Supplementary material to: Schoonen et al. S Afr J Sci. 2019;115(3/4), Art. #4876, 3 pages

How to cite:
Schoonen M, Seyffert AS, Van der Westhuizen FH, Smuts I. A bioinformatics pipeline for rare genetic diseases in South African patients [supplementary material]. S Afr J Sci. 2019;115(3/4), Art. #4876, 10 pages. https://doi.org/10.17159/sajs.2019/4876/suppl

Appendix 1: The **vep_single.sh** script

Script that takes an Ion Torrent Variant Caller Format file and annotates it using Ensembl's offline VEP runner

```
#!/usr/bin/env bash
# This script takes an Ion Torrent VCF file and annotates it using Ensembl VEP
# Example: bash vep_single.sh /path/to/vcf/input.vcf.gz /path/to/vcf/input.vcf

# NOTE: Filenames starting with "tmp_" indicate temporary files that are deleted

# ==== Preparation ====
num_forks=4
# Directory where VEP is installed. [This may differ for your installation]
vep_dir="${HOME}/scripts/ensembl-vep"

# ---- VCF annotation fields ----
# Specify the .vcf annotation fields. They can be added in groups as below.
vep_fields="Consequence,Codons,Amino_acids,Gene,SYMBOL,Feature,EXON"
vep_fields="${vep_fields},PolyPhen,SIFT,check_alleles"

# ---- VEP arguments ----
# Specify the arguments to pass to VEP. They can be added in groups as below.
# NOTE: There is a space after "${vep_args}" in each additional group.
vep_args="--cache --offline --format vcf --vcf --buffer_size 25000"
```

```
vep_args="${vep_args} --sift b --polyphen b --symbol --numbers"
vep_args="${vep_args} --af_1kg --pubmed"
vep_args="${vep_args} --fields ${vep_fields}"

# ---- Handle arguments ----
in_file=$1                          # Path to input .vcf.gz file
out_file=$2                         # Path to output .vcf file

# ==== Run VEP ====
cp ${in_file} ${vep_dir}/tmp_input.vcf.gz
cd ${vep_dir}

echo "... running Ensembl VEP..."
./vep -i tmp_input.vcf.gz -o tmp_output.vcf -fork ${num_forks} ${vep_args}

# ---- Cleanup ----
rm tmp_input.vcf.gz
cd -
mv ${vep_dir}/tmp_output.vcf ${out_file}
mv ${vep_dir}/tmp_output.vcf_summary.html ${out_file}_summary.html
```

Appendix 2: The **gemini_single.sh** script

The annotated VEP file generated from Ensembl VEP runner is loaded into a SQLite database (GEMINI) for querying.

```
#!/usr/bin/env bash
# This script takes a VEP-annotated VCF file and loads it into a SQLite
#   database for querying. It produces a set of variant lists, as well as a
#   meta_info.txt file in the output directory.
# Example: bash gemini_single.sh /path/to/vcf/input.vcf /path/to/output/dir/

# ==== Preparation ====
# Specify the number of CPU cores GEMINI can use, and the queries_spec file
num_cores=4
qfile="../../queries_spec.txt"

# ---- GEMINI query columns ----
# Define columns of interest. They can be added in groups as below.
cols="chrom, start, end, gene, exon, ref, alt, qual, type, cyto_band"
cols="${cols}, is_coding, codon_change, aa_change"
cols="${cols}, num_hom_ref, num_hom_alt, num_het"
cols="${cols}, impact, impact_severity, is_lof, is_conserved"
cols="${cols}, depth, qual_depth, rs_ids, in_omim, pfam_domain"
cols="${cols}, clinvar_sig, clinvar_origin, clinvar_disease_name"
cols="${cols}, polyphen_pred, polyphen_score, sift_pred, sift_score"
cols="${cols}, in_hm3, in_esp, in_1kg, aaf_1kg_all, aaf_1kg_eur, aaf_1kg_afr"

# ---- GEMINI load arguments ----
gemini_args="--skip-gerp-bp"

# ---- Handle arguments ----
in_file=$1              # Annotated .vcf input file
```

```
out_dir=$2                # Output directory for query results files

# ==== Run GEMINI ====
# ---- Load into SQLite database ----
cp ${in_file} tmp_vcf.vcf
mkdir tmp_out_dir

gemini load -v tmp_vcf.vcf -t VEP --cores ${num_cores} ${gemini_args} tmp_gS.db

# ---- Query database ----
echo "Querying..."
while read line
do
   qname=$(echo ${line} | cut -d "," -f 1)
   qsnip=$(echo ${line} | cut -d "," -f 2)
   query="select ${cols} from variants where ${qsnip}"

   printf "... ${qname}...\n"
   echo gemini query -q "${query}" tmp_gS.db > tmp_out_dir/${qname}.txt
   printf "done\n"
done < ${qfile}

# ---- Generate meta_info.txt ----
for in_file in tmp_out_dir/*.txt; do
   wc -l ${in_file} >> tmp_out_dir/meta_info.txt

# ---- Move files to out_dir ----
if [ -d ${out_dir}/queries ]; then
   echo "Clobbering contents of queries folder..."
   rm -r ${out_dir}/queries/*
else
```

```
    echo "Creating queries folder"
    mkdir ${out_dir}/queries
fi

# NOTE: This works because meta_info.txt gets moved _before_ the rest...
mv ./tmp_out_dir/meta_info.txt ${output_dir}
mv ./tmp_out_dir/* ${out_dir}/queries/

# ---- Cleanup ----
rm ./tmp_gS.db ./tmp_vcf.vcf
rmdir tmp_out_dir
```

Appendix 3: The **queries_spec.txt** text file

Text file listing the queries GEMINI should make; called in the **gemini_single.sh** script. Queries can easily be removed or added.

```
LoF,is_lof = 1
RareNovelLoF,is_lof = 1 and (in_dbsnp = 0 or aaf <= 0.01)
omimClinivar,in_omim = 1 or clinvar_disease_name is not NULL
siftPolyphen,sift_pred = 'deleterious' or polyphen_pred = 'probably_damaging'
allVariants,
coding,is_coding = 1
nonCoding,is_coding = 0
codingSynonymous,is_coding = 1 and impact LIKE 'synonymous_coding'
codingNonSynonymous,is_coding = 1 and impact LIKE 'non_syn_coding'
codingStopgain,is_coding = 1 and impact LIKE 'stop_gain'
codingStoploss,is_coding = 1 and impact LIKE 'stop_loss'
codingFrameshift,is_coding = 1 and impact LIKE 'frame_shift'
reported,in_dbsnp = 1
reportedLoF,is_lof = 1 and in_dbsnp = 1
reportedNonLoF,is_lof = 0 and in_dbsnp = 1
reportedCoding,in_dbsnp = 1 and is_coding = 1
reportedCodingLoF,is_lof = 1 and in_dbsnp = 1 and is_coding = 1
reportedCodingNonLoF,is_lof = 0 and in_dbsnp = 1 and is_coding = 1
novel,in_dbsnp = 0
novelLoF,is_lof = 1 and in_dbsnp = 0
novelNonLoF,is_lof = 0 and in_dbsnp = 0
novelCoding,in_dbsnp = 0 and is_coding = 1
novelCodingLoF,is_lof = 1 and in_dbsnp = 0 and is_coding = 1
novelCodingNonLoF,is_lof = 0 and in_dbsnp = 0 and is_coding = 1
```

Appendix 4: The **vep_batch.sh** script

Files are annotated in a batch matter, where each file (*.vcf.gz*) in a folder is annotated.

```
#!/usr/bin/env bash
# This script is a wrapper for vep_single.sh which calls it for each .vcf.gz
#   file in the input directory. It outputs a VEP-annotated VCF file for each
#   input file in the output directory.
# Example: bash vep_batch.sh /path/to/input/dir/ /path/to/output/dir/
# NOTE: Absolute paths are safest since they're unambiguous

# ---- Handle arguments ----
in_dir=$1            # Path to input directory
out_dir=$2           # Path to output directory

# ---- Run VEP for batch ----
for in_file in ${in_dir}/*.vcf.gz
do
   echo "Processing ${in_file} [VEP]..."
   out_file_name="$(basename ${in_file} .gz)"
   bash vep_single.sh ${in_file} ${out_dir}/${out_file_name}
done
```

Appendix 5: The **gemini_batch.sh** script

Files are queried in a batch matter, where each file (*.vcf*) in a folder is queried.

```
#!/usr/bin/env bash
# This script is a wrapper for gemini_single.sh which calls it for each .vcf
#   file in the input directory. See that script for output details.
# Example: bash gemini_batch.sh /path/to/input/dir/ path/to/output/dir/
# NOTE: Absolute paths are safest since they're unambiguous

in_dir=$1               # Path to input directory
out_dir=$2              # Path to output directory

# ---- Run GEMINI for batch ----
for in_file in ${in_dir}/*.vcf
do
   echo "Processing ${in_file} [GEMINI]..."

   out_subdir="$(basename ${in_file} .vcf)"

   # Create output subdirectory (each file gets its own)
   out_dir_full=${out_dir}/${out_subdir}
   mkdir ${out_dir_full}

   bash gemini_single.sh ${in_file} ${out_dir_full}
done
```

Appendix 6: The **hetero_annotate.sh** script

This script uses the vep_single.sh and gemini_single.sh scripts (via their batch wrappers) to build a batch-capable variant annotation and filtering pipeline.

```
#!/usr/bin/env bash
# This script uses the vep_single.sh and gemini_single.sh scripts (via their
#   batch wrappers) to build a batch-capable variant annotation and filtering
#   pipeline. The underlying tools are the offline Ensembl VEP runner and GEMINI
# This script should ideally be used on patient folders in a batch fashion.
# Example: bash hetero_annotate.sh /path/to/input/dir/ /path/to/output/dir/
# NOTE: Absolute paths are safest since they're unambiguous

# ---- Handle arguments ----
in_dir=$1             # Path to input directory
out_dir=$2            # Path to output directory

# ---- Prepare out_dir ----
# Check whether out_dir exists (1) and is empty (2)
if ! [ -d ${out_dir} ]; then
   echo "Creating main output directory: ${out_dir}..."
   mkdir ${out_dir}
else
   if ! [ -z "$(ls -A ${out_dir})" ]; then
      echo "Empty main output directory first..."
      exit 1
   fi
fi
mkdir ${out_dir}/vep_out

# ---- Run VEP ----
cd src/vep_scripts
```

```
bash vep_batch.sh ${in_dir} ${out_dir}/vep_out
cd -

# ---- Run GEMINI ----
cd src/gemini_scripts
bash gemini_batch.sh ${out_dir}/vep_out ${out_dir}
cd -
```