UoS-HGIG / GenePy-1.4 Public

Latest GenePy version using Ensembl VEP annotation, gnomAD random forest flag. Incorporation of tri-allelic variants *in progress*

5 stars 0 forks

Star Notifications

Code Issues 1 Pull requests Actions Projects Security Insights

main Go to file

isstafford ... on Sep 16, 2021

View code

README.md

GenePy-1.4

Latest GenePy version using Ensembl VEP annotation, gnomAD random forest flag. Incorporation of tri-allelic variants *in progress*

GenePy v1.4 implements the following improvements from v1.3

- VCF annotation of allele frequencies and CADD is now completed through Ensembl-VEP, which is maintained and updated more consistently than ANNOVAR.
- CADD v.1.6 (also known as CADD-Splice), which improves the scoring of splicing variants in comparison to previous release CADD v.1.5
- Implementation of gnomAD v.3.1.1 Random Forest (RF) flag as an additional variant quality filter.

GenePy Requirements

- A (multi)sample VCF file (can accept compressed vcf.gz)
- List of genes to generate GenePy scores (gene.list)
- List of variant types to include (variant_type.list)
- Ensembl-VEP installed (https://www.ensembl.org/info/docs/tools/vep/script/vep_download.html#new)

- Ensembl-VEP CADD Plugin
- GnomAD v.2.1.1 (exomes) sites vcf
- CADD v.1.6 Installed, with tsv files: whole_genome_SNVs and gnomad genomes indels (https://cadd.gs.washington.edu/download)
- Optional: ANNOVAR with gnomAD RF flag annotation files
- Vcftools
- Python 2.7.x

Pre-processing for GenePy Scores

Prior to running the GenePy script, we need to annotate our VCF file, and filter accordingly to create the appropriate input file (ALL_genepy.meta)

Bi-allelic variants only

We want to filter to include only bi-allelic variants

```
# Keep bi-allelic variants
vcftools --gzvcf GENOTYPED_ALL.vcf.gz --min-alleles 2 --max-alleles 2 --recode --
remove-filtered-all --out FINAL
```

Annotation

To run Ensembl-VEP, remember to softlink the location of the VEP software to your home directory

```
ln -s /software/location/VEP /your/home/directory/.vep
```

Annotate with VEP and ANNOVAR (optional)

```
# The VEP annotation script will perform the annotation of chromosomes in parallel
to speed-up this process
grep "^#" FINAL.recode.vcf > vcf_header
grep -v "^#" FINAL.recoded.vcf > FINAL_noheader.recode.vcf

sbatch --array=1-22 vep.sh FINAL_noheader.recode.vcf
sbatch vep_x.sh FINAL_noheader.recode.vcf

# Reassemble the annotated vcf
seq 1 22 > chr
grep "^#" ANNO_CHR1.vcf > vcf_header
while read l; do grep -v "^#" ANNO_CHR${l}.vcf >> temp_vcf; done < chr
cat vcf_header temp_vcf > FINAL_ANNOTATION.vcf
```

```
# Annovar annotation
sbatch annotation.sh
```

Our FINAL_ANNOTATION.vcf requires some manipulation to convert it into the correct input for GenePy. In addition, VEP annotation leaves some variants without a CADD score, so these need to be annotated online (https://cadd.gs.washington.edu/score), and the scores inserted into our annotated vcf. Name your online annotations extra_CADDV1.6_hg38_online.tsv and remove the header in the file generated by online CADD.

```
grep -v "^#" FINAL_ANNOTATION.vcf | cut -f 10- > genotypes #Alter columns here if
you want GenePy scores on a subset of your cohort
grep -v "^#" FINAL_ANNOTATION.vcf | cut -f 1,2,4,5 > variants
grep -v "^#" FINAL_ANNOTATION.vcf | cut -f 8 > info_temp

# CADD gap fill

awk -F '|' '{print match($7, /[^ ]/) ? $7 : "99999999"}' info_temp > rawscore
#Column 7 should be the CADD rawscore

# Create master input file for combine_annotations.py

cut -f 1,2 variants > b1
paste b1 rawscore > master_rawscore.txt
sed -i 's/^chr//g' master_rawscore.txt
rm b1

#Run combine_annotations.py

python combine_annotations.py > rawscore_annotations.txt

# Format info_temp

sed -e "s/^|/.|/" -e "s/||/|.|/g" -e "s/|$/|./" -i info_temp
sed -i 's/||/|.|/g' info_temp
sed -i 's/|/\t/g' info_temp

# Select columns for GenePy input
# The columns you should have are as follows: Consequence, SYMBOL, gnomadE_AF

cut -f 2,3,5 info_temp > a1

# Select RF flag column from ANNOVAR output

cut -f 28 ALL_genepy.hg38_multianno.txt > RF
tail -n +2 RF > a3

# Compile Columns
```

```
cut -f 3 rawscore_annotations.txt > a2
paste variants a1 a2 a3 genotypes > annotation_body.txt
rm variants a1 a2 a3 genotypes
```

At this stage, you should have a complete file with the following columns: Chr, Start, Ref, Alt, Consequence, Gene, AF, Rawscore, RF (if using). This is followed by the gnotypes of the individuals in the cohort. You will need to create your own tab-delimited header for this file.

```
cat header annotation_body.txt > FULL_ANNOTATION
```

Final Filtering

GnomAD RF flag filtering. Filter to only include variants that PASS or have no information ("."). You must remove this RF column after filtering.

```
awk 'BEGIN{ FS=OFS="\t" }{ if(($9 ~ '/PASS/' || $9 ==".")) { print $0 } }'
FULL_ANNOTATION.txt > FULL_ANNOTATION_RF.txt
cut -f 1-8,10- FULL_ANNOTATION_RF.txt > ALL_genepy.meta
head -n 1 ALL_genepy.meta > header
```

Variant filtering. Currently, we use only exonic and splicing variants in the GenePy score. The list of variant types can be changed depending on what you would like to include. The full list of Ensembl VEP consequence types can be found in VEP_consequence.list (v.104). The list of variants we considered exonic and splicing can be found in variant_type.list

```
# Store the variant types you want to select for in file variant_type.list

awk -F '\t' 'NR==FNR{c[$1]++;next};c[$5] > 0' exonic_variants ALL_genepy.meta >
temp
cat header temp > ALL_genepy_exonic.meta
rm temp

#Fix VEP formatting issue
sed -i 's/|/\//g' ALL_genepy_exonic.meta

# Potential VEP formatting issue fix
# If VEP has recorded a heterozygous genotype in the following way: 1/0, GenePy
will not recognise this as a genotype. You can switch the formatting like os
sed -i 's/1\/0/0\/1/g' ALL_genepy_exonic.meta
```

Note: this header file is a dependent file when generating GenePy scores, do not delete this file.

Compute GenePy Scores

Once ALL_genepy_exonic.meta file is created, GENEPY_1.3.sh can be run by iterating through a list of desired genes. Note: the make_scores_mat_6.py file **must** be in the same directory as GENEPY_1.3.sh.

```
mkdir CADD15_RAW

cut -f 7 ALL_genepy.hg38_multianno.txt | grep -v ";" | grep -v "Gene.refGene" |
sort | uniq >gene.list

#Run GenePy
while read gene;
do sh GENEPY_1.3.sh $gene;
done< gene.list
```

We can use the subber.sh script to parallelise the generation of GenePy scores as follows

```
# Split the gene list in batches of 400 genes
split -d -l 400 gene.list batch_
#put all the batch names in a file
ls batch_* >parts
#check how many batches we have (52 in my case)
wc -l parts
#submit the job array
sbatch --array=1-52 subber.sh
```

Finally, create your individual x gene matrix using the MatrixMaker.sh script

```
sh MatrixMaker.sh CADD15_RAW/ $nameyourmatrix
```

You now have a GenePy matrix ready for any downstream analysis 😁

Releases

No releases published

Packages

No packages published

Languages


Python 64.1%
Shell 35.9%