  :param bowtie2_index: Bowtie index path
  :param R1_fastq: Path for R1 fastq file
  :param R2_fastq: Path for R2 fastq file
  :param output_path: Output dir path, default bowtie2_output in current dir
  :returns: bowtie2 alignment in sam
  '''
  try:
    output_path = os.path.abspath(output_path)
    !mkdir -p $output_path
    output_sam = os.path.join(output_path,'alignment_{0}.sam'.format(sample_name))
    !bowtie2 \
      -x $bowtie2_index \
      --very-fast  \
      -1 $R1_fastq \
      -2 $R2_fastq \
      --threads $CPU_THREADS \
      -S  $output_sam
    return output_sam
  except Exception as e:
    raise ValueError(
            'Failed to run bowtie2 for sample {0}, error: {1}'.format(sample_name,e))
```

Run bowtie2 mapping for all the samples

[17]:
```
%%time

## following step may take upto 30 min (per sample)
for entry in list_of_samples_data:
  sample_name = entry.get('sample_name')
  fastq_files = entry.get('fastq_files')
  R1_fastq = fastq_files[0]
```

(continues on next page)

(continued from previous page)

```
  R2_fastq = fastq_files[1]
  output_sam = \
    run_bowtie2_mapping(
      sample_name=sample_name,
      bowtie2_index=bowtie2_ref,
      R1_fastq=R1_fastq,
      R2_fastq=R2_fastq)
  entry.update({'bowtie2_sam':output_sam})
```

```
28282964 reads; of these:
  28282964 (100.00%) were paired; of these:
    28224324 (99.79%) aligned concordantly 0 times
    58640 (0.21%) aligned concordantly exactly 1 time
    0 (0.00%) aligned concordantly >1 times
    ----
    28224324 pairs aligned concordantly 0 times; of these:
      1870 (0.01%) aligned discordantly 1 time
    ----
    28222454 pairs aligned 0 times concordantly or discordantly; of these:
      56444908 mates make up the pairs; of these:
        56443948 (100.00%) aligned 0 times
        960 (0.00%) aligned exactly 1 time
        0 (0.00%) aligned >1 times
0.22% overall alignment rate
CPU times: user 18 s, sys: 2.95 s, total: 21 s
Wall time: 21min 33s
```

[18]:
```
## now we have the list of bowtie2 output files present in the sample list

list_of_samples_data
```

[18]:
```
[{'sample_name': 'SRR10971381',
  'fastq_files': ['/tmp/SRR10971381_1.fastq.gz',
   '/tmp/SRR10971381_2.fastq.gz'],
  'subsample_fastq_files': ['/tmp/SRR10971381_sub_1.fastq',
   '/tmp/SRR10971381_sub_2.fastq'],
  'fastv_files': ['/home/vmuser/examples/fastv_output/SRR10971381.fastv.html',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.json',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.log'],
  'megahit_output_dir': '/home/vmuser/examples/megahit_output/megahit_assembly_
↪SRR10971381',
  'megahit_fastg': '/home/vmuser/examples/megahit_output/megahit_assembly_SRR10971381/
↪SRR10971381_k31.fastg',
  'bowtie2_sam': '/home/vmuser/examples/bowtie2_output/alignment_SRR10971381.sam'}]
```

1.1.8 Consensus genome building from mapped reads

Prepare function for consensus fasta building

[19]:
```
def aln_to_consensus_fasta(sample_name,sam_file,reference_fasta,output_path='samtools_
↪dir'):
  '''
  A function for aligned sam file to consensus fasta generation

  :param sample_name: Sample name
```

(continues on next page)

(continued from previous page)

```
  :param sam_file: sam format alignment file
  :param reference_fasta: reference fasta file
  :param output_path: Output dir path, default samtools_dir
  :returns: sorted_bam_file,flagstat_file,bcftools_call,consensus_fasta
  '''
  try:
    output_path = os.path.abspath(output_path)
    !mkdir -p $output_path
    bam_file = os.path.join('/tmp','{0}_raw.bam'.format(sample_name))
    sorted_bam_file = os.path.join(output_path,'{0}_sorted.bam'.format(sample_name))
    flagstat_file = os.path.join(output_path,'{0}_sorted.flagstat'.format(sample_
↪name))
    bcftools_call = os.path.join(output_path,'{0}_calls.vcf.gz'.format(sample_name))
    consensus_fasta = os.path.join(output_path,'{0}_consensus.fasta'.format(sample_
↪name))
    !samtools view -q 5 -bo $bam_file $sam_file
    !samtools sort $bam_file > $sorted_bam_file
    !samtools index $sorted_bam_file
    !samtools flagstat $sorted_bam_file > $flagstat_file
    !bcftools mpileup -f $reference_fasta $sorted_bam_file | bcftools call --ploidy 1␣
↪ -mv -Oz -o $bcftools_call
    !bcftools index $bcftools_call
    !cat $reference_fasta | bcftools consensus $bcftools_call > $consensus_fasta
    return sorted_bam_file,flagstat_file,bcftools_call,consensus_fasta
  except Exception as e:
    raise ValueError(
            'Failed to run consensus fasta generation for sample {0}, error: {1}'.
↪format(sample_name,e))
```

Run consensus fasta building for all samples

[20]:
```
%%time

## following step may take upto 30 min (per sample)
for entry in list_of_samples_data:
  sample_name = entry.get('sample_name')
  bowtie2_sam = entry.get('bowtie2_sam')
  sorted_bam_file,flagstat_file,bcftools_call,consensus_fasta = \
    aln_to_consensus_fasta(
      sample_name=sample_name,
      sam_file=bowtie2_sam,
      reference_fasta='/home/vmuser/examples/ref_genome/NC_045512.2.fasta')
  entry.update({
    'bowtie2_sorted_aln':sorted_bam_file,
    'flagstat_file':flagstat_file,
    'bcftools_call':bcftools_call,
    'consensus_fasta':consensus_fasta})
```

```
[mpileup] 1 samples in 1 input files
[mpileup] maximum number of reads per input file set to -d 250
Note: the --sample option not given, applying all records regardless of the genotype
The site NC_045512.2:3 overlaps with another variant, skipping...
The site NC_045512.2:4 overlaps with another variant, skipping...
Applied 1 variants
CPU times: user 6.94 s, sys: 1.19 s, total: 8.13 s
Wall time: 8min 14s
```

[21]:
```
## now we have the list of consensus fasta output files present in the sample list

list_of_samples_data
```

[21]:
```
[{'sample_name': 'SRR10971381',
  'fastq_files': ['/tmp/SRR10971381_1.fastq.gz',
   '/tmp/SRR10971381_2.fastq.gz'],
  'subsample_fastq_files': ['/tmp/SRR10971381_sub_1.fastq',
   '/tmp/SRR10971381_sub_2.fastq'],
  'fastv_files': ['/home/vmuser/examples/fastv_output/SRR10971381.fastv.html',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.json',
   '/home/vmuser/examples/fastv_output/SRR10971381.fastv.log'],
  'megahit_output_dir': '/home/vmuser/examples/megahit_output/megahit_assembly_
↪SRR10971381',
  'megahit_fastg': '/home/vmuser/examples/megahit_output/megahit_assembly_SRR10971381/
↪SRR10971381_k31.fastg',
  'bowtie2_sam': '/home/vmuser/examples/bowtie2_output/alignment_SRR10971381.sam',
  'bowtie2_sorted_aln': '/home/vmuser/examples/samtools_dir/SRR10971381_sorted.bam',
  'flagstat_file': '/home/vmuser/examples/samtools_dir/SRR10971381_sorted.flagstat',
  'bcftools_call': '/home/vmuser/examples/samtools_dir/SRR10971381_calls.vcf.gz',
  'consensus_fasta': '/home/vmuser/examples/samtools_dir/SRR10971381_consensus.fasta'}
↪]
```

[ ]:

1.2 Multiple sequence alignment and tree building

- **Notebook version**: `v0.0.1`
- **Created by:** Dr. Hiren Ghosh, `Imperial BRC Genomics Facility`
- **Maintained by:** `Imperial BRC Genomics Facility`
- **Docker image:** imperialgenomicsfacility/viral-genome-analysis-notebooks
- **Github repository:** imperial-genomics-facility/viral-genome-notebook-image
- **Created on:** `2020-April-21 14:42`
- **Contact us:** Imperial BRC Genomics Facility
- **License:** Apache License 2.0

1.2.1 Data fetching

List of genomes:

- **NC_045512.2**: Severe acute respiratory syndrome coronavirus 2 isolate Wuhan-Hu-1, complete genome
- **MN988668.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV WHU01, complete genome
- **MN938384.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV_HKU-SZ-002a_2020, complete genome
- **MN975262.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV_HKU-SZ-005b_2020, complete genome

- **MN985325.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV/USA-WA1/2020, complete genome
- **MN988713.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV/USA-IL1/2020, complete genome
- **MN994467.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV/USA-CA1/2020, complete genome
- **MN994468.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV/USA-CA2/2020, complete genome
- **MN997409.1**: Severe acute respiratory syndrome coronavirus 2 isolate 2019-nCoV/USA-AZ1/2020, complete genome

[1]:
```
import os
import requests
import json
import time
from ete3 import Tree, TreeStyle,SeqMotifFace
os.environ['QT_QPA_PLATFORM']='offscreen'
```

[2]:
```
def fetch_genome_fasta_from_ncbi(refseq_id,output_path='.',file_format='fasta'):
  '''
  A function for fetching the genome fasta sequences from NCBI

  :param refseq_id: NCBI genome id
  :param output_path: Path to dump genome files, default '.'
  :param file_format: Output fileformat, default fasta, supported formats are 'fasta'␣
↪and 'gb'
  :return: output_file
  '''
  try:
    url = \
      'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nucleotide&id={0}&
↪rettype={1}'.\
        format(refseq_id,file_format)
    r = requests.get(url)
    if r.status_code != 200:
      raise ValueError(
              'Failed to download data from NCBI for id {0}, error code: {1}'.\
                format(refseq_id,r.status_code))
    fasta_data = r.content.decode('utf-8')
    output_file = \
      os.path.join(
        os.path.abspath(output_path),
        '{0}.{1}'.format(refseq_id,file_format))
    with open(output_file,'w') as fp:
      fp.write(fasta_data)
    print('Downloaded genome seq for {0}'.format(refseq_id))
    return output_file
  except Exception as e:
    print('Failed to download data for {0} from NCBI, error: {1}'.format(refseq_id,e))
```

[3]:
```
genome_list = [
    'MN988668.1',
    'NC_045512.2',
    'MN938384.1',
```

(continues on next page)

(continued from previous page)

```
    'MN975262.1',
    'MN985325.1',
    'MN988713.1',
    'MN994467.1',
    'MN994468.1',
    'MN997409.1'
]
```

[5]:
```
fasta_list = []
for id in genome_list:
  fasta_list.\
    append(
      fetch_genome_fasta_from_ncbi(
        refseq_id=id,
        output_path='/tmp'))
  time.sleep(2)
fasta_list
```

```
Downloaded genome seq for MN988668.1
Downloaded genome seq for NC_045512.2
Downloaded genome seq for MN938384.1
Downloaded genome seq for MN975262.1
Downloaded genome seq for MN985325.1
Downloaded genome seq for MN988713.1
Downloaded genome seq for MN994467.1
Downloaded genome seq for MN994468.1
Downloaded genome seq for MN997409.1
```

[5]:
```
['/tmp/MN988668.1.fasta',
 '/tmp/NC_045512.2.fasta',
 '/tmp/MN938384.1.fasta',
 '/tmp/MN975262.1.fasta',
 '/tmp/MN985325.1.fasta',
 '/tmp/MN988713.1.fasta',
 '/tmp/MN994467.1.fasta',
 '/tmp/MN994468.1.fasta',
 '/tmp/MN997409.1.fasta']
```

1.2.2 Multiple sequence alignment with Muscle

Merge all fasta files to a single filne and create multiple sequence alignment

[6]:
```
MERGED_FASTA = '/tmp/corona_merged.fasta'
!rm -f $MERGED_FASTA
for f in fasta_list:
  !cat $f >> $MERGED_FASTA

## count the genome sequences in the merged fasta file
!grep '>' $MERGED_FASTA|wc -l
```

```
9
```

[9]:
```
%%time

MUSCLE_MSA = '/tmp/corona_merged.afa'
!muscle -in $MERGED_FASTA -maxiters 2 -out $MUSCLE_MSA
```

```
MUSCLE v3.8.1551 by Robert C. Edgar

http://www.drive5.com/muscle
This software is donated to the public domain.
Please cite: Edgar, R.C. Nucleic Acids Res 32(5), 1792-97.

corona_merged 9 seqs, lengths min 29838, max 29903, avg 29880
00:00:00     20 MB(2%)  Iter   1  100.00%  K-mer dist pass 1
00:00:00     20 MB(2%)  Iter   1  100.00%  K-mer dist pass 2
00:02:27  1126 MB(100%)  Iter   1  100.00%  Align node
00:02:27  1126 MB(100%)  Iter   1  100.00%  Root alignment
00:03:58  1126 MB(100%)  Iter   2  100.00%  Refine tree
00:03:58  1126 MB(100%)  Iter   2  100.00%  Root alignment
00:03:58  1126 MB(100%)  Iter   2  100.00%  Root alignment
CPU times: user 5.8 s, sys: 1.41 s, total: 7.21 s
Wall time: 3min 58s
```

1.2.3 Tree building using FastTree

Build phylogenetic tree with FastTree

[10]:
```
%%time

TREE_FILE = '/tmp/corona_merged.tree'
!FastTree -nt -gtr < $MUSCLE_MSA > $TREE_FILE
```

```
FastTree Version 2.1.10 Double precision (No SSE3)
Alignment: standard input
Nucleotide distances: Jukes-Cantor Joins: balanced Support: SH-like 1000
Search: Normal +NNI +SPR (2 rounds range 10) +ML-NNI opt-each=1
TopHits: 1.00*sqrtN close=default refresh=0.80
ML Model: Generalized Time-Reversible, CAT approximation with 20 rate categories
Ignored unknown character S (seen 1 times)
Ignored unknown character W (seen 1 times)
Ignored unknown character Y (seen 6 times)
Initial topology in 0.05 seconds
Refining topology: 13 rounds ME-NNIs, 2 rounds ME-SPRs, 6 rounds ML-NNIs
Total branch-length 0.001 after 0.50 sec, 1 of 7 splits

WARNING! This alignment consists of closely-related and very-long sequences.
WARNING! FastTree (or other standard maximum-likelihood tools)
may not be appropriate for aligments of very closely-related sequences
like this one, as FastTree does not account for recombination or gene conversion

ML-NNI round 1: LogLk = -41615.807 NNIs 2 max delta 0.00 Time 0.85
GTR Frequencies: 0.2990 0.1837 0.1962 0.3211ep 12 of 12
GTR rates(ac ag at cg ct gt) 0.0721 0.5133 0.3246 0.8927 4.7540 1.0000
Switched to using 20 rate categories (CAT approximation)19 of 20
Rate categories were divided by 0.623 so that average rate = 1.0
CAT-based log-likelihoods may not be comparable across runs
Use -gamma for approximate but comparable Gamma(20) log-likelihoods
ML-NNI round 2: LogLk = -40709.584 NNIs 2 max delta 0.00 Time 5.19
Turning off heuristics for final round of ML NNIs (converged)
ML-NNI round 3: LogLk = -40709.584 NNIs 0 max delta 0.00 Time 5.75 (final)
Optimize all lengths: LogLk = -40709.584 Time 5.95
```

(continues on next page)

(continued from previous page)

```
Total time: 7.27 seconds Unique: 9/9 Bad splits: 0/6
CPU times: user 210 ms, sys: 42 ms, total: 252 ms
Wall time: 7.95 s
```

1.2.4 Plotting trees

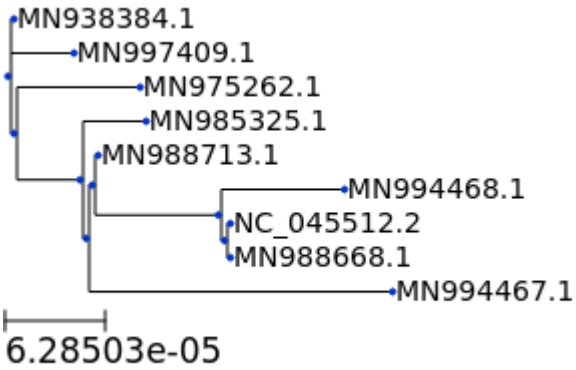
[11]:
```
t = Tree(TREE_FILE)
```

[12]:
```
## ascii tree
print(t)
```

```
   /-MN938384.1
  |
  |--MN997409.1
--|
  |   /-MN975262.1
  |  |
   \-|   /-MN985325.1
     |  |
     |  |      /-MN988713.1
      \-|   /-|
        |  |  |   /-MN994468.1
        |  |   \-|
         \-|     |   /-NC_045512.2
           |      \-|
           |         \-MN988668.1
           |
            \-MN994467.1
```

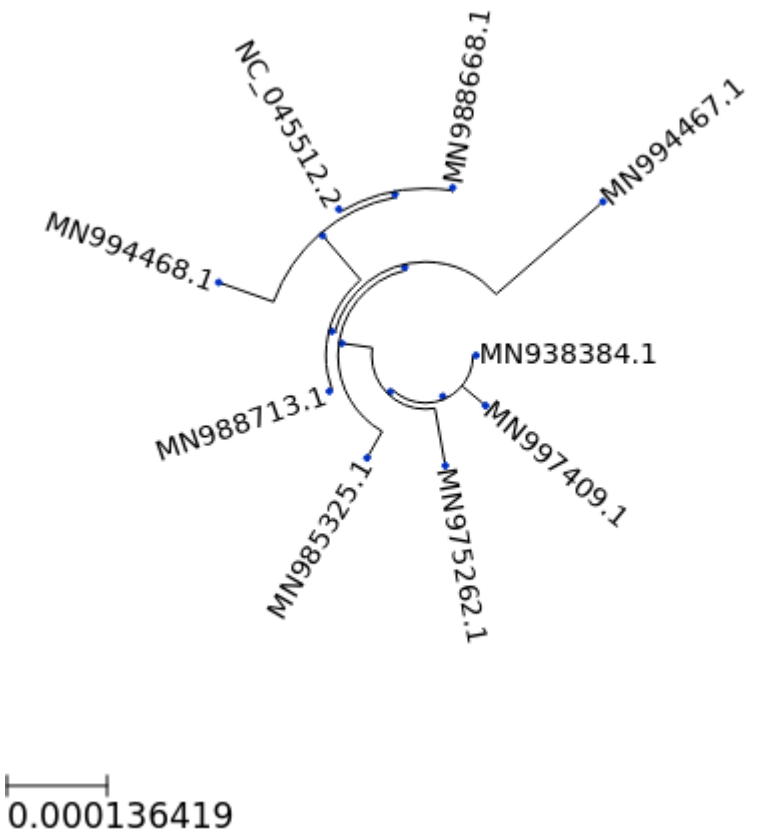
[13]:
```
## plain tree
t.render("%%inline")
```

[13]:



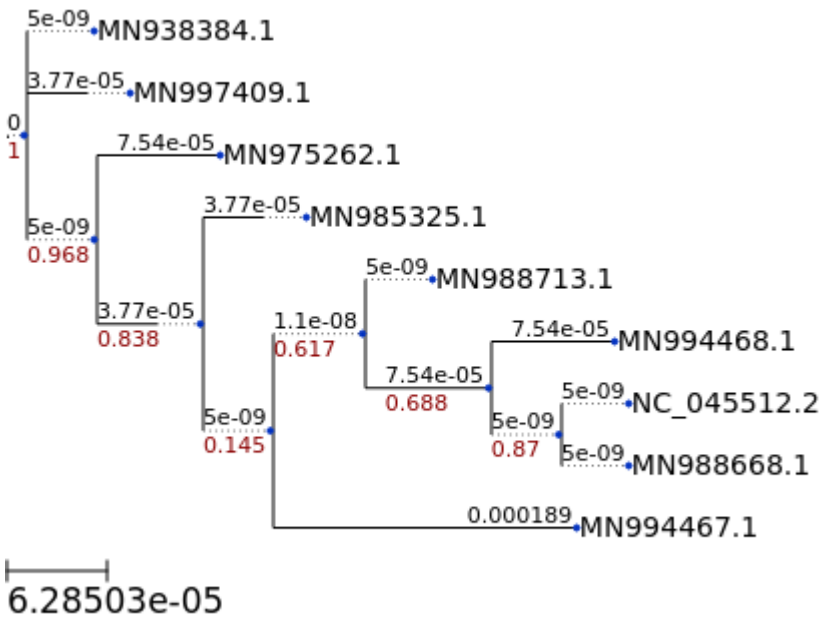
[14]:
```
## circular tree
ts = TreeStyle()
ts.mode = "c"
t.render("%%inline",  tree_style=ts)
```

[14]:



[15]:
```
## tree with branch length
ts = TreeStyle()
ts.show_leaf_name = True
ts.show_branch_length = True
ts.show_branch_support = True
t.render("%%inline", tree_style=ts)
```

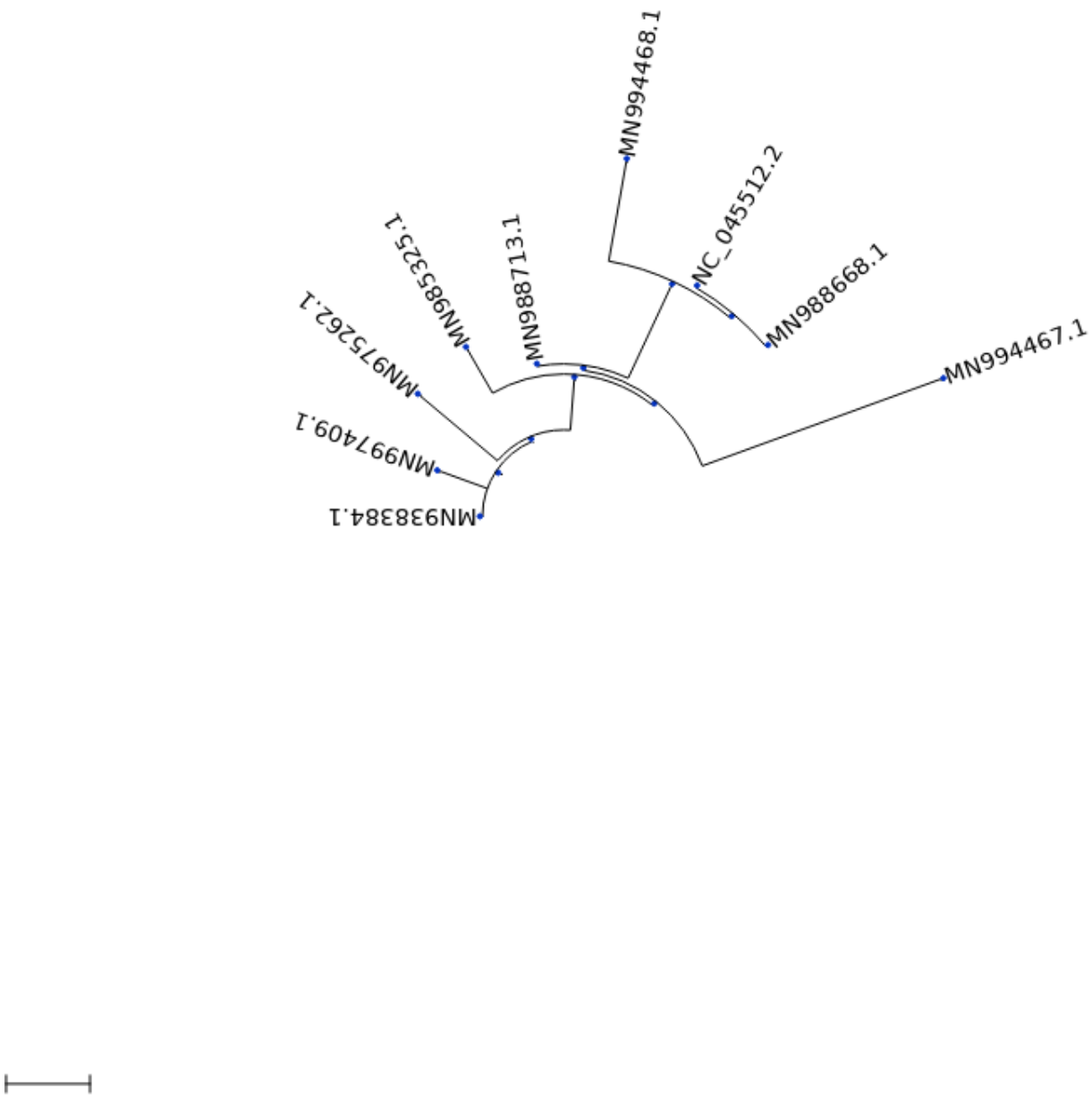
[15]:



[16]:

```
## 180 deg circular tree
ts = TreeStyle()
ts.show_leaf_name = True
ts.mode = "c"
ts.arc_start = -180 # 0 degrees = 3 o'clock
ts.arc_span = 180
t.render("%%inline", tree_style=ts)
```

[16]:



[18]:
```
## read aligned fastq file
fasta_data = dict()
with open(MUSCLE_MSA,'r') as fp:
    header = ''
    fasta_list = list()
    for line in fp:
        line = line.strip()
        if line.startswith('>'):
            if header != '':
                fasta_data.update({header:''.join(fasta_list)})
```

(continues on next page)

(continued from previous page)

```
            header = line.split()[0].replace('>','')
            fasta_list = list()
        else:
            fasta_list.append(line)

    fasta_data.update({header:''.join(fasta_list)})
```

[19]:
```
## tree with aligned seq
for seq_id,seq in fasta_data.items():
    seqFace = SeqMotifFace(seq, gapcolor="red")
    (t & "{0}".format(seq_id)).add_face(seqFace, 0, "aligned")
ts = TreeStyle()
ts.tree_width = 100
t.render("tree_with_aln.png", tree_style=ts);
## check png using image viewer
```

1.2.5 Referenc

- Edgar, R.C. MUSCLE: a multiple sequence alignment method with reduced time and space complexity. BMC Bioinformatics 5, 113 (2004). https://doi.org/10.1186/1471-2105-5-113
- Price, M.N., Dehal, P.S., and Arkin, A.P. (2009) FastTree: Computing Large Minimum-Evolution Trees with Profiles instead of a Distance Matrix. Molecular Biology and Evolution 26:1641-1650, doi:10.1093/molbev/msp077.
- Price, M.N., Dehal, P.S., and Arkin, A.P. (2010) FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments. PLoS ONE, 5(3):e9490. doi:10.1371/journal.pone.0009490.
- ETE 3: Reconstruction, analysis and visualization of phylogenomic data. Jaime Huerta-Cepas, Francois Serra and Peer Bork. Mol Biol Evol 2016; doi: 10.1093/molbev/msw046

[ ]:
```
```

1.3 Tree building for 153 Coronavirus genomes

- **Notebook version**: `v0.0.1`
- **Created by:** Dr. Hiren Ghosh, `Imperial BRC Genomics Facility`
- **Maintained by:** `Imperial BRC Genomics Facility`
- **Docker image:** imperialgenomicsfacility/viral-genome-analysis-notebooks
- **Github repository:** imperial-genomics-facility/viral-genome-notebook-image
- **Created on:** `2020-April-21 14:42`
- **Contact us:** Imperial BRC Genomics Facility
- **License:** Apache License 2.0

1.3.1 Fetch data

Download the aligned corona virus genomes from Genexa

[1]:
```
## Downloading aligned fasta format

!wget https://storage.googleapis.com/sars-cov-2/MSA_SARS2_20200329.fasta
```

```
--2020-04-02 18:49:21--  https://storage.googleapis.com/sars-cov-2/MSA_SARS2_20200329.
↪fasta
Resolving storage.googleapis.com (storage.googleapis.com)... 74.125.129.128, 2607:
↪f8b0:4001:c03::80
Connecting to storage.googleapis.com (storage.googleapis.com)|74.125.129.128|:443...␣
↪connected.
HTTP request sent, awaiting response... 200 OK
Length: 4668949 (4.5M) [application/octet-stream]
Saving to: 'MSA_SARS2_20200329.fasta'

MSA_SARS2_20200329. 100%[===================>]   4.45M  --.-KB/s    in 0.06s

2020-04-02 18:49:21 (70.8 MB/s) - 'MSA_SARS2_20200329.fasta' saved [4668949/4668949]
```

[37]:
```
## DOwnloading data in Clustal format

!wget -O MSA_SARS2_20200329.aln https://storage.googleapis.com/sars-cov-2/MSA_SARS2_
↪20200329.clw
```

```
--2020-04-02 19:10:59--  https://storage.googleapis.com/sars-cov-2/MSA_SARS2_20200329.
↪clw
Resolving storage.googleapis.com (storage.googleapis.com)... 172.217.214.128, 2607:
↪f8b0:4001:c07::80
Connecting to storage.googleapis.com (storage.googleapis.com)|172.217.214.128|:443...␣
↪connected.
HTTP request sent, awaiting response... 200 OK
Length: 5932931 (5.7M) [application/octet-stream]
Saving to: 'MSA_SARS2_20200329.aln'

MSA_SARS2_20200329. 100%[===================>]   5.66M  --.-KB/s    in 0.1s

2020-04-02 19:10:59 (57.8 MB/s) - 'MSA_SARS2_20200329.aln' saved [5932931/5932931]
```

1.3.2 Build phylogenetic tree with FastTree

[2]:
```
!FastTree -nt -gtr < MSA_SARS2_20200329.fasta >MSA_SARS2_20200329.tree
```

```
FastTree Version 2.1.10 Double precision (No SSE3)
Alignment: standard input
Nucleotide distances: Jukes-Cantor Joins: balanced Support: SH-like 1000
Search: Normal +NNI +SPR (2 rounds range 10) +ML-NNI opt-each=1
TopHits: 1.00*sqrtN close=default refresh=0.80
ML Model: Generalized Time-Reversible, CAT approximation with 20 rate categories
Ignored unknown character D (seen 1 times)
Ignored unknown character H (seen 13 times)
Ignored unknown character K (seen 1 times)
Ignored unknown character R (seen 1 times)
```

(continues on next page)

(continued from previous page)

```
Ignored unknown character S (seen 3 times)
Ignored unknown character V (seen 1 times)
Ignored unknown character W (seen 3 times)
Ignored unknown character X (seen 42 times)
Ignored unknown character Y (seen 17 times)
Initial topology in 2.67 seconds0 of    123   126 seqs (at seed    100)
Refining topology: 28 rounds ME-NNIs, 2 rounds ME-SPRs, 14 rounds ML-NNIs
Total branch-length 0.010 after 19.69 sec, 101 of 124 splits, 0 changes   ax delta 0.
↪000)

WARNING! This alignment consists of closely-related and very-long sequences.
WARNING! FastTree (or other standard maximum-likelihood tools)
may not be appropriate for aligments of very closely-related sequences
like this one, as FastTree does not account for recombination or gene conversion

ML-NNI round 1: LogLk = -45578.360 NNIs 57 max delta 6.62 Time 27.53ges (max delta 6.
↪617)
GTR Frequencies: 0.2990 0.1837 0.1962 0.3211ep 12 of 12
GTR rates(ac ag at cg ct gt) 0.4703 1.4001 0.5361 0.6047 2.2160 1.0000
Switched to using 20 rate categories (CAT approximation)20 of 20
Rate categories were divided by 0.626 so that average rate = 1.0
CAT-based log-likelihoods may not be comparable across runs
Use -gamma for approximate but comparable Gamma(20) log-likelihoods
ML-NNI round 2: LogLk = -44402.405 NNIs 34 max delta 0.00 Time 89.79ges (max delta 0.
↪000)
Turning off heuristics for final round of ML NNIs (converged)
ML-NNI round 3: LogLk = -44401.762 NNIs 27 max delta 0.55 Time 104.20 (final)delta 0.
↪546)
Optimize all lengths: LogLk = -44401.740 Time 107.77
Total time: 128.59 seconds Unique: 126/153 Bad splits: 0/123rnal splits
```

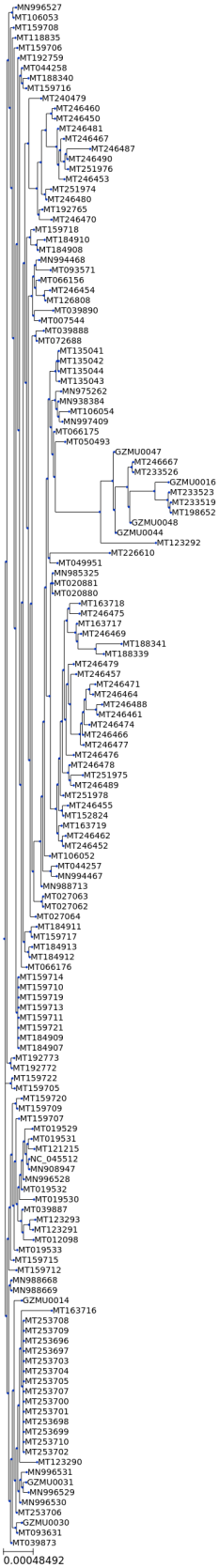
1.3.3 Plot trees

[3]:
```
import os
import requests
import json
import time
from ete3 import Tree, TreeStyle,SeqMotifFace
from PIL import Image
os.environ['QT_QPA_PLATFORM']='offscreen'
```

[4]:
```
t = Tree("MSA_SARS2_20200329.tree")
```

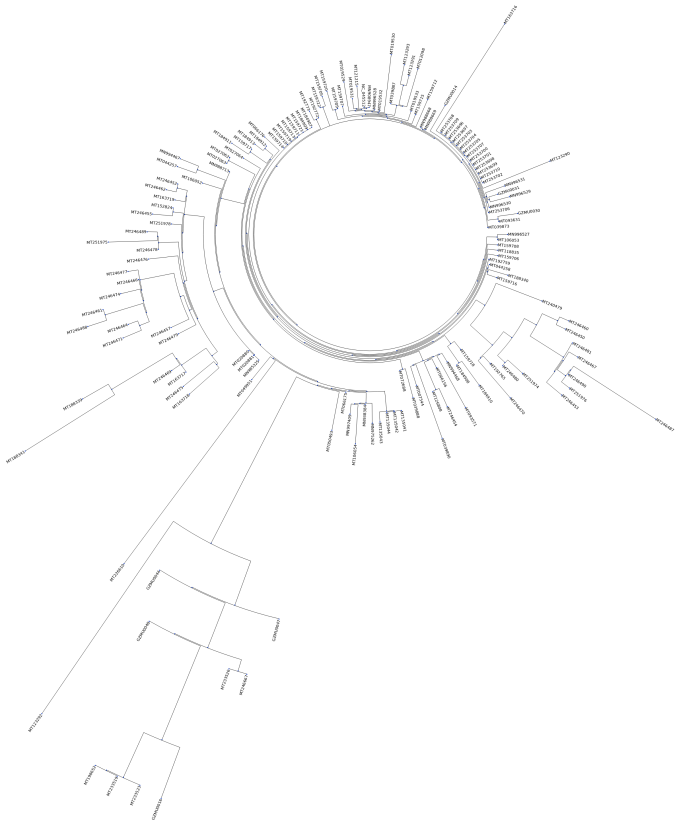
[9]:
```
## plain tree
t.render("%%inline")
```

[9]:


MN996527
MT106053
MT159708
MT118835
MT159706
MT192759
MT044258
MT188340
MT159716
MT240479
MT246460
MT246450
MT246481
MT246467
MT246487
MT246490
MT251976
MT246453
MT251974
MT246480
MT192765
MT246470
MT159718
MT184910
MT184908
MN994468
MT093571
MT066156
MT246454
MT126808
MT039890
MT007544
MT039888
MT072688
MT135041
MT135042
MT135044
MT135043
MN975262
MN938384
MT106054
MN997409
MT066175
MT050493
GZMU0047
MT246667
MT233526
GZMU0016
MT233523
MT233519
MT198652
GZMU0048
GZMU0044
MT123292
MT226610
MT049951
MN985325
MT020881
MT020880
MT163718
MT246475
MT163717
MT246469
MT188341
MT188339
MT246479
MT246457
MT246471
MT246464
MT246488
MT246461
MT246474
MT246466
MT246477
MT246476
MT246478
MT251975
MT246489
MT251978
MT246455
MT152824
MT163719
MT246462
MT246452
MT106052
MT044257
MN994467
MN988713
MT027063
MT027062
MT027064
MT184911
MT159717
MT184913
MT184912
MT066176
MT159714
MT159710
MT159719
MT159713
MT159711
MT159721
MT184909
MT184907
MT192773
MT192772
MT159722
MT159705
MT159720
MT159709
MT159707
MT019529
MT019531
MT121215
NC_045512
MN908947
MN996528
MT019532
MT019530
MT039887
MT123293
MT123291
MT012098
MT019533
MT159715
MT159712
MN988668
MN988669
GZMU0014
MT163716
MT253708
MT253709
MT253696
MT253697
MT253703
MT253704
MT253705
MT253707
MT253700
MT253701
MT253698
MT253699
MT253710
MT253702
MT123290
MN996531
GZMU0031
MN996529
MN996530
MT253706
GZMU0030
MT093631
MT039873
0.00048492

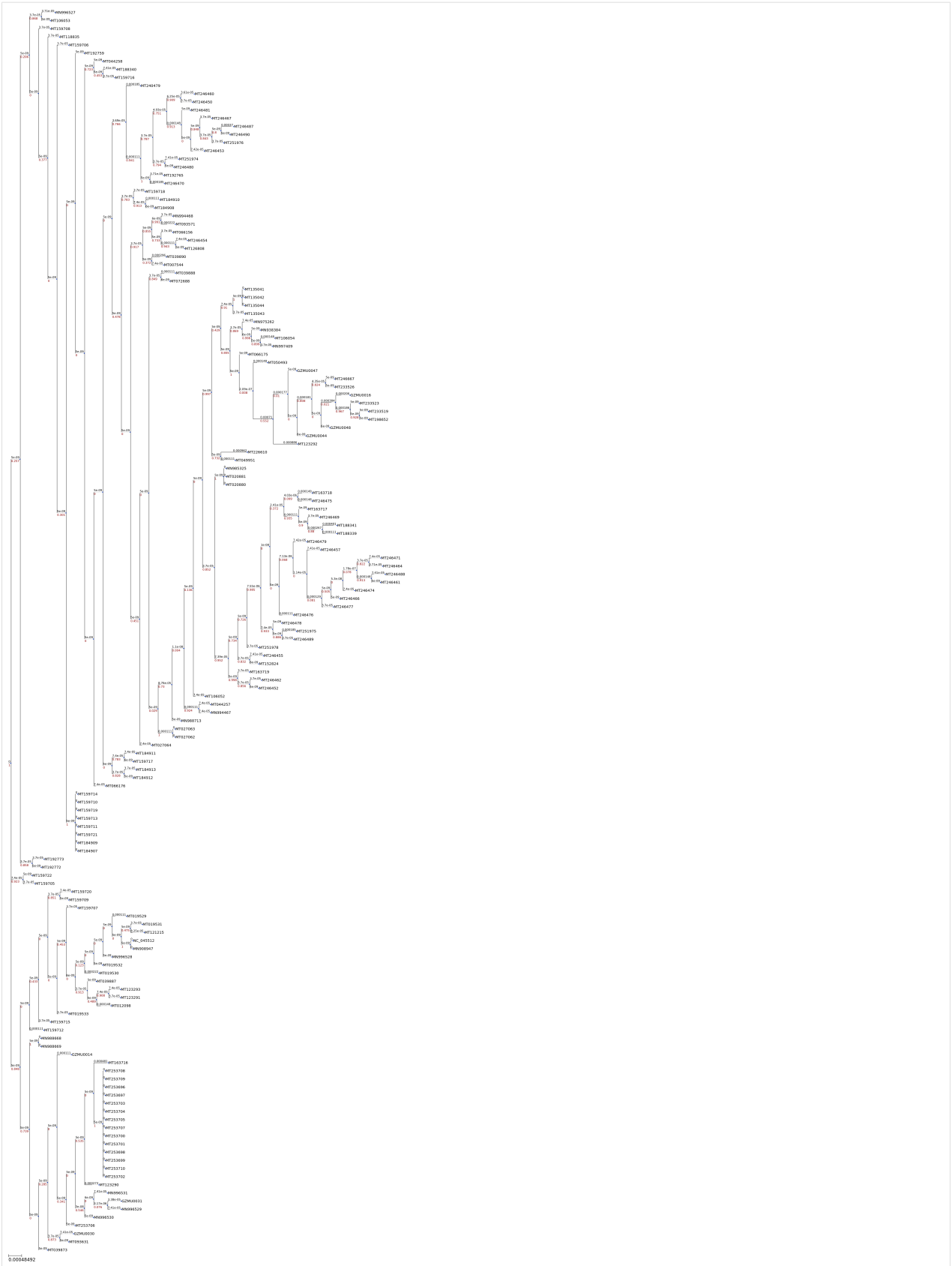
[32]:
```
## circular tree
ts = TreeStyle()
ts.mode = "c"
#ts.scale = 20
t.render("%%inline",tree_style=ts)
```

[32]:



[12]:
```
## tree with branch length
ts = TreeStyle()
ts.show_leaf_name = True
ts.show_branch_length = True
ts.show_branch_support = True
t.render("%%inline",tree_style=ts)
```

[12]:


MN996527
MT106053
MT159708
MT118835
MT159706
MT192759
MT044258
MT188340
MT159716
MT240479
MT246460
MT246450
MT246481
MT246467
MT246487
MT246490
MT251976
MT246453
MT251974
MT246480
MT192765
MT246470
MT159718
MT184910
MT184908
MN994468
MT093571
MT086156
MT246454
MT126808
MT039890
MT007544
MT039888
MT072688
MT135041
MT135042
MT135044
MT135043
MN975262
MN938304
MN997409
MT066175
MT106054
MT050493
GZMU0047
MT246467
MT233526
GZMU0016
MT233523
MT233519
MT198652
GZMU0048
GZMU0044
MT123292
MT226610
MT049951
MN985125
MT020881
MT020880
MT163718
MT246475
MT163717
MT246469
MT188341
MT188339
MT246473
MT246457
MT246471
MT246464
MT246468
MT246441
MT246474
MT246466
MT246477
MT246476
MT246478
MT251975
MT246489
MT251978
MT246455
MT152824
MT163719
MT246462
MT246452
MT196652
MT044257
MN994467
MN908715
MT027063
MT027062
MT027064
MT184911
MT159717
MT184913
MT184912
MT066176
MT159714
MT159710
MT159719
MT159713
MT159711
MT159721
MT184909
MT184907
MT192773
MT192772
MT159722
MT159705
MT159720
MT159709
MT159707
MT019529
MT019531
MT121215
MG_045512
MN908947
MN996528
MT019532
MT019530
MT039887
MT123293
MT123291
MT012098
MT019533
MT159715
MT159712
MN908668
MN908669
GZMU0014
MT163716
MT253708
MT253709
MT253696
MT253697
MT253703
MT253704
MT253705
MT253707
MT253700
MT253701
MT253698
MT253699
MT253710
MT253702
MT123290
MN996531
GZMU0031
MN996529
MN996530
MT253706
GZMU0030
MT039873
0.0048492

[31]:
```
## 180 deg circular tree
ts = TreeStyle()
ts.show_leaf_name = True
ts.mode = "c"
ts.arc_start = -180 # 0 degrees = 3 o'clock
ts.arc_span = 180
t.render("180_deg_circular_tree.png",tree_style=ts)
t.render("%%inline",tree_style=ts)
```

[33]:
```
## read aligned fastq file
fasta_data = dict()
with open('MSA_SARS2_20200329.fasta','r') as fp:
    header = ''
    fasta_list = list()
    for line in fp:
        line = line.strip()
        if line.startswith('>'):
            if header != '':
                fasta_data.update({header:''.join(fasta_list)})
            header = line.split()[0].replace('>','')
            fasta_list = list()
        else:
            fasta_list.append(line)

    fasta_data.update({header:''.join(fasta_list)})
```

[35]:
```
## plot tree with alignment
for seq_id,seq in fasta_data.items():
    seqFace = SeqMotifFace(seq, gapcolor="red")
    (t & "{0}".format(seq_id)).add_face(seqFace, 0, "aligned")
ts = TreeStyle()
ts.tree_width = 100
t.render("tree_with_aln.png", tree_style=ts);
```

1.3.4 Reference

- Price, M.N., Dehal, P.S., and Arkin, A.P. (2009) FastTree: Computing Large Minimum-Evolution Trees with Profiles instead of a Distance Matrix. Molecular Biology and Evolution 26:1641-1650, doi:10.1093/molbev/msp077.
- Price, M.N., Dehal, P.S., and Arkin, A.P. (2010) FastTree 2 – Approximately Maximum-Likelihood Trees for Large Alignments. PLoS ONE, 5(3):e9490. doi:10.1371/journal.pone.0009490.
- ETE 3: Reconstruction, analysis and visualization of phylogenomic data. Jaime Huerta-Cepas, Francois Serra and Peer Bork. Mol Biol Evol 2016; doi: 10.1093/molbev/msw046

[ ]:
```
```

1.4 Alignment and tree building using Nextstrain

- **Notebook version**: `v0.0.1`

- **Created by:** Dr. Hiren Ghosh, Imperial BRC Genomics Facility
- **Maintained by:** Imperial BRC Genomics Facility
- **Docker image:** imperialgenomicsfacility/viral-genome-analysis-notebooks
- **Github repository:** imperial-genomics-facility/viral-genome-notebook-image
- **Created on:** 2020-April-21 14:42
- **Contact us:** Imperial BRC Genomics Facility
- **License:** Apache License 2.0

1.4.1 Introduction

Nextstrain is an open-source project to harness the scientific and public health potential of pathogen genome data. Real time tracking of pathogen evolution data and visualization is provided by nextstrain.org. For example, the real time tracking of genomic data for COVID-19 pandemic can be found here: nextstrain.org/ncov/global. This notebook will focus on using the bioinformatics toolkits provided by Nextstrain and setting up a docker based local Nextstrain server for output visualization.

1.4.2 Configure notebook for run

[1]:
```
CPU_THREADS = 1
```

1.4.3 Load required python libraries

[2]:
```
import os,requests,json,time
import pandas as pd
```

1.4.4 Fetch genome sequences

[3]:
```
def fetch_genome_fasta_from_ncbi(refseq_id,output_path='.',file_format='fasta'):
  '''
  A function for fetching the genome fasta sequences from NCBI

  :param refseq_id: NCBI genome id
  :param output_path: Path to dump genome files, default '.'
  :param file_format: Output fileformat, default fasta, supported formats are 'fasta'␣
↪and 'gb'
  :return: output_file
  '''
  try:
    url = \
      'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nucleotide&id={0}&
↪rettype={1}'.\
        format(refseq_id,file_format)
    r = requests.get(url)
    if r.status_code != 200:
      raise ValueError(
              'Failed to download data from NCBI for id {0}, error code: {1}'.\
                format(refseq_id,r.status_code))
```

(continues on next page)

(continued from previous page)

```
    fasta_data = r.content.decode('utf-8')
    output_file = \
      os.path.join(
        os.path.abspath(output_path),
        '{0}.{1}'.format(refseq_id,file_format))
    with open(output_file,'w') as fp:
      fp.write(fasta_data)
    print('Downloaded genome seq for {0}'.format(refseq_id))
    return output_file
  except Exception as e:
    print('Failed to download data for {0} from NCBI, error: {1}'.format(refseq_id,e))
```

[4]:
```
genome_list = [
    'MN988668.1',
    'MN938384.1',
    'MN975262.1',
    'MN985325.1',
    'MN988713.1',
    'MN994467.1',
    'MN994468.1',
    'MN997409.1'
]
```

[5]:
```
fasta_list = []
for id in genome_list:
  fasta_list.\
    append(
      fetch_genome_fasta_from_ncbi(
        refseq_id=id,
        output_path='/tmp'))
  time.sleep(2)
fasta_list
```

```
Downloaded genome seq for MN988668.1
Downloaded genome seq for MN938384.1
Downloaded genome seq for MN975262.1
Downloaded genome seq for MN985325.1
Downloaded genome seq for MN988713.1
Downloaded genome seq for MN994467.1
Downloaded genome seq for MN994468.1
Downloaded genome seq for MN997409.1
```

[5]:
```
['/tmp/MN988668.1.fasta',
 '/tmp/MN938384.1.fasta',
 '/tmp/MN975262.1.fasta',
 '/tmp/MN985325.1.fasta',
 '/tmp/MN988713.1.fasta',
 '/tmp/MN994467.1.fasta',
 '/tmp/MN994468.1.fasta',
 '/tmp/MN997409.1.fasta']
```

[6]:
```
reference_gb = \
  fetch_genome_fasta_from_ncbi(
    'NC_045512.2',
    file_format='gb',
    output_path='/tmp')
reference_gb
```

```
Downloaded genome seq for NC_045512.2
```

[6]:
```
'/tmp/NC_045512.2.gb'
```

[7]:
```
MERGED_FASTA = '/tmp/corona_merged.fasta'
!rm -f $MERGED_FASTA
for f in fasta_list:
  !cat $f >> $MERGED_FASTA

## count the genome sequences in the merged fasta file
!grep '>' $MERGED_FASTA|wc -l
```

```
8
```

1.4.5 Multiple sequence alignment and tree building

[8]:
```
ALIGNED_FASTA = '/tmp/corona_merged.afa'
!augur align \
  --sequences $MERGED_FASTA \
  --reference-sequence $reference_gb \
  --nthreads $CPU_THREADS \
  --output $ALIGNED_FASTA \
  --fill-gaps
```

```
using mafft to align via:
        mafft --reorder --anysymbol --nomemsave --adjustdirection --thread 1 /tmp/
↪corona_merged.afa.to_align.fasta 1> /tmp/corona_merged.afa 2> /tmp/corona_merged.
↪afa.log

        Katoh et al, Nucleic Acid Research, vol 30, issue 14
        https://doi.org/10.1093%2Fnar%2Fgkf436

No gaps in alignment to trim (with respect to the reference, NC_045512.2)
```

[9]:
```
RAW_TREE = '/tmp/corona_raw.tree'
!augur tree \
  --alignment $ALIGNED_FASTA \
  --method fasttree \
  --output $RAW_TREE \
  --nthreads $CPU_THREADS
```

```
Cannot specify model unless using IQTree. Model specification ignored.
Building a tree via:
        FastTreeMP -nosupport -nt /tmp/corona_merged.afa  1> /tmp/corona_raw.tree 2> /
↪tmp/corona_raw.tree.log
        Price et al: FastTree 2 - Approximately Maximum-Likelihood Trees for Large␣
↪Alignments.
        PLoS ONE 5(3): e9490. https://doi.org/10.1371/journal.pone.0009490


Building original tree took 1.8078341484069824 seconds
```

1.4.6 Metadata configs for Nextstrain

[10]:
```
metadata = [{
  'strain':'NC_045512.2',
  'virus':'SARS-CoV-2',
  'accession':'NC_045512',
  'date':'2020-03-30',
  'region':'China',
  'country':'China',
  'division':'China',
  'city':'Wuhan',
  'db':'genbank',
  'segment':'genome',
  'authors':'Baranov,P.V., Henderson,C.M., Anderson,C.B., Gesteland,R.F.,Atkins,J.F.␣
↪and Howard,M.T.',
  'url':'https://www.ncbi.nlm.nih.gov/nuccore/NC_045512.2',
  'title':'Programmed ribosomal frameshifting in decoding the SARS-CoV genome',
  'journal':'Virology 332 (2), 498-510 (2005)',
  'paper_url':'https://www.ncbi.nlm.nih.gov/pubmed/15680415'
  },
  {
  'strain':'MN988668.1',
  'virus':'SARS-CoV-2',
  'accession':'MN988668',
  'date':'2020-03-11',
  'region':'China',
  'country':'China',
  'division':'China',
  'city':'Wuhan',
  'db':'genbank',
  'segment':'genome',
  'authors':'Chen,L., Liu,W., Zhang,Q., Xu,K., Ye,G., Wu,W., Sun,Z., Liu,F.,Wu,K.,␣
↪Mei,Y., Zhang,W., Chen,Y., Li,Y., Shi,M., Lan,K. and Liu,Y.',
  'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN988668.1',
  'title':'RNA based mNGS approach identifies a novel human coronavirus from two␣
↪individual pneumonia cases in 2019 Wuhan outbreak',
  'journal':'Emerg Microbes Infect (2019) In press',
  'paper_url':''
  },
  {
  'strain':'MN938384.1',
  'virus':'SARS-CoV-2',
  'accession':'MN938384',
  'date':'2020-02-11',
  'region':'China',
  'country':'China',
  'division':'China',
  'city':'Shenzhen',
  'db':'genbank',
  'segment':'genome',
  'authors':"""Chan,J.F.-W., Yuan,S., Kok,K.H., To,K.K.-W., Chu,H., Yang,J.,
            Xing,F., Liu,J., Yip,C.C.-Y., Poon,R.W.-S., Tsai,H.W., Lo,S.K.-F.,
            Chan,K.H., Poon,V.K.-M., Chan,W.M., Ip,J.D., Cai,J.P.,
            Cheng,V.C.-C., Chen,H., Hui,C.K.-M. and Yuen,K.Y.""",
  'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN938384.1',
  'title':'A familial cluster of pneumonia associated with the 2019 novel coronavirus␣
↪indicating person-to-person transmission: a study of a family cluster',
```

(continues on next page)

(continued from previous page)

```
 'journal':'Lancet (2020) In press',
 'paper_url':''
 },
 {
 'strain':'MN975262.1',
 'virus':'SARS-CoV-2',
 'accession':'MN975262',
 'date':'2020-02-11',
 'region':'China',
 'country':'China',
 'division':'China',
 'city':'Shenzhen',
 'db':'genbank',
 'segment':'genome',
 'authors':"""Chan,J.F.-W., Yuan,S., Kok,K.H., To,K.K.-W., Chu,H., Yang,J.,
          Xing,F., Liu,J., Yip,C.C.-Y., Poon,R.W.-S., Tsai,H.W., Lo,S.K.-F.,
          Chan,K.H., Poon,V.K.-M., Chan,W.M., Ip,J.D., Cai,J.P.,
          Cheng,V.C.-C., Chen,H., Hui,C.K.-M. and Yuen,K.Y.""",
 'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN975262.1',
 'title':"""A familial cluster of pneumonia associated with the 2019 novel
          coronavirus indicating person-to-person transmission: a study of a
          family cluster""",
 'journal':'Lancet (2020) In press',
 'paper_url':''
 },
 {
 'strain':'MN985325.1',
 'virus':'SARS-CoV-2',
 'accession':'MN985325',
 'date':'2020-03-27',
 'region':'USA',
 'country':'USA',
 'division':'USA',
 'city':'Atlanta',
 'db':'genbank',
 'segment':'genome',
 'authors':"""Harcourt,J., Tamin,A., Lu,X., Kamili,S., Sakthivel,S.K., Murray,J.,
          Queen,K., Tao,Y., Paden,C.R., Zhang,J., Li,Y., Uehara,A., Wang,H.,
          Goldsmith,C., Bullock,H.A., Wang,L., Whitaker,B., Lynch,B.,
          Gautam,R., Schindewolf,C., Lokugamage,K.G., Scharton,D.,
          Plante,J.A., Mirchandani,D., Widen,S.G., Narayanan,K., Makino,S.,
          Ksiazek,T.G., Plante,K.S., Weaver,S.C., Lindstrom,S., Tong,S.,
          Menachery,V.D. and Thornburg,N.J.""",
 'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN985325.1',
 'title':"""Severe Acute Respiratory Syndrome Coronavirus 2 from Patient with
          2019 Novel Coronavirus Disease, United States""",
 'journal':'Emerging Infect. Dis. 26 (6) (2020) In press',
 'paper_url':''
 },
 {
 'strain':'MN988713.1',
 'virus':'SARS-CoV-2',
 'accession':'MN988713',
 'date':'2020-02-11',
 'region':'USA',
 'country':'USA',
 'division':'USA',
```

(continues on next page)

(continued from previous page)

```
 'city':'Illinois',
 'db':'genbank',
 'segment':'genome',
 'authors':"""Tao,Y., Queen,K., Paden,C.R., Zhang,J., Li,Y., Uehara,A., Lu,X.,
           Lynch,B., Sakthivel,S.K.K., Whitaker,B.L., Kamili,S., Wang,L.,
           Murray,J.R., Gerber,S.I., Lindstrom,S. and Tong,S.""",
 'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN988713.1',
 'title':'Novel betacoronavirus, Illinois',
 'journal':'',
 'paper_url':''
 },
 {
 'strain':'MN994467.1',
 'virus':'SARS-CoV-2',
 'accession':'MN994467',
 'date':'2020-02-11',
 'region':'USA',
 'country':'USA',
 'division':'USA',
 'city':'California',
 'db':'genbank',
 'segment':'genome',
 'authors':"""Uehara,A., Queen,K., Tao,Y., Li,Y., Paden,C.R., Zhang,J., Lu,X.,
           Lynch,B., Sakthivel,S.K.K., Whitaker,B.L., Kamili,S., Wang,L.,
           Murray,J.R., Gerber,S.I., Lindstrom,S. and Tong,S.""",
 'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN994467.1',
 'title':'nCoV-2019 California case 1',
 'journal':'',
 'paper_url':''
 },
 {
 'strain':'MN994468.1',
 'virus':'SARS-CoV-2',
 'accession':'MN994468',
 'date':'2020-02-11',
 'region':'USA',
 'country':'USA',
 'division':'USA',
 'city':'California',
 'db':'genbank',
 'segment':'genome',
 'authors':"""Uehara,A., Queen,K., Tao,Y., Li,Y., Paden,C.R., Zhang,J., Lu,X.,
           Lynch,B., Sakthivel,S.K.K., Whitaker,B.L., Kamili,S., Wang,L.,
           Murray,J.R., Gerber,S.I., Lindstrom,S. and Tong,S.""",
 'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN994468.1',
 'title':'nCoV-2019 California case 2',
 'journal':'',
 'paper_url':''
 },
 {
 'strain':'MN997409.1',
 'virus':'SARS-CoV-2',
 'accession':'MN997409',
 'date':'2020-02-11',
 'region':'USA',
 'country':'USA',
 'division':'USA',
```

(continues on next page)

(continued from previous page)

```
  'city':'Arizona',
  'db':'genbank',
  'segment':'genome',
  'authors':"""Tao,Y., Paden,C.R., Queen,K., Uehara,A., Li,Y., Zhang,J., Lu,X.,
            Lynch,B., Sakthivel,S.K.K., Whitaker,B.L., Kamili,S., Wang,L.,
            Murray,J.R., Gerber,S.I., Lindstrom,S. and Tong,S.""",
  'url':'https://www.ncbi.nlm.nih.gov/nuccore/MN997409.1',
  'title':'nCoV-2019 sequence from Arizona case',
  'journal':'',
  'paper_url':''
  }
]
```

[11]:
```
## load metadata and write tsv files
METADATA_FILE = '/tmp/metadata.tsv'

df = pd.DataFrame(metadata)
df.to_csv(METADATA_FILE,sep='\t',index=False)
```

[12]:
```
REFINED_TREE = '/tmp/corona_refined.tree'
BRANCH_LENGTHS_JSON = '/tmp/corona_branch_lengths.json'
!augur refine \
  --tree $RAW_TREE \
  --alignment $ALIGNED_FASTA \
  --metadata $METADATA_FILE \
  --output-tree $REFINED_TREE \
  --output-node-data $BRANCH_LENGTHS_JSON \
  --timetree \
  --coalescent opt \
  --date-confidence \
  --date-inference marginal \
  --clock-filter-iqd 4
```

```
1.07    TreeTime.reroot: with method or node: least-squares

1.07    TreeTime.reroot: rerooting will ignore covariance and shared ancestry.

1.09    TreeTime.reroot: with method or node: least-squares

1.09    TreeTime.reroot: rerooting will ignore covariance and shared ancestry.

1.11    WARNING: Previous versions of TreeTime (<0.7.0) RECONSTRUCTED sequences of
        tips at positions with AMBIGUOUS bases. This resulted in unexpected
        behavior is some cases and is no longer done by default. If you want to
        replace those ambiguous sites with their most likely state, rerun with
        `reconstruct_tip_states=True` or `--reconstruct-tip-states`.

1.16    TreeTime.reroot: with method or node: least-squares

1.16    TreeTime.reroot: rerooting will account for covariance and shared ancestry.

1.26    ###TreeTime.run: INITIAL ROUND

1.69    TreeTime.reroot: with method or node: least-squares
```

(continues on next page)

(continued from previous page)

```
1.70    TreeTime.reroot: rerooting will account for covariance and shared ancestry.

1.72    ###TreeTime.run: rerunning timetree after rerooting

2.17    ###TreeTime.run: ITERATION 1 out of 2 iterations

2.92    ###TreeTime.run: ITERATION 2 out of 2 iterations

3.75    ###TreeTime.run: CONVERGED

5.71    ###TreeTime.run: FINAL ROUND - confidence estimation via marginal
        reconstruction

Inferred a time resolved phylogeny using TreeTime:
        Sagulenko et al. TreeTime: Maximum-likelihood phylodynamic analysis
        Virus Evolution, vol 4, https://academic.oup.com/ve/article/4/1/vex042/4794731

updated tree written to /tmp/corona_refined.tree
node attributes written to /tmp/corona_branch_lengths.json
```

[13]:
```
## Annotate the Phylogeny
### Reconstruct Ancestral Traits
TRAITS_JSON = '/tmp/corona_traits.json'
!augur traits \
  --tree $REFINED_TREE \
  --metadata $METADATA_FILE \
  --output $TRAITS_JSON \
  --columns region country \
  --confidence
```

```
Assigned discrete traits to 9 out of 9 taxa.

NOTE: previous versions (<0.7.0) of this command made a 'short-branch
length assumption. TreeTime now optimizes the overall rate numerically
and thus allows for long branches along which multiple changes
accumulated. This is expected to affect estimates of the overall rate
while leaving the relative rates mostly unchanged.
Assigned discrete traits to 9 out of 9 taxa.

NOTE: previous versions (<0.7.0) of this command made a 'short-branch
length assumption. TreeTime now optimizes the overall rate numerically
and thus allows for long branches along which multiple changes
accumulated. This is expected to affect estimates of the overall rate
while leaving the relative rates mostly unchanged.

Inferred ancestral states of discrete character using TreeTime:
        Sagulenko et al. TreeTime: Maximum-likelihood phylodynamic analysis
        Virus Evolution, vol 4, https://academic.oup.com/ve/article/4/1/vex042/4794731

results written to /tmp/corona_traits.json
```

[14]:
```
### Infer Ancestral Sequences
NT_MUTS_JSON = '/tmp/corona_nt_muts.json'
!augur ancestral \
  --tree $REFINED_TREE \
```

(continues on next page)

(continued from previous page)

```
  --alignment $ALIGNED_FASTA \
  --output-node-data $NT_MUTS_JSON \
  --inference joint
```

```
Inferred ancestral sequence states using TreeTime:
        Sagulenko et al. TreeTime: Maximum-likelihood phylodynamic analysis
        Virus Evolution, vol 4, https://academic.oup.com/ve/article/4/1/vex042/4794731

ancestral mutations written to /tmp/corona_nt_muts.json
```

[15]:
```
### Identify Amino-Acid Mutations
AA_MUTS_JSON = '/tmp/corona_aa_muts.json'
!augur translate \
  --tree $REFINED_TREE \
  --ancestral-sequences $NT_MUTS_JSON \
  --reference-sequence $reference_gb \
  --output $AA_MUTS_JSON
```

```
Read in 12 features from reference sequence file
amino acid mutations written to /tmp/corona_aa_muts.json
```

[16]:
```
%%file /tmp/lat_longs.tsv
country,china,31.0740838,107.7450643
country,usa,42.6584011,-80.2716978
region,china,31.0740838,107.7450643
region,usa,42.6584011,-80.2716978
```

```
Writing /tmp/lat_longs.tsv
```

[17]:
```
%%file /tmp/colors.tsv
country,china,#511EA8
country,usa,#cd5700
region,china,#511EA8
region,usa,#cd5700
```

```
Writing /tmp/colors.tsv
```

[18]:
```
!sed -i 's|,|\t|g' /tmp/lat_longs.tsv
```

[19]:
```
!sed -i 's|,|\t|g' /tmp/colors.tsv
```

[20]:
```
%%file /tmp/auspice_config.json
{
  "title": "Covid-19 5 samples",
  "maintainers": [
    {"name": "Your name", "url": "your url"}
  ],
  "build_url": "your build url",
  "colorings": [
    {
      "key": "gt",
      "title": "Genotype",
      "type": "categorical"
    },
    {
```

(continues on next page)

(continued from previous page)

```
      "key": "num_date",
      "title": "Date",
      "type": "continuous"
    },
    {
      "key": "author",
      "title": "Author",
      "type": "categorical"
    },
    {
      "key": "country",
      "title": "Country",
      "type": "categorical"
    },
    {
      "key": "region",
      "title": "Region",
      "type": "categorical"
    }
  ],
  "geo_resolutions": [
    "country",
    "region"
  ],
  "panels": [
     "tree",
     "map"
  ],
  "display_defaults": {
    "map_triplicate": true
  },
  "filters": [
    "country",
    "region",
    "author"
  ]
}
```

```
Writing /tmp/auspice_config.json
```

[21]:
```
### Export the Results
NEXTSTRAIN_SERVER_JSON = 'corona.json'
!augur export v2 \
  --tree $REFINED_TREE \
  --metadata $METADATA_FILE \
  --node-data $BRANCH_LENGTHS_JSON \
              $TRAITS_JSON \
              $NT_MUTS_JSON \
              $AA_MUTS_JSON \
  --colors /tmp/colors.tsv \
  --lat-longs /tmp/lat_longs.tsv \
  --auspice-config /tmp/auspice_config.json \
  --output $NEXTSTRAIN_SERVER_JSON
```

```
Validating schema of '/tmp/corona_aa_muts.json'...
Validating config file /tmp/auspice_config.json against the JSON schema
Validating schema of '/tmp/auspice_config.json'...
```

(continues on next page)

(continued from previous page)

```
Validating produced JSON
Validating schema of 'corona.json'...
Validating that the JSON is internally consistent...
Validation of 'corona.json' succeeded.
```

1.4.7 Nextstrain visualization

- Download corona.json file to `/local/path/data`
- Install docker on local PC
- Get docker image: `docker pull nextstrain/base`
- Run docker image: `docker run -it -p4000:4000 -v /local/path:/nextstrain.org -e HOST=0.0.0.0 nextstrain/base auspice view --datasetDir /nextstrain.org/data`
- Access `localhost:4000` in browser

1.4.8 References

- Hadfield et al., Nextstrain: real-time tracking of pathogen evolution, Bioinformatics (2018)

[ ]: