**Instructions for updating RVDB**

**1. Create folders for update**

First, create a directory for the update. This should be in “RVDBv$version” format, for example the following line would create the folder for version 12.0:

*mkdir RVDBv12.0*

The folder structure for an RVDB update is three separate main folders for GenBank, TPA, and RefSeq sequences, in \*\_“$month”.”$year” format. For example, if the month were nov and the year 201, the folders could be created using Windows cmd.exe using the following command:

*cd RVDBv12.0*

*mkdir GenBank\_raw\_data\_nov.2017 && mkdir TPA\_raw\_data\_nov.2017 && mkdir RefSeq\_raw\_data\_nov.2017*

The sub-folder structure for an RVDB update depends on the main folder. All main folders have “log” and “scripts” sub-folders. The main GenBank and TPA folders also have poskw\_out\_”$month”.”$year”, sizemirna\_out\_”$month”.”$year”, and negkw\_out\_”$month”.”$year sub-folders. So, for example, for the GenBank main folder you could enter the following commands:

*cd GenBank*

*mkdir log && mkdir scripts && mkdir poskw\_out\_dec.2017 && mkdir sizemirna\_out\_dec.2017 && mkdir negkw\_out\_dec.2017*

**2. Download raw sequences from NCBI FTP**

**RefSeq viral.** Navigate to the RefSeq main folder, log on to NCBI’s RefSeq ftp site, [ftp.ncbi.nih.gov/refseq/release/viral](ftp://ftp.ncbi.nih.gov/refseq/release/viral) (note, from cmd.exe you have to ftp into [ftp.ncbi.nih.gov](ftp://ftp.ncbi.nih.gov) first, provide a username and password, type “binary” and “prompt” to switch the binary mode to “I” and to turn off file download prompts, then navigate to refseq/release/viral), and download the four files: viral.1.1.genomic.fna.gz (fasta file), viral.2.1.genomic.fna.gz (fasta file), viral.1.genomic.gbff.gz (genbank flat file), and viral.2.genomic.gbff.gz (genbank flat file). This can be done using the ftp command:

*ftp* [*ftp.ncbi.nih.gov*](ftp://ftp.ncbi.nih.gov)

*anonymous*

*anonymous*

*cd refseq*

*cd release*

*cd viral*

*binary*

*prompt*

*mget viral\*genomic\*gz*

**RefSeq viral neighbors.** This is one of only two parts that requires opening a web browser and download some files. Log on to: <https://www.ncbi.nlm.nih.gov/genome/viruses/> and under “Download Viral Genome Data” click on “Accession list of all viral genomes”. Save target in the Refseq data folder as “refseqviral\_neighbors\_mapping.$date.nbr”, where $date is the full date in “month.day.year” format (example “refseqviral\_neighbors\_mapping.dec.28.2017.nbr”) in the Refseq main folder. Open the .nbr file in Excel using the “delimited” option with only “tab” selected (this should be the default). Resave as a .csv (example “refseqviral\_neighbors\_mapping.dec.28.2017.csv”). You can delete the original .nbr file after completing this step.

**Phage.** There is an in-house list of phage keywords that are used to identify and remove phage sequences. It should be saved in the RefSeq main folder, log sub-folder, as “phage\_kws.txt”. It contains the following search strings:

‘ phage’

‘corticovir’

‘cystovir’

‘fusellovir’

‘ inovir’

‘plectrovir’

‘levivir’

‘lipothrixvir’

‘microvir

‘myovir

‘plasmavir’

‘podovir’

‘rudivir’

‘siphovir’

‘tectivir’

**GenBank.** Navigate to the Genbank main folder**, l**og on to NCBI’s Genbank ftp site, <ftp://ftp.ncbi.nih.gov/genbank> , and download gb flat files from the following 10 divisions: ENV, HTC, INV, MAM, PLN, PRI, ROD, VRL, VRT. This can be done using the following ftp command:

*ftp* [*ftp.ncbi.nih.gov*](ftp://ftp.ncbi.nih.gov)

*anonymous*

*anonymous*

*cd genbank*

*binary*

*prompt*

*mget gbenv\*seq.gz gbhtc\*.seq.gz gbinv\*.seq.gz gbmam\*.seq.gz gbpln\*.seq.gz gbpri\*.seq.gz gbrod\*.seq.gz gbvrl\*.seq.gz gbvrt\*.seq.gz*

Also, the official release notes must be downloaded from the GenBank website using a web browser. While this could be done using ftp, the name of the release notes file has to be passed as a parameter later, so it’s best to directly download it and save the file name for later. Visit: <ftp://ftp.ncbi.nih.gov/genbank/release.notes/> and download the most recent file. Save this file in “gb\_releasenotes\_v$version\_$month.$year.txt” format, for example gb\_releasenotes\_v221\_dec.2017.txt.

**TPA.** Navigate to the TPA main folder**, l**og on to NCBI’s TPA ftp site, [ftp.ncbi.nih.gov/tpa/release](ftp://ftp.ncbi.nih.gov/tpa/release) , and download TPA sequence files tpa\_cu.fsa\_nt.gz and con\_tpa\_cu.fsa\_nt.gz. Note that there is no meta-data and therefore not .gbff format files for TPA sequences. The download can be done using the following ftp command:

*ftp* [*ftp.ncbi.nih.gov*](ftp://ftp.ncbi.nih.gov)

*anonymous*

*anonymous*

*cd tpa*

*cd release*

*binary*

*prompt*

*mget \*tpa\*nt.gz*

**3. Running the main pipeline – RefSeq and GenBank.**

The main pipeline performs the core series of operations on the downloaded RefSeq, GenBank, and TPA files. In order, this includes unzipping RefSeq viral, removing phage, pulling in viral neighbor annotation, identifying duplicates of RefSeq (original GenBank entries from which RefSeq entries were created), unzipping and formatting GenBank entries, running checkpoint2 to cross-reference GenBank file contents with the official release notes, and running the positive, size/mirna, and negative screens on GenBank files.

**Main pipeline – RefSeq and GenBank - command block.** Use the following concatenated commands (described individually beneath the command block):

*python F:/UPDATE\_SCRIPTS\_LOGS/parse\_raw\_refseq\_PIPE.py F: dec.2017 12.1 viral.1.1.genomic.fna.gz viral.2.1.genomic.fna.gz && python F:/UPDATE\_SCRIPTS\_LOGS/multiple\_gzunzip\_PIPE.py F: dec.2017 12.1 viral.1.genomic.gbff.gz viral.2.genomic.gbff.gz viral.genomic.gbff && python F:/UPDATE\_SCRIPTS\_LOGS/fileops\_PIPE.py F: dec.2017 12.1 gbff 1000000 && python F:/UPDATE\_SCRIPTS\_LOGS/rs\_acc\_mapping\_PIPE.py F: dec.2017 12.1 && python F:/UPDATE\_SCRIPTS\_LOGS/VDBunzip\_reformat\_gb\_to\_fasta\_PIPE.py F: dec.2017 12.1 gb && python F:/UPDATE\_SCRIPTS\_LOGS/VDBupdate\_checkpoint2\_PIPE.py F: dec.2017 12.1 gb\_releasenotes\_v222\_dec.2017.txt && python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py F: dec.2017 12.1 poskw gb && python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py F: dec.2017 12.1 sizemirna gb && python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py F: dec.2017 12.1 negkw gb*

**Description of commands and scripts.** These scripts called in the command block above do the following:

*python F:/UPDATE\_SCRIPTS\_LOGS/parse\_raw\_refseq.py dec.2017 12.1 viral.1.1.genomic.fna.gz viral.2.1.genomic.fna.gz*

Takes the two RefSeq viral files and outputs a eukaryotic viral fasta file formatted with two lines per entry (header and sequences), as well as a phage file (same format). “dec.2017” is the date of the update, “12.1” is the version of RVDB; these parameters are needed to identify the directory for the update.

*python F:/UPDATE\_SCRIPTS\_LOGS/multiple\_gzunzip\_PIPE.py dec.2017 12.1 viral.1.genomic.gbff.gz viral.2.genomic.gbff.gz viral.genomic.gbff*

Combines the two GenBank flat files for refseq viral into one. “dec.2017” is the date of the update, “12.1” is the version of RVDB; these parameters are needed to identify the directory for the update.

*python F:/UPDATE\_SCRIPTS\_LOGS/fileops\_PIPE.py gbff 1000000*

Splits the combined GenBank flat file into multiple files, so that each can be read into Python. “gbff” is the file type used as input, and 1000000 is the number of entries to include in each split. Note, this script pipes the output filenames as output

*python F:/UPDATE\_SCRIPTS\_LOGS/rs\_acc\_mapping\_PIPE.py*

Using the GenBank flat file metadata for RefSeq viral, finds the duplicate entries’ accessions (original entries, upon which RefSeq viral entries were based). Also uses the RefSeq viral neighbors mapping file to complete the mapping (“F:/RVDBv12.1/RefSeq\_raw\_data.dec.2017/ refseqviral\_neighbors\_mapping.dec.26.2017.csv”). The neighbors are saved in the file (“F:/RVDBv12.1/RefSeq\_raw\_data.dec.2017/neighbor\_accs.txt”); this filename is hard-coded into the next script, which is the unzipping script. The RefSeq duplicate accessions are saved in the file “F:/RVDBv12.1/RefSeq\_raw\_data.dec.2017/refseq\_viral\_originalaccs.txt” ; this filename is also hard-coded as input for the unzipping script.

*python F:/UPDATE\_SCRIPTS\_LOGS/VDBunzip\_reformat\_gb\_to\_fasta\_PIPE.py F:\\RVDBv12.1\\GenBank\_raw\_data\_ dec.2017 gbenv*

Unzips the GenBank division files, labels sequences that are RefSeq viral neighbors during the unzipping. “dec.2017” is the date of the update, “12.1” is the version of RVDB; these parameters are needed to identify the directory for the update. Please note that a modified form of the GenBank Scanner.py script (found in Biopython, typically in Python sub-folder: Lib/site-packages/Bio/GenBank) should be used, in place of the original. The modified version of the Scanner.py script can simply be copied and pasted into the same directory as the original Scanner.py script (overwrite previous). To avoid having to change additional lines in calling scripts to accommodate a different name for Scanner.py, the name of Scanner.py was not changed. The modifications should not hinder any existing functionality of the script, so it can safely be used in place of the original. The modified Scanner.py contains a try/except block in two places, to correct for an error that occurred in a small number of entries, related to the ‘Structured Comments’ metadata. In a very small number of cases, it was noted that the standard Scanner.py script was trying to extract the ‘Structured Comments’ metadata, when this metadata was in fact not present.

*python F:/UPDATE\_SCRIPTS\_LOGS/VDBupdate\_checkpoint2\_PIPE.py*

Runs checkpoint2, generates four output files: F:\UPDATE\_SCRIPTS\\*month\*\*year\*\_checkpt2[a,b,c,d].log”. Note, the names of unzipped files are hard-coded into the semantic screen script that is called next: SEM-R\_PIPE.py, which is described below.

The first file (“a.log” ending) is a print-out of a running total of files / division, seqs / division, after each .seq.gz file is read. This file is time-stamped, so it’s main purpose is to show a continuous timeline of the unzipping process.

The second file (“b.log” ending) is a summary of the unzipping process, showing total #sequences for each file (basically shortened version of 2a). This is more convenient for looking at entry totals. This format is the same format as the official release notes.

The third file (“c.log” ending) is a side-by-side list of all file counts, the official release notes counts and the downloaded + unzipped counts.

The fourth file (“d.log” ending) is a side-by-side list of all file division counts, the official release notes counts and the downloaded + unzipped counts. This is a summary form of c.log, with totals by division rather than file.

*python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R.py 12.1 dec.2017 poskw gb*

Runs the positive keyword screen. “dec.2017” is the date of the update, “12.1” is the version of RVDB, “poskw” is the type of screen, “gb” is the source database. Generates files ending in “pscreen” as output. All files generated as output are used as input for the sizemirna screen.

*python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py 12.1 dec.2017 sizemirna gb*

Runs the size/mirna screen. “dec.2017” is the date of the update, “12.1” is the version of RVDB, “sizemirna” is the type of screen, “gb” is the source database. Two type of files are generated as output: those ending in “FLAG” and those ending in “OK”. Files ending in “OK” pass the sizemirna screen and are used as input for the negkw screen.

*python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py 12.1 dec.2017 negkw gb*

Runs the negative keyword screen. “dec.2017” is the date of the update, “12.1” is the version of RVDB, “negkw” is the type of screen, “gb” is the source database. Generated four types of files as output” those ending in “FLAG”, those ending in “OK”, those ending in “AMB”, and those ending in “VRL”. The files ending in “OK”, “AMB” (for “ambiguous”), and “VRL” (coming from the GenBank “VRL” division) can be manually reviewed (see below, section 5) to generate the U-RVDB.

**4. Running the main pipeline – TPA.**

The main pipeline consists of unzipping the TPA files and running the positive, size/mirna, and negative screens on the unzipped TPA files.

**Main pipeline – TPA – command block.** Use the following concatenated and piped commands (described individually below).

*python F:/UPDATE\_SCRIPTS\_LOGS/VDBunzip\_tpa\_PIPE.py F: dec.2017 12.1 fsa\_nt.gz && python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py F: dec.2017 12.1 poskw tpa && python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py F: dec.2017 12.1 sizemirna tpa && python F:/UPDATE\_SCRIPTS\_LOGS/SEM-R\_PIPE.py F: dec.2017 12.1 negkw tpa*

For the positive screen, sizemirna screen, and negkw screen the output files are the same as for the GenBank pipeline, with the except that no “VRL” files are generated by the negkw screen because there is no VRL division in TPA. The files ending in “OK” and “AMB” (for “ambiguous”), can be manually reviewed (see below, section 5) to generate the U-RVDB.

**5. Manual review**

Following the running of the main pipeline, all sequences passing the SEM-R screen will be in files in the RefSeq, GenBank, and TPA directories. In the RefSeq directory, the file “viral.genomic.eukviral.fasta” can be incorporated into the final U-RVDB fasta file without any further manual review. In the GenBank folder, in the negative keyword out sub-folder (e.x. “negkw\_out\_dec.2017”, all files ending in “OK”, and “VRL” can be incorporated into the final U-RVDB fasta file without any further manual review, except for sequences that are new compared to previous versions of the RVDB (new sequences should be manually reviewed). Additionally, sequences in files ending in “AMB” (for ambiguous) should be manually reviewed before incorporating into the final U-RVDB fasta file. The goal of manual review is to identify any non-viral sequences – the accessions from such sequences should be made into a list and stored in the main update directory. This list should have one accession per line, and be named “RVDBv$version$.removeaccs.txt”, for example (“RVDBv12.0.removeaccs.txt”).

**6. Post-manual review – creation of U-RVDB fasta file**

Following manual review, the unclustered RVDB fasta file can be generated using the script “create\_U-RVDB\_file.py”. In windows cmd.exe, navigate to the working directory and enter the following command:

>python F:\UPDATE\_SCRIPTS\_LOGS\create\_U-RVDB\_file.py F: dec.2017 12.0 RVDBv12.0.removeaccs.txt