TransVar Documentation

Release 2.5.4

Wanding Zhou

Apr 24, 2019

Contents

Contents:

CHAPTER 1

Download and Install

1.1 Install using pip

```
sudo pip install transvar
```

or locally

```
pip install --user transvar
```

to upgrade from a previous version

```
pip install -U transvar
```

1.2 Use the docker images

The pre-built docker image is easy to try out. The docker images can be found here

Assuming the existence of *~/references/hg38/hg38.fa* and *~/references/hg38/hg38.fa.fai*.

Without downloading anything, the transvar docker has pre-built hg38 annotation.

```
docker run -v ~/references/hg38:/ref zhouwanding/transvar:latest
   transvar panno -i PIK3CA:p.E545K --ensembl --reference /ref/hg38.fa
```

To use other genome build, one needs to download annotations. Here I am using *~/test* as an example of local path for storing the transvar annotations. Note that this local path needs be imaged to */anno* inside the docker image. This is done by (showing hg19):

```
docker run -v ~/test:/anno zhouwanding/transvar:latest
   transvar config --download_anno --refversion hg19 --skip_reference
```

Now one can use hg19, but note again one needs to image the path of downloaded annotation to */anno*. One also needs the fa-indexed reference.

```
docker run -v ~/test:/anno -v ~/references/hg19:/ref
   zhouwanding/transvar:latest transvar panno -i PIK3CA:p.E545K
   --ensembl --reference /ref/hg19.fa
```

1.3 Download the program

1.3.1 Current release

Latest release is available here

For all previous versions, see here

1.3.2 Other old stable releases

- stable 2.0.x version v2.0.12.20150626
- stable 1.x version v1.40

1.4 Dependency

The only requirement for building TransVar are Python 2.7 and a reasonably modern C compiler such as gcc.

1.5 Install from source

1.5.1 Local install

```
python setup.py install --prefix [folder]
```

The installation will create two subfolders: `[folder]/lib` (which would contain libraries) and `[folder]/bin` (which would contain transvar executable).

When you run transvar, make sure `[folder]/lib/python2.7/site-packages` is in your PYTHONPATH. In some occasions, you need to `mkdir -p [folder]/lib/python2.7/site-packages` to make sure it exists before you could run `setup.py`. You can add it by putting

```
export PYTHONPATH=$PYTHONPATH:[folder]/lib/python-2.7/site-packages/
```

to your `.bashrc` or `.profile` depending on your OS.

The installed executable is **[folder]/bin/transvar**.

1.5.2 System-wise install (need root)

```
sudo python setup.py install
```

CHAPTER 2

Quick Start

Here we show how one can use TransVar on human hg19 (GRCh37).

```
# set up databases
transvar config --download_anno --refversion hg19

# in case you don't have a reference
transvar config --download_ref --refversion hg19

# in case you do have a reference to link
transvar config -k reference -v [path_to_hg19.fa] --refversion hg19
```

Test an input:

```
$ transvar panno -i 'PIK3CA:p.E545K' --ucsc --ccds
```

outputs show two hits from the two databases, i.e., UCSC and CCDS.

```
PIK3CA:p.E545K       NM_006218 (protein_coding)      PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_10]
   CSQN=Missense;reference_codon=GAG;candidate_codons=AAG,AAA;candidate_mnv_vari
   ants=chr3:g.178936091_178936093delGAGinsAAA;dbsnp=rs104886003(chr3:178936091G
   >A);source=UCSCRefGene
PIK3CA:p.E545K       CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_9]
   CSQN=Missense;reference_codon=GAG;candidate_codons=AAG,AAA;candidate_mnv_vari
   ants=chr3:g.178936091_178936093delGAGinsAAA;dbsnp=rs104886003(chr3:178936091G
   >A);source=CCDS
```

One could provide input based on transcript ID, e.g **NM_006218.1:p.E545K** and TransVar would automatically restrict to the provided transcript.

```
$ transvar panno -i 'NM_006218.2:p.E545K' --ucsc --ccds
```

outputs

```
NM_006218.2:p.E545K  NM_006218 (protein_coding)      PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_10]
   CSQN=Missense;reference_codon=GAG;candidate_codons=AAG,AAA;candidate_mnv_vari
   ants=chr3:g.178936091_178936093delGAGinsAAA;dbsnp=rs104886003(chr3:178936091G
   >A);source=UCSCRefGene
```

CHAPTER 3

Setup and Customize

3.1 Use environment variables

3.1.1 TRANSVAR_CFG

store the path to transvar.cfg

```
export TRANSVAR_CFG=path_to_transvar.cfg
```

If not specified, TransVar will use **[installdir]/lib/transvar/transvar.cfg** directory or your local **~/.transvar.cfg** if the installation directory is inaccessible.

3.1.2 TRANSVAR_DOWNLOAD_DIR

store the path to the directory where auto-download of annotation and reference go

```
export TRANSVAR_DOWNLOAD_DIR=path_to_transvar_download_directory
```

If not specified, TransVar will use **[installdir]/lib/transvar/transvar.download** directory or your local **~/.transvar.download** if the installation directory is inaccessible.

3.2 Install and specify reference genome assembly

3.2.1 Download from TransVar database

For some genome assembly (currently hg18, hg19, hg38, mm9 and mm10) we provide download via

```
transvar config --download_ref --refversion [reference name]
```

See `transvar config -h` for all choices of `[reference name]`).

3.2.2 Manual download and index

For other genome assemblies, one could manually download the genome as one file and index it manually by,

```
samtools faidx [fasta]
```

Once downloaded and indexed, the genome can be used through the "–reference" option followed by path to the genome:

```
transvar ganno -i "chr1:g.30000000_30000001" --gencode --reference path_to_hg19.fa
```

or "–refversion" followed by the short version id.

```
transvar ganno -i "chr1:g.30000000_30000001" --gencode --refversion hg19
```

One can store the location in *transvar.cfg* file. To set the default location of genome file for a reference version, say, to path_to_hg19.fa,

```
transvar config -k reference -v path_to_hg19.fa --refversion hg19
```

will create in transvar.cfg an entry

```
[hg19]
reference = hg19.fa
```

so that there is no need to specify the location of reference on subsequent usages.

3.3 Install and specify transcript annotations

3.3.1 Download from TransVar database

One could automatically download transcript annotations via E.g.,

```
transvar config --download_anno --refversion hg19
```

which download annotation from TransVar database to **[installdir]/lib/transvar/transvar.download** directory or your local **~/.transvar.download** if the installation directory is inaccessible. See **transvar config -h** for all version names. These will also create default mappings under the corresponding reference version section of **transvar.cfg** like

```
[hg19]
ucsc = /home/wzhou1/download/hg19.ucsc.txt.gz
```

3.3.2 Index from GTF files

TransVar databases can be obtained from indexing a GTF file. For example,

```
transvar index --refseq hg38.refseq.gff.gz
```

The above will create a bunch of transvar databaase files with the suffix hg38.refseq.gff.gz.transvardb*.

3.3.3 Download from Ensembl ftp

One also has the option of downloading from Ensembl collection.

```
transvar config --download_ensembl --refversion mus_musculus
```

Without specifying the refversion, user will be prompted a collection of options to choose from.

3.4 Know Current configuration

To show the location and the content of currently used transvar.cfg, one may also run

```
transvar config
```

which returns information about the setup regarding to the current reference selection, including the location of the reference file and database file.

```
Current reference version: mm10
reference: /home/wzhou/genomes_link/mm10/mm10.fa
Available databases:
refseq: /home/wzhou/tools/transvar/transvar/transvar.download/mm10.refseq.gff.gz
ccds: /home/wzhou/tools/transvar/transvar/transvar.download/mm10.ccds.txt
ensembl: /home/wzhou/tools/transvar/transvar/transvar.download/mm10.ensembl.gtf.gz
```

specifying `--refversion` displays the information under that reference version (without changing the default reference version setup).

3.5 Set default reference builds

To switch reference build

```
transvar config --switch_build mm10
```

switches the default reference build to mm10. This is equivalent to

```
transvar config -k refversion -v mm10
```

which sets the refversion slot explicitly.

3.6 Use Additional Resources

TransVar uses optional additional resources for annotation.

3.6.1 dbSNP

For example, one could annotate SNP with dbSNP id by downloading the dbSNP files. This can be done by

```
transvar config --download_dbsnp
```

TransVar automatically download dbSNP file which correspoding to the current default reference version (as set in **transvar.cfg**). This also sets the entry in **transvar.cfg**. With dbSNP file downloaded, TransVar automatically looks for dbSNP id when performing annotation.

```
transvar panno -i 'A1CF:p.A309A' --ccds
```

```
A1CF:p.A309A CCDS7243 (protein_coding)       A1CF    -
   chr10:g.52576004T>G/c.927A>C/p.A309A      inside_[cds_in_exon_7]
   CSQN=Synonymous;reference_codon=GCA;candidate_codons=GCC,GCG,GCT;candidate_sn
   v_variants=chr10:g.52576004T>C,chr10:g.52576004T>A;dbsnp=rs201831949(chr10:52
   576004T>G);source=CCDS
```

Note that in order to use dbSNP, one must download the dbSNP database through

```
transvar config --download_dbsnp
```

or by configure the `dbsnp` slot in the configure file via

```
transvar config -k dbsnp -v [path to dbSNP VCF]
```

Manually set path for dbSNP file must have the file tabix indexed.

3.7 Control the length of reference sequence

TransVar reduces the reference sequence in a deletion to its length when the deleted reference sequence is too long. For example

```
$ transvar ganno -i 'chr14:g.101347000_101347023del' --ensembl
```

outputs

```
chr14:g.101347000_101347023del       ENST00000534062 (protein_coding)       RTL1    -
   chr14:g.101347000_101347023del24/c.4074+29_4074+52del24/. inside_[3-UTR;noncoding_
↪exon_1]
   CSQN=3-UTRDeletion;left_align_gDNA=g.101347000_101347023del24;unaligned_gDNA=
   g.101347000_101347023del24;left_align_cDNA=c.4074+29_4074+52del24;unalign_cDN
   A=c.4074+29_4074+52del24;aliases=ENSP00000435342;source=Ensembl
```

where the deletion sequence is reduced to its length (*del24*). The *–seqmax* option changes the length threshold (default:10) when this behavior occur. When *–seqmax* is negative, the threshold is lifted such that the reference sequence is always reported regardless of its length, i.e.,

```
$ transvar ganno -i 'chr14:g.101347000_101347023del' --ensembl --seqmax -1
```

outputs the full reference sequence:

```
chr14:g.101347000_101347023del       ENST00000534062 (protein_coding)       RTL1    -
   chr14:g.101347000_101347023delTTGGGGTGAGAAATAGAGGGGACT/c.4074+29_
↪4074+52delAGTCCCCTCTATTTCTCACCCCAA/.     inside_[3-UTR;noncoding_exon_1]
   CSQN=3-UTRDeletion;left_align_gDNA=g.101347000_101347023delTTGGGGTGAGAAATAGAG
   GGGACT;unaligned_gDNA=g.101347000_101347023delTTGGGGTGAGAAATAGAGGGGACT;left_a
   lign_cDNA=c.4074+29_4074+52delAGTCCCCTCTATTTCTCACCCCAA;unalign_cDNA=c.4074+29
   _4074+52delAGTCCCCTCTATTTCTCACCCCAA;aliases=ENSP00000435342;source=Ensembl
```

CHAPTER 4

Genomic level annotation

Annotation from genomic level is handled by the *ganno* subcommand in TransVar.

4.1 Short genomic regions

To annotate a short genomic region in a gene,

```
$ transvar ganno --ccds -i 'chr3:g.178936091_178936192'
```

outputs

```
chr3:g.178936091_178936192   CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091_178936192/c.1633_1664+70/p.E545_R555     from_[cds_in_exon_9]_to_
↪[intron_between_exon_9_and_10]
   C2=donor_splice_site_on_exon_9_at_chr3:178936123_included;start_codon=1789360
   91-178936092-178936093;end_codon=178936121-178936122-178936984;source=CCDS
```

Results indicates the beginning position is at coding region while ending position is at intronic region (c.1633_1664+70). Note that there is no consequence label (*CSQN* tag) when performing a region annotation (instead of a variant).

For intergenic sites, TransVar also reports the identity and distance to the gene upstream and downstream. For example, *chr6:116991832* is simply annotated as intergenic in the original annotation. TransVar reveals that it is 1,875 bp downstream to ZUFSP and 10,518 bp upstream to KPNA5 showing a vicinity to the gene ZUFSP. There is no limit in the reported distance. If a site is at the end of the chromosome, TransVar is able to report the distance to the telomere.

4.2 Long genomic regions

```
$ transvar ganno -i 'chr19:g.41978629_41983350' --ensembl --refversion mm10
```

```
chr19:g.41978629_41983350    ENSMUST00000167927 (nonsense_mediated_decay),
↪ENSMUST00000026170 (protein_coding)        MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
chr19:g.41978629_41983350    ENSMUST00000171561 (protein_coding),ENSMUST00000026170␣
↪(protein_coding) MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
chr19:g.41978629_41983350    ENSMUST00000163398 (nonsense_mediated_decay),
↪ENSMUST00000026170 (protein_coding)        MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
chr19:g.41978629_41983350    ENSMUST00000164776 (nonsense_mediated_decay),
↪ENSMUST00000026170 (protein_coding)        MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
chr19:g.41978629_41983350    ENSMUST00000026168 (protein_coding),ENSMUST00000026170␣
↪(protein_coding) MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
chr19:g.41978629_41983350    ENSMUST00000171755 (retained_intron),ENSMUST00000026170␣
↪(protein_coding)        MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
chr19:g.41978629_41983350    ENSMUST00000169775 (nonsense_mediated_decay),
↪ENSMUST00000026170 (protein_coding)        MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
chr19:g.41978629_41983350    ENSMUST00000168484 (nonsense_mediated_decay),
↪ENSMUST00000026170 (protein_coding)        MMS19,UBTD1     -,+
   chr19:g.41978629_41983350/./.     from_[intron_between_exon_1_and_2;MMS19]_to_
↪[intron_between_exon_1_and_2;UBTD1]
   .
```

Results indicates a 4721 bp region spanning the promoters of two closely located, opposite-oriented genes MMS19 and UBTD1. The starting point and ending point are situated in the first introns of the two genes.

```
$ transvar ganno -i '9:g.133750356_137990357' --ccds
```

outputs

```
9:g.133750356_137990357      CCDS35165.1 (protein_coding),CCDS6986.1 (protein_coding)␣
↪       ABL1,OLFM1      +,+
   chr9:g.133750356_137990357/./.    from_[cds_in_exon_7;ABL1]_to_[intron_between_
↪exon_4_and_5;OLFM1]_spanning_[51_genes]
   .
9:g.133750356_137990357      CCDS35166.1 (protein_coding),CCDS6986.1 (protein_coding)␣
↪       ABL1,OLFM1      +,+
   chr9:g.133750356_137990357/./.    from_[cds_in_exon_7;ABL1]_to_[intron_between_
↪exon_4_and_5;OLFM1]_spanning_[51_genes]
   .
```

The result indicates that the region span 53 genes. The beginning of the region resides in the coding sequence of ABL1, c.1187A and the ending region resides in the intronic region of OLFM1, c.622+6C. 2 different usage of transcripts in annotating the starting position is represented in two lines, each line corresponding to a combination of transcript usage. This annotation not only shows the coverage of the region, also reveals the fine structure of the boundary.

In another example, where the ending position exceeds the length of the chromosome, TransVar truncates the region and outputs upstream and downstream information of the ending position.

```
$ transvar ganno -i '9:g.133750356_1337503570' --ccds
```

outputs

```
9:g.133750356_1337503570     CCDS35165.1 (protein_coding),   ABL1,   +
   chr9:g.133750356_141213431/./.    from_[cds_in_exon_7;ABL1]_to_[intergenic_between_
↪EHMT1(484,026_bp_downstream)_and_3'-telomere(0_bp)]_spanning_[136_genes]
   .
9:g.133750356_1337503570     CCDS35166.1 (protein_coding),   ABL1,   +
   chr9:g.133750356_141213431/./.    from_[cds_in_exon_7;ABL1]_to_[intergenic_between_
↪EHMT1(484,026_bp_downstream)_and_3'-telomere(0_bp)]_spanning_[136_genes]
   .
```

4.3 Genomic variant

4.3.1 Single nucleotide variation (SNV)

This is the forward annotation

```
$ transvar ganno --ccds -i 'chr3:g.178936091G>A'
```

outputs

```
chr3:g.178936091G>A  CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_9]
   CSQN=Missense;dbsnp=rs104886003(chr3:178936091G>A);codon_pos=178936091-178936
   092-178936093;ref_codon_seq=GAG;source=CCDS
```

Another example:

```
$ transvar ganno -i "chr9:g.135782704C>G" --ccds
```

outputs

```
chr9:g.135782704C>G  CCDS55350.1 (protein_coding)    TSC1    -
   chr9:g.135782704C>G/c.1164G>C/p.L388L     inside_[cds_in_exon_10]
   CSQN=Synonymous;dbsnp=rs770692313(chr9:135782704C>G);codon_pos=135782704-1357
   82705-135782706;ref_codon_seq=CTG;source=CCDS
chr9:g.135782704C>G  CCDS6956.1 (protein_coding)     TSC1    -
   chr9:g.135782704C>G/c.1317G>C/p.L439L     inside_[cds_in_exon_11]
   CSQN=Synonymous;dbsnp=rs770692313(chr9:135782704C>G);codon_pos=135782704-1357
   82705-135782706;ref_codon_seq=CTG;source=CCDS
```

and a nonsense mutation:

```
$ transvar ganno -i 'chr1:g.115256530G>A' --ensembl
```

outputs

```
chr1:g.115256530G>A  ENST00000369535 (protein_coding)        NRAS    -
   chr1:g.115256530G>A/c.181C>T/p.Q61*       inside_[cds_in_exon_3]
   CSQN=Nonsense;codon_pos=115256528-115256529-115256530;ref_codon_seq=CAA;alias
   es=ENSP00000358548;source=Ensembl
```

CSQN fields indicates a nonsense mutation.

4.3.2 Deletions

A frameshift deletion

```
$ transvar ganno -i "chr2:g.234183368_234183380del" --ccds
```

outputs

```
chr2:g.234183368_234183380del        CCDS2502.2 (protein_coding)     ATG16L1 +
   chr2:g.234183368_234183380del13/c.841_853del13/p.T281Lfs*5        inside_[cds_in_
↪exon_8]
   CSQN=Frameshift;left_align_gDNA=g.234183367_234183379del13;unaligned_gDNA=g.2
   34183368_234183380del13;left_align_cDNA=c.840_852del13;unalign_cDNA=c.841_853
   del13;source=CCDS
chr2:g.234183368_234183380del        CCDS2503.2 (protein_coding)     ATG16L1 +
   chr2:g.234183368_234183380del13/c.898_910del13/p.T300Lfs*5        inside_[cds_in_
↪exon_9]
   CSQN=Frameshift;left_align_gDNA=g.234183367_234183379del13;unaligned_gDNA=g.2
   34183368_234183380del13;left_align_cDNA=c.897_909del13;unalign_cDNA=c.898_910
   del13;source=CCDS
chr2:g.234183368_234183380del        CCDS54438.1 (protein_coding)    ATG16L1 +
   chr2:g.234183368_234183380del13/c.409_421del13/p.T137Lfs*5        inside_[cds_in_
↪exon_5]
   CSQN=Frameshift;left_align_gDNA=g.234183367_234183379del13;unaligned_gDNA=g.2
   34183368_234183380del13;left_align_cDNA=c.408_420del13;unalign_cDNA=c.409_421
   del13;source=CCDS
```

Note the difference between left-aligned identifier and the right aligned identifier.

An in-frame deletion

```
$ transvar ganno -i "chr2:g.234183368_234183379del" --ccds
```

outputs

```
chr2:g.234183368_234183379del        CCDS2502.2 (protein_coding)     ATG16L1 +
   chr2:g.234183368_234183379del12/c.841_852del12/p.T281_G284delTHPG inside_[cds_in_
↪exon_8]
   CSQN=InFrameDeletion;left_align_gDNA=g.234183367_234183378del12;unaligned_gDN
   A=g.234183368_234183379del12;left_align_cDNA=c.840_851del12;unalign_cDNA=c.84
   1_852del12;left_align_protein=p.T281_G284delTHPG;unalign_protein=p.T281_G284d
   elTHPG;source=CCDS
chr2:g.234183368_234183379del        CCDS2503.2 (protein_coding)     ATG16L1 +
   chr2:g.234183368_234183379del12/c.898_909del12/p.T300_G303delTHPG inside_[cds_in_
↪exon_9]
   CSQN=InFrameDeletion;left_align_gDNA=g.234183367_234183378del12;unaligned_gDN
   A=g.234183368_234183379del12;left_align_cDNA=c.897_908del12;unalign_cDNA=c.89
   8_909del12;left_align_protein=p.T300_G303delTHPG;unalign_protein=p.T300_G303d
```

(continues on next page)

(continued from previous page)

```
   elTHPG;source=CCDS
chr2:g.234183368_234183379del        CCDS54438.1 (protein_coding)    ATG16L1 +
   chr2:g.234183368_234183379del12/c.409_420del12/p.T137_G140delTHPG inside_[cds_in_
↪exon_5]
   CSQN=InFrameDeletion;left_align_gDNA=g.234183367_234183378del12;unaligned_gDN
   A=g.234183368_234183379del12;left_align_cDNA=c.408_419del12;unalign_cDNA=c.40
   9_420del12;left_align_protein=p.T137_G140delTHPG;unalign_protein=p.T137_G140d
   elTHPG;source=CCDS
```

Another example

```
$ transvar ganno --ccds -i 'chr12:g.53703425_53703427del'
```

outputs

```
chr12:g.53703425_53703427del CCDS53797.1 (protein_coding)    AAAS    -
   chr12:g.53703427_53703429delCCC/c.670_672delGGG/p.G224delG        inside_[cds_in_
↪exon_7]
   CSQN=InFrameDeletion;left_align_gDNA=g.53703424_53703426delCCC;unaligned_gDNA
   =g.53703425_53703427delCCC;left_align_cDNA=c.667_669delGGG;unalign_cDNA=c.669
   _671delGGG;left_align_protein=p.G223delG;unalign_protein=p.G223delG;source=CC
   DS
chr12:g.53703425_53703427del CCDS8856.1 (protein_coding)    AAAS    -
   chr12:g.53703427_53703429delCCC/c.769_771delGGG/p.G257delG        inside_[cds_in_
↪exon_8]
   CSQN=InFrameDeletion;left_align_gDNA=g.53703424_53703426delCCC;unaligned_gDNA
   =g.53703425_53703427delCCC;left_align_cDNA=c.766_768delGGG;unalign_cDNA=c.768
   _770delGGG;left_align_protein=p.G256delG;unalign_protein=p.G256delG;source=CC
   DS
```

Note the difference between left and right-aligned identifiers on both protein level and cDNA level.

An in-frame out-of-phase deletion

```
$ transvar ganno -i "chr2:g.234183372_234183383del" --ccds
```

outputs

```
chr2:g.234183372_234183383del        CCDS2502.2 (protein_coding)    ATG16L1 +
   chr2:g.234183372_234183383del12/c.845_856del12/p.H282_G286delinsR inside_[cds_in_
↪exon_8]
   CSQN=MultiAAMissense;left_align_gDNA=g.234183372_234183383del12;unaligned_gDN
   A=g.234183372_234183383del12;left_align_cDNA=c.845_856del12;unalign_cDNA=c.84
   5_856del12;source=CCDS
chr2:g.234183372_234183383del        CCDS2503.2 (protein_coding)    ATG16L1 +
   chr2:g.234183372_234183383del12/c.902_913del12/p.H301_G305delinsR inside_[cds_in_
↪exon_9]
   CSQN=MultiAAMissense;left_align_gDNA=g.234183372_234183383del12;unaligned_gDN
   A=g.234183372_234183383del12;left_align_cDNA=c.902_913del12;unalign_cDNA=c.90
   2_913del12;source=CCDS
chr2:g.234183372_234183383del        CCDS54438.1 (protein_coding)    ATG16L1 +
   chr2:g.234183372_234183383del12/c.413_424del12/p.H138_G142delinsR inside_[cds_in_
↪exon_5]
   CSQN=MultiAAMissense;left_align_gDNA=g.234183372_234183383del12;unaligned_gDN
   A=g.234183372_234183383del12;left_align_cDNA=c.413_424del12;unalign_cDNA=c.41
   3_424del12;source=CCDS
```

4.3.3 Insertions

An in-frame insertion of three nucleotides

```
$ transvar ganno -i 'chr2:g.69741762_69741763insTGC' --ccds
```

outputs

```
chr2:g.69741762_69741763insTGC       CCDS1893.2 (protein_coding)     AAK1    -
   chr2:g.69741780_69741782dupCTG/c.1614_1616dupGCA/p.Q546dupQ       inside_[cds_in_
↪exon_12]
   CSQN=InFrameInsertion;left_align_gDNA=g.69741762_69741763insTGC;unalign_gDNA=
   g.69741762_69741763insTGC;left_align_cDNA=c.1596_1597insCAG;unalign_cDNA=c.16
   14_1616dupGCA;left_align_protein=p.Y532_Q533insQ;unalign_protein=p.Q539dupQ;p
   hase=2;source=CCDS
```

Note the proper right-alignment of protein level insertion Q. The left-aligned identifier is also given in the *LEFTALN* field.

A frame-shift insertion of two nucleotides

```
$ transvar ganno -i 'chr7:g.121753754_121753755insCA' --ccds
```

outputs

```
chr7:g.121753754_121753755insCA      CCDS5783.1 (protein_coding)     AASS    -
   chr7:g.121753754_121753755insCA/c.1064_1065insGT/p.I355Mfs*10     inside_[cds_in_
↪exon_9]
   CSQN=Frameshift;left_align_gDNA=g.121753753_121753754insAC;unalign_gDNA=g.121
   753754_121753755insCA;left_align_cDNA=c.1063_1064insTG;unalign_cDNA=c.1063_10
   64insTG;source=CCDS
```

```
$ transvar ganno -i 'chr17:g.79093270_79093271insGGGCGT' --ccds
```

outputs

```
chr17:g.79093270_79093271insGGGCGT   CCDS45807.1 (protein_coding)    AATK    -
   chr17:g.79093282_79093287dupTGGGCG/c.3988_3993dupACGCCC/p.T1330_P1331dupTP
↪inside_[cds_in_exon_13]
   CSQN=InFrameInsertion;left_align_gDNA=g.79093270_79093271insGGGCGT;unalign_gD
   NA=g.79093270_79093271insGGGCGT;left_align_cDNA=c.3976_3977insCGCCCA;unalign_
   cDNA=c.3988_3993dupACGCCC;left_align_protein=p.A1326_P1327insPT;unalign_prote
   in=p.T1330_P1331dupTP;phase=0;source=CCDS
```

Notice the difference in the inserted sequence when left-alignment and right-alignment conventions are followed.

A frame-shift insertion of one nucleotides in a homopolymer

```
$ transvar ganno -i 'chr7:g.117230474_117230475insA' --ccds
```

outputs

```
chr7:g.117230474_117230475insA       CCDS5773.1 (protein_coding)     CFTR    +
   chr7:g.117230479dupA/c.1752dupA/p.E585Rfs*4       inside_[cds_in_exon_13]
   CSQN=Frameshift;left_align_gDNA=g.117230474_117230475insA;unalign_gDNA=g.1172
   30474_117230475insA;left_align_cDNA=c.1747_1748insA;unalign_cDNA=c.1747_1748i
   nsA;source=CCDS
```

Notice the right alignment of cDNA level insertion and the left alignment reported as additional information.

A in-frame, in-phase insertion

```
$ transvar ganno -i 'chr12:g.109702119_109702120insACC' --ccds
```

```
chr12:g.109702119_109702120insACC    CCDS31898.1 (protein_coding)    ACACB   +
   chr12:g.109702119_109702120insACC/c.6870_6871insACC/p.Y2290_H2291insT     inside_
↪[cds_in_exon_49]
   CSQN=InFrameInsertion;left_align_gDNA=g.109702118_109702119insCAC;unalign_gDN
   A=g.109702119_109702120insACC;left_align_cDNA=c.6869_6870insCAC;unalign_cDNA=
   c.6870_6871insACC;left_align_protein=p.Y2290_H2291insT;unalign_protein=p.Y229
   0_H2291insT;phase=0;source=CCDS
```

4.3.4 Block substitutions

A block-substitution that results in a frameshift.

```
$ transvar ganno -i 'chr10:g.27329002_27329002delinsAT' --ccds
```

```
chr10:g.27329002_27329002delinsAT    CCDS41499.1 (protein_coding)    ANKRD26 -
   chr10:g.27329009dupT/c.2266dupA/p.M756Nfs*6       inside_[cds_in_exon_21]
   CSQN=Frameshift;left_align_gDNA=g.27329002_27329003insT;unalign_gDNA=g.273290
   02_27329003insT;left_align_cDNA=c.2259_2260insA;unalign_cDNA=c.2266dupA;sourc
   e=CCDS
```

A block-substitution that is in-frame,

```
$ transvar ganno -i 'chr10:g.52595929_52595930delinsAA' --ccds
```

```
chr10:g.52595929_52595930delinsAA    CCDS7243.1 (protein_coding)     A1CF    -
   chr10:g.52595929_52595930delinsAA/c.532_533delinsTT/p.P178L       inside_[cds_in_
↪exon_4]
   CSQN=Missense;codon_cDNA=532-533-534;source=CCDS
chr10:g.52595929_52595930delinsAA    CCDS7241.1 (protein_coding)     A1CF    -
   chr10:g.52595929_52595930delinsAA/c.508_509delinsTT/p.P170L       inside_[cds_in_
↪exon_4]
   CSQN=Missense;codon_cDNA=508-509-510;source=CCDS
chr10:g.52595929_52595930delinsAA    CCDS7242.1 (protein_coding)     A1CF    -
   chr10:g.52595929_52595930delinsAA/c.508_509delinsTT/p.P170L       inside_[cds_in_
↪exon_4]
   CSQN=Missense;codon_cDNA=508-509-510;source=CCDS
```

4.4 Promoter region

One can define the promoter boundary through the *–prombeg* and *–promend* option. Default promoter region is defined from 1000bp upstream of the transcription start site to the transcription start site. One could customize this setting to e.g., [-1000bp, 2000bp] by

```
$ transvar ganno -i 'chr19:g.41978629_41980350' --ensembl --prombeg 2000 --promend␣
↪1000 --refversion mm10
```

```
chr19:g.41978629_41980350    ENSMUST00000167927 (nonsense_mediated_decay)    MMS19   -
   chr19:g.41978629_41980350/c.115+649_115+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_237_bp(13.76%);aliases=ENSMUSP000001324
   83;source=Ensembl
chr19:g.41978629_41980350    ENSMUST00000171561 (protein_coding)     MMS19   -
   chr19:g.41978629_41980350/c.115+649_115+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_194_bp(11.27%);aliases=ENSMUSP000001309
   00;source=Ensembl
chr19:g.41978629_41980350    ENSMUST00000163398 (nonsense_mediated_decay)    MMS19   -
   chr19:g.41978629_41980350/c.115+649_115+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_234_bp(13.59%);aliases=ENSMUSP000001268
   64;source=Ensembl
chr19:g.41978629_41980350    ENSMUST00000164776 (nonsense_mediated_decay)    MMS19   -
   chr19:g.41978629_41980350/c.115+649_115+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_215_bp(12.49%);aliases=ENSMUSP000001294
   78;source=Ensembl
chr19:g.41978629_41980350    ENSMUST00000026168 (protein_coding)     MMS19   -
   chr19:g.41978629_41980350/c.115+649_115+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_219_bp(12.72%);aliases=ENSMUSP000000261
   68;source=Ensembl
chr19:g.41978629_41980350    ENSMUST00000171755 (retained_intron)    MMS19   -
   chr19:g.41978629_41980350/c.141+649_141+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_212_bp(12.31%);source=Ensembl
chr19:g.41978629_41980350    ENSMUST00000169775 (nonsense_mediated_decay)    MMS19   -
   chr19:g.41978629_41980350/c.115+649_115+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_214_bp(12.43%);aliases=ENSMUSP000001282
   34;source=Ensembl
chr19:g.41978629_41980350    ENSMUST00000168484 (nonsense_mediated_decay)    MMS19   -
   chr19:g.41978629_41980350/c.115+649_115+2370/.    inside_[intron_between_exon_1_
↪and_2]
   promoter_region_of_[MMS19]_overlaping_221_bp(12.83%);aliases=ENSMUSP000001268
   81;source=Ensembl
```

The last result shows that 12-13% of the target region is inside the promoter region. The overlap is as long as ~200 base pairs.

4.5 Splice sites

Consider a splice donor site chr7:5568790_5568791 (a donor site, intron side by definition, reverse strand, chr7:5568792- is the exon),

The 1st exonic nucleotide before donor splice site:

```
$ transvar ganno -i 'chr7:5568792C>G' --ccds
```

output a exonic variation and a missense variation

```
chr7:5568792C>G      CCDS5341.1 (protein_coding)      ACTB     -
   chr7:g.5568792C>G/c.363G>C/p.Q121H        inside_[cds_in_exon_2]
   CSQN=Missense;C2=NextToSpliceDonorOfExon2_At_chr7:5568791;codon_pos=5568792-5
   568793-5568794;ref_codon_seq=CAG;source=CCDS
```

The 1st nucleotide in the canonical donor splice site (intron side, this is commonly regarded as the splice site location):

```
$ transvar ganno -i 'chr7:5568791C>G' --ccds
```

output a splice variation

```
chr7:5568791C>G      CCDS5341.1 (protein_coding)      ACTB     -
   chr7:g.5568791C>G/c.363+1G>C/.    inside_[intron_between_exon_2_and_3]
   CSQN=SpliceDonorSNV;C2=SpliceDonorOfExon2_At_chr7:5568791;source=CCDS
```

The 2nd nucleotide in the canonical donor splice site (2nd on the intron side, still considered part of the splice site):

```
$ transvar ganno -i 'chr7:5568790A>G' --ccds
```

output a splice variation

```
chr7:5568790A>G      CCDS5341.1 (protein_coding)      ACTB     -
   chr7:g.5568790A>G/c.363+2T>C/.    inside_[intron_between_exon_2_and_3]
   CSQN=SpliceDonorSNV;C2=SpliceDonorOfExon2_At_chr7:5568791;source=CCDS
```

The 1st nucleotide downstream next to the canonical donor splice site (3rd nucleotide in the intron side, not part of the splice site):

```
$ transvar ganno -i 'chr7:5568789C>G' --ccds
```

output a pure intronic variation

```
chr7:5568789C>G      CCDS5341.1 (protein_coding)      ACTB     -
   chr7:g.5568789C>G/c.363+3G>C/.    inside_[intron_between_exon_2_and_3]
   CSQN=IntronicSNV;source=CCDS
```

4.6 UTR region

```
$ transvar ganno -i 'chr2:25564781G>T' --refseq
```

results in a UTR-containing CSQN field

```
chr2:25564781G>T     NM_022552.4 (protein_coding)     DNMT3A   -
   chr2:g.25564781G>T/c.1-27928C>A/. inside_[5-UTR;noncoding_exon_1]
   CSQN=5-UTRSNV;dbxref=GeneID:1788,HGNC:2978,HPRD:04141,MIM:602769;aliases=NP_0
   72046;source=RefSeq
chr2:25564781G>T     NM_175629.2 (protein_coding)     DNMT3A   -
   chr2:g.25564781G>T/c.1-27928C>A/. inside_[5-UTR;intron_between_exon_1_and_2]
   CSQN=IntronicSNV;dbxref=GeneID:1788,HGNC:2978,HPRD:04141,MIM:602769;aliases=N
   P_783328;source=RefSeq
chr2:25564781G>T     NM_175630.1 (protein_coding)     DNMT3A   -
   chr2:g.25564781G>T/c.1-27928C>A/. inside_[5-UTR;intron_between_exon_1_and_2]
   CSQN=IntronicSNV;dbxref=GeneID:1788,HGNC:2978,HPRD:04141,MIM:602769;aliases=N
   P_783329;source=RefSeq
```

4.7 Non-coding RNA

Given Ensembl, GENCODE or RefSeq database, one could annotate non-coding transcripts such as lncRNA. E.g.,

```
$ transvar ganno --gencode -i 'chr1:g.3985200_3985300' --refversion mm10
```

results in

```
chr1:g.3985200_3985300       ENSMUST00000194643.1 (lincRNA)  GM37381 -
   chr1:g.3985200_3985300/c.121_221/.        inside_[noncoding_exon_2]
   source=GENCODE
chr1:g.3985200_3985300       ENSMUST00000192427.1 (lincRNA)  GM37381 -
   chr1:g.3985200_3985300/c.685_785/.        inside_[noncoding_exon_1]
   source=GENCODE
```

or

```
$ transvar ganno --refseq -i 'chr14:g.20568338_20569581' --refversion mm10
```

results in

```
chr14:g.20568338_20569581    NR_033571.1 (lncRNA)    1810062O18RIK   +
   chr14:g.20568338_20569581/c.260-1532_260-289/.    inside_[intron_between_exon_4_
↪and_5]
   dbxref=GeneID:75602,MGI:MGI:1922852;source=RefSeq
chr14:g.20568338_20569581    NM_030180.2 (protein_coding)    USP54   -
   chr14:g.20568338_20569581/c.2188+667_2188+1910/.  inside_[intron_between_exon_15_
↪and_16]
   dbxref=GeneID:78787,MGI:MGI:1926037;aliases=NP_084456;source=RefSeq
chr14:g.20568338_20569581    XM_006519703.3 (protein_coding) USP54   -
   chr14:g.20568338_20569581/c.2359+667_2359+1910/.  inside_[intron_between_exon_16_
↪and_17]
   dbxref=GeneID:78787,MGI:MGI:1926037;aliases=XP_006519766;source=RefSeq
chr14:g.20568338_20569581    XM_011245226.2 (protein_coding) USP54   -
   chr14:g.20568338_20569581/c.1972+667_1972+1910/.  inside_[intron_between_exon_13_
↪and_14]
   dbxref=GeneID:78787,MGI:MGI:1926037;aliases=XP_011243528;source=RefSeq
chr14:g.20568338_20569581    XM_011245225.2 (protein_coding) USP54   -
   chr14:g.20568338_20569581/c.2359+667_2359+1910/.  inside_[intron_between_exon_16_
↪and_17]
   dbxref=GeneID:78787,MGI:MGI:1926037;aliases=XP_011243527;source=RefSeq
chr14:g.20568338_20569581    XM_006519705.3 (protein_coding) USP54   -
   chr14:g.20568338_20569581/c.2188+667_2188+1910/.  inside_[intron_between_exon_15_
↪and_16]
   dbxref=GeneID:78787,MGI:MGI:1926037;aliases=XP_006519768;source=RefSeq
chr14:g.20568338_20569581    XM_011245227.2 (protein_coding) USP54   -
   chr14:g.20568338_20569581/c.2359+667_2359+1910/.  inside_[intron_between_exon_16_
↪and_17]
   dbxref=GeneID:78787,MGI:MGI:1926037;aliases=XP_011243529;source=RefSeq
chr14:g.20568338_20569581    XM_017316224.1 (protein_coding) USP54   -
   chr14:g.20568338_20569581/c.2359+667_2359+1910/.  inside_[intron_between_exon_16_
↪and_17]
   dbxref=GeneID:78787,MGI:MGI:1926037;aliases=XP_017171713;source=RefSeq
```

or using Ensembl

```
$ transvar ganno --ensembl -i 'chr1:g.29560_29570'
```

results in

```
chr1:g.29560_29570   ENST00000488147 (unprocessed_pseudogene)        WASH7P  -
   chr1:g.29560_29570/c.1_11/.       inside_[noncoding_exon_1]
   promoter_region_of_[WASH7P]_overlaping_1_bp(9.09%);source=Ensembl
chr1:g.29560_29570   ENST00000538476 (unprocessed_pseudogene)        WASH7P  -
   chr1:g.29560_29570/c.237_247/.    inside_[noncoding_exon_1]
   source=Ensembl
chr1:g.29560_29570   ENST00000473358 (lincRNA)       MIR1302-10      +
   chr1:g.29560_29570/c.7_17/.       inside_[noncoding_exon_1]
   source=Ensembl
```

4.8 Coding Start and Stop

The following illustrates deletion of a coding start.

```
$ transvar ganno -i "chr7:g.5569279_5569288del" --ccds
```

results in

```
chr7:g.5569279_5569288del    CCDS5341.1 (protein_coding)     ACTB    -
   chr7:g.5569279_5569288delCATCATCCAT/c.3_12delGGATGATGAT/. inside_[cds_in_exon_1]
   CSQN=CdsStartDeletion;left_align_gDNA=g.5569277_5569286delATCATCATCC;unaligne
   d_gDNA=g.5569279_5569288delCATCATCCAT;left_align_cDNA=c.1_10delATGGATGATG;una
   lign_cDNA=c.1_10delATGGATGATG;cds_start_at_chr7:5569288_lost;source=CCDS
```

Deletion of a coding stop

```
$ transvar ganno -i "chr7:g.5567379_5567380del" --ccds
```

results in

Coding start loss due to SNP

```
$ transvar ganno -i "chr7:g.5568911T>A" --refseq
```

results in

```
chr7:g.5568911T>A    NM_001101.3 (protein_coding)    ACTB    -
   chr7:g.5568911T>A/c.244A>T/p.M82L inside_[cds_in_exon_3]
   CSQN=Missense;codon_pos=5568909-5568910-5568911;ref_codon_seq=ATG;dbxref=Gene
   ID:60,HGNC:132,HPRD:00032,MIM:102630;aliases=NP_001092;source=RefSeq
chr7:g.5568911T>A    XM_005249818.1 (protein_coding) ACTB    -
   chr7:g.5568911T>A/c.244A>T/p.M82L inside_[cds_in_exon_3]
   CSQN=Missense;codon_pos=5568909-5568910-5568911;ref_codon_seq=ATG;dbxref=Gene
   ID:60,HGNC:132,HPRD:00032,MIM:102630;aliases=XP_005249875;source=RefSeq
chr7:g.5568911T>A    XM_005249819.1 (protein_coding) ACTB    -
   chr7:g.5568911T>A/c.1A>T/.        inside_[cds_in_exon_2]
   CSQN=CdsStartSNV;C2=cds_start_at_chr7:5568911;dbxref=GeneID:60,HGNC:132,HPRD:
   00032,MIM:102630;aliases=XP_005249876;source=RefSeq
chr7:g.5568911T>A    XM_005249820.1 (protein_coding) ACTB    -
   chr7:g.5568911T>A/c.1-564A>T/.    inside_[5-UTR;noncoding_exon_3]
   CSQN=5-UTRSNV;dbxref=GeneID:60,HGNC:132,HPRD:00032,MIM:102630;aliases=XP_0052
   49877;source=RefSeq
```

Coding stop loss due to SNP

```
$ transvar ganno -i "chr7:g.5567379C>A" --refseq
```

results in

```
chr7:g.5567379C>A    NM_001101.3 (protein_coding)    ACTB    -
   chr7:g.5567379C>A/c.1128G>T/.      inside_[cds_in_exon_6]
   CSQN=CdsStopSNV;C2=cds_end_at_chr7:5567379;dbxref=GeneID:60,HGNC:132,HPRD:000
   32,MIM:102630;aliases=NP_001092;source=RefSeq
chr7:g.5567379C>A    XM_005249818.1 (protein_coding) ACTB    -
   chr7:g.5567379C>A/c.1128G>T/.      inside_[cds_in_exon_6]
   CSQN=CdsStopSNV;C2=cds_end_at_chr7:5567379;dbxref=GeneID:60,HGNC:132,HPRD:000
   32,MIM:102630;aliases=XP_005249875;source=RefSeq
chr7:g.5567379C>A    XM_005249819.1 (protein_coding) ACTB    -
   chr7:g.5567379C>A/c.885G>T/.       inside_[cds_in_exon_5]
   CSQN=CdsStopSNV;C2=cds_end_at_chr7:5567379;dbxref=GeneID:60,HGNC:132,HPRD:000
   32,MIM:102630;aliases=XP_005249876;source=RefSeq
chr7:g.5567379C>A    XM_005249820.1 (protein_coding) ACTB    -
   chr7:g.5567379C>A/c.762G>T/.       inside_[cds_in_exon_7]
   CSQN=CdsStopSNV;C2=cds_end_at_chr7:5567379;dbxref=GeneID:60,HGNC:132,HPRD:000
   32,MIM:102630;aliases=XP_005249877;source=RefSeq
```

4.9 Batch processing

To Illustrate batch processing with the following small batch input

```
$ cat data/small_batch_input
```

```
chr3 178936091       G       A       CCDS43171
chr9 135782704       C       G       CCDS6956
```

```
$ transvar ganno -l data/small_batch_input -g 1 -n 2 -r 3 -a 4 -t 5 --ccds
```

```
chr3|178936091|G|A|CCDS43171 CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_9]
   CSQN=Missense;dbsnp=rs104886003(chr3:178936091G>A);codon_pos=178936091-178936
   092-178936093;ref_codon_seq=GAG;source=CCDS
chr9|135782704|C|G|CCDS6956  CCDS6956.1 (protein_coding)     TSC1    -
   chr9:g.135782704C>G/c.1317G>C/p.L439L     inside_[cds_in_exon_11]
   CSQN=Synonymous;dbsnp=rs770692313(chr9:135782704C>G);codon_pos=135782704-1357
   82705-135782706;ref_codon_seq=CTG;source=CCDS
```

One can also make a HGVS-like input and call

```
$ cat data/small_batch_hgvs
```

```
CCDS43171    chr3:g.178936091G>A
CCDS6956     chr9:g.135782704C>G
```

```
$ transvar ganno -l data/small_batch_hgvs -m 2 -t 1 --ccds
```

```
CCDS43171|chr3:g.178936091G>A        CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_9]
```

(continues on next page)

(continued from previous page)

```
   CSQN=Missense;dbsnp=rs104886003(chr3:178936091G>A);codon_pos=178936091-178936
   092-178936093;ref_codon_seq=GAG;source=CCDS
CCDS6956|chr9:g.135782704C>G CCDS6956.1 (protein_coding)     TSC1     -
   chr9:g.135782704C>G/c.1317G>C/p.L439L     inside_[cds_in_exon_11]
   CSQN=Synonymous;dbsnp=rs770692313(chr9:135782704C>G);codon_pos=135782704-1357
   82705-135782706;ref_codon_seq=CTG;source=CCDS
```

The first column for transcript ID restriction is optional.

CHAPTER 5

Protein level annotation

Protein level inputs are handled by the *panno* subcommand.

5.1 Protein sites

To use uniprot id as protein name, one must first download the uniprot id map by

```
transvar config --download_idmap
```

Then one could use protein id instead of gene name by applying the *–idmap uniprot* option to TransVar. For example,

```
$ transvar panno --ccds -i 'Q5VUM1:47' --idmap uniprot
```

```
Q5VUM1:47    CCDS4972.1 (protein_coding)     C6ORF57 +
   chr6:g.71289191_71289193/c.139_141/p.47S  inside_[cds_in_exon_2]
   protein_sequence=S;cDNA_sequence=TCC;gDNA_sequence=TCC;source=CCDS
```

TransVar use a keyword extension *ref* in *Q5VUM1:p.47refS* to differentiate from the synonymous mutation *Q5VUM1:p.47S*. The former notation specifies that the reference protein sequence is *S* while the later specifies the target protein sequence is *S*.

5.2 Protein motif

For example, one can find the genomic location of a DRY motif in protein P28222 by issuing the following command,

```
$ transvar panno -i 'P28222:p.146_148refDRY' --ccds --idmap uniprot
```

```
P28222:p.146_148refDRY       CCDS4986.1 (protein_coding)     HTR1B   -
   chr6:g.78172677_78172685/c.436_444/p.D146_Y148    inside_[cds_in_exon_1]
```

(continues on next page)

(continued from previous page)

```
   protein_sequence=DRY;cDNA_sequence=GACCGCTAC;gDNA_sequence=GTAGCGGTC;source=C
   CDS
```

One can also use wildcard *x* (lowercase) in the motif.

```
$ transvar panno -i 'HTR1B:p.365_369refNPxxY' --ccds --seqmax 30
```

```
HTR1B:p.365_369refNPxxY      CCDS4986.1 (protein_coding)     HTR1B   -
   chr6:g.78172014_78172028/c.1093_1107/p.N365_Y369  inside_[cds_in_exon_1]
   protein_sequence=NPIIY;cDNA_sequence=AACCCCATAATCTAT;gDNA_sequence=ATAGATTATG
   GGGTT;source=CCDS
```

5.3 Protein region

```
$ transvar panno --ccds -i 'ABCB11:p.200_400'
```

outputs

```
ABCB11:p.200_400     CCDS46444.1 (protein_coding)    ABCB11  -
   chr2:g.169833195_169851872/c.598_1200/p.T200_K400 inside_[cds_in_exons_[6,7,8,9,10,
↪11]]
   protein_sequence=TRF..DRK;cDNA_sequence=ACA..AAA;gDNA_sequence=TTT..TGT;sourc
   e=CCDS
```

5.4 Protein variants

5.4.1 Single amino acid substitution

Mutation formats acceptable in TransVar are `PIK3CA:p.E545K` or without reference or alternative amino acid identity, e.g., `PIK3CA:p.545K` or `PIK3CA:p.E545`. TransVar takes native HGVS format inputs and outputs. The reference amino acid is used to narrow the search scope of candidate transcripts. The alternative amino acid is used to infer nucleotide change which results in the amino acid.

```
$ transvar panno -i PIK3CA:p.E545K --ensembl
```

outputs

```
PIK3CA:p.E545K       ENST00000263967 (protein_coding)        PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_10]
   CSQN=Missense;reference_codon=GAG;candidate_codons=AAG,AAA;candidate_mnv_vari
   ants=chr3:g.178936091_178936093delGAGinsAAA;dbsnp=rs104886003(chr3:178936091G
   >A);aliases=ENSP00000263967;source=Ensembl
```

One may encounter **ambiguous cases** where the multiple substitutions exist in explaining the amino acid change. For example,

```
$ transvar panno -i ACSL4:p.R133R --ccds
```

outputs

```
ACSL4:p.R133R        CCDS14548.1 (protein_coding)    ACSL4    -
   chrX:g.108926078G>T/c.399C>A/p.R133R      inside_[cds_in_exon_2]
   CSQN=Synonymous;reference_codon=CGC;candidate_codons=AGG,AGA,CGA,CGG,CGT;cand
   idate_snv_variants=chrX:g.108926078G>C,chrX:g.108926078G>A;candidate_mnv_vari
   ants=chrX:g.108926078_108926080delGCGinsCCT,chrX:g.108926078_108926080delGCGi
   nsTCT;source=CCDS
```

In those cases, TransVar prioritizes all the candidate base changes by minimizing the edit distance between the reference codon sequence and the target codon sequence. One of the optimal base changes is arbitrarily chosen as the default and all the candidates are included in the appended *CddMuts* entry.

Ambiguous amino acid code

TransVar instantiates input of ambiguous amino acid code such as ('B', for "Asx", which stands for "Asp" or "Asn") to more specific amino acid. Even if the reference amino acid is a subset of the ambiguous alternative amino acid, TransVar assume a mutation on the nucleotide level (can still deduce synonymous mutations):

```
$ transvar panno -i 'APC:p.D326B' --ccds
```

```
APC:p.D326B  CCDS4107.1 (protein_coding)     APC     +
   chr5:g.112154705G>A/c.976G>A/p.D326N      inside_[cds_in_exon_9]
   CSQN=Missense;reference_codon=GAT;candidate_codons=AAC,AAT,GAC;candidate_snv_
   variants=chr5:g.112154707T>C;candidate_mnv_variants=chr5:g.112154705_11215470
   7delGATinsAAC;source=CCDS
```

Here input alternative amino acids is B (D or N). After TransVar processing, a 'N' is derived (though a D is equally likely, as shown in the candidates).

5.4.2 Insertion

```
$ transvar panno --ccds -i 'AATK:p.P1331_A1332insTP'
```

```
AATK:p.P1331_A1332insTP      CCDS45807.1 (protein_coding)    AATK    -
   chr17:g.79093270_79093271insAGGTGT/c.3993_3994insACACCT/p.T1330_P1331dupTP
↪inside_[cds_in_exon_13]
   CSQN=InFrameInsertion;left_align_protein=p.A1326_P1327insPT;unalign_protein=p
   .T1330_P1331dupTP;left_align_gDNA=g.79093270_79093271insAGGTGT;unalign_gDNA=g
   .79093270_79093271insAGGTGT;left_align_cDNA=c.3993_3994insACACCT;unalign_cDNA
   =c.3993_3994insACACCT;16_CandidatesOmitted;source=CCDS
```

5.4.3 Deletion

```
$ transvar panno --ccds -i 'AADACL4:p.W263_I267delWRDAI'
```

```
AADACL4:p.W263_I267delWRDAI  CCDS30590.1 (protein_coding)    AADACL4 +
   chr1:g.12726310_12726324del15/c.788_802del15/p.W263_I267delWRDAI  inside_[cds_in_
↪exon_4]
   CSQN=InFrameDeletion;left_align_gDNA=g.12726308_12726322del15;unaligned_gDNA=
   g.12726309_12726323del15;left_align_cDNA=c.786_800del15;unalign_cDNA=c.787_80
   1del15;left_align_protein=p.W263_I267delWRDAI;unalign_protein=p.W263_I267delW
   RDAI;imprecise;source=CCDS
```

5.4.4 Block substitution

```
$ transvar panno --ccds -i 'ABCC3:p.Y556_V557delinsRRR'
```

```
ABCC3:p.Y556_V557delinsRRR   CCDS32681.1 (protein_coding)    ABCC3   +
   chr17:g.48745254_48745259delinsAGGAGGAGG/c.1666_1671delinsAGGAGGAGG/p.Y556_
↪V557delinsRRR  inside_[cds_in_exon_13]
   CSQN=MultiAAMissense;216_CandidatesOmitted;source=CCDS
```

Sometimes block substitution comes from in-frame deletion on the nucleotide level.

```
$ transvar panno -i 'MAP2K1:p.F53_Q58delinsL' --ensembl
```

```
MAP2K1:p.F53_Q58delinsL      ENST00000307102 (protein_coding)        MAP2K1  +
   chr15:g.66727443_66727457del15/c.159_173del15/p.F53_Q58delinsL    inside_[cds_in_
↪exon_2]
   CSQN=MultiAAMissense;left_align_gDNA=g.66727443_66727457del15;unaligned_gDNA=
   g.66727443_66727457del15;left_align_cDNA=c.159_173del15;unalign_cDNA=c.159_17
   3del15;candidate_alternative_sequence=CTT/CTG/CTA/CTC/TTA/TTG;aliases=ENSP000
   00302486;source=Ensembl
```

5.4.5 Frame-shift variants

Frame-shift variants can be results of either insertion or deletion. In the cases where both are plausible the variants are prioritized by the length of the insertion/deletion. Mutations of smallest variants are given as the most likely inference. Other candidates are in given in the *candidates* field.

```
$ transvar panno --refseq -i 'PTEN:p.T319fs*1' --max-candidates 2
```

```
PTEN:p.T319fs*1      NM_000314.4 (protein_coding)    PTEN    +
   chr10:g.89720803_89720804insTA/c.954_955insTA/p.T319fs*1  inside_[cds_in_exon_8]
   CSQN=Frameshift;left_align_cDNA=c.954_955insTA;left_align_gDNA=g.89720803_897
   20804insTA;candidates=g.89720803_89720804insTG/c.954_955insTG/g.89720803_8972
   0804insTG/c.954_955insTG,g.89720804_89720807delACTT/c.955_958delACTT/g.897207
   99_89720802delTACT/c.950_953delTACT;1_CandidatesOmitted;dbxref=GeneID:5728,HG
   NC:9588,MIM:601728;aliases=NP_000305;source=RefSeq
```

In this example, both deletion *c.950_953delTACT* and insertion *c.954_955insTA* are possible. Both insertion involves fewer nucleotides and is chosen as the most likely inference. Deletion is given in the *candidates* tag.

The *candidates* field shows the right-aligned genomic, right-aligned cDNA, left-aligned genomic and left-aligned cDNA identifiers separated by */*.

```
$ transvar panno --ccds -i 'A1BG:p.G132fs*2' --max-candidates 1
```

```
A1BG:p.G132fs*2      CCDS12976.1 (protein_coding)    A1BG    -
   chr19:g.58863868delC/c.395delG/p.G132fs*2 inside_[cds_in_exon_4]
   CSQN=Frameshift;left_align_cDNA=c.394delG;left_align_gDNA=g.58863867delC;cand
   idates=g.58863873delG/c.393delC/g.58863869delG/c.389delC;13_CandidatesOmitted
   ;source=CCDS
```

Frameshift variants can be difficult since there might be too many valid underlying nucleotide variants. Suppose we have a relatively long insertion,

```
$ transvar ganno -i 'chr11:g.32417908_32417909insACCGTACA' --ccds
```

```
chr11:g.32417908_32417909insACCGTACA CCDS55750.1 (protein_coding)    WT1     -
   chr11:g.32417908_32417909insACCGTACA/c.456_457insTGTACGGT/p.A153Cfs*70    inside_
↪[cds_in_exon_6]
   CSQN=Frameshift;left_align_gDNA=g.32417908_32417909insACCGTACA;unalign_gDNA=g
   .32417908_32417909insACCGTACA;left_align_cDNA=c.456_457insTGTACGGT;unalign_cD
   NA=c.456_457insTGTACGGT;source=CCDS
chr11:g.32417908_32417909insACCGTACA CCDS55751.1 (protein_coding)    WT1     -
   chr11:g.32417908_32417909insACCGTACA/c.507_508insTGTACGGT/p.A170Cfs*70    inside_
↪[cds_in_exon_7]
   CSQN=Frameshift;left_align_gDNA=g.32417908_32417909insACCGTACA;unalign_gDNA=g
   .32417908_32417909insACCGTACA;left_align_cDNA=c.507_508insTGTACGGT;unalign_cD
   NA=c.507_508insTGTACGGT;source=CCDS
chr11:g.32417908_32417909insACCGTACA CCDS44561.1 (protein_coding)    WT1     -
   chr11:g.32417908_32417909insACCGTACA/c.1092_1093insTGTACGGT/p.A365Cfs*70  inside_
↪[cds_in_exon_6]
   CSQN=Frameshift;left_align_gDNA=g.32417908_32417909insACCGTACA;unalign_gDNA=g
   .32417908_32417909insACCGTACA;left_align_cDNA=c.1092_1093insTGTACGGT;unalign_
   cDNA=c.1092_1093insTGTACGGT;source=CCDS
chr11:g.32417908_32417909insACCGTACA CCDS44562.1 (protein_coding)    WT1     -
   chr11:g.32417908_32417909insACCGTACA/c.1143_1144insTGTACGGT/p.A382Cfs*70  inside_
↪[cds_in_exon_7]
   CSQN=Frameshift;left_align_gDNA=g.32417908_32417909insACCGTACA;unalign_gDNA=g
   .32417908_32417909insACCGTACA;left_align_cDNA=c.1143_1144insTGTACGGT;unalign_
   cDNA=c.1143_1144insTGTACGGT;source=CCDS
chr11:g.32417908_32417909insACCGTACA CCDS7878.2 (protein_coding)     WT1     -
   chr11:g.32417908_32417909insACCGTACA/c.1143_1144insTGTACGGT/p.A382Cfs*70  inside_
↪[cds_in_exon_7]
   CSQN=Frameshift;left_align_gDNA=g.32417908_32417909insACCGTACA;unalign_gDNA=g
   .32417908_32417909insACCGTACA;left_align_cDNA=c.1143_1144insTGTACGGT;unalign_
   cDNA=c.1143_1144insTGTACGGT;source=CCDS
```

But now suppose we only know its protein identifier and forget about the original identifier. Using *panno*, we can get roughly how the original identifier look like:

```
$ transvar panno -i 'WT1:p.A170Cfs*70' --ccds --max-candidates 2
```

would return more than 80 underlying variants. In this case the argument *–max-candidates* (default to 10) controls the maximum number of candidates output.

```
WT1:p.A170Cfs*70     CCDS55751.1 (protein_coding)    WT1     -
   chr11:g.32417908_32417909insTTGGGGCA/c.507_508insTGCCCCAA/p.A170Cfs*70    inside_
↪[cds_in_exons_[7,8,9]]
   CSQN=Frameshift;left_align_cDNA=c.507_508insTGCCCCAA;left_align_gDNA=g.324179
   08_32417909insTTGGGGCA;candidates=g.32417908_32417909insTTGNNNCA/c.507_508ins
   TGNNNCAA/g.32417908_32417909insTTGNNNCA/c.507_508insTGNNNCAA,g.32417908_32417
   909insGTGNNNCA/c.507_508insTGNNNCAC/g.32417908_32417909insGTGNNNCA/c.507_508i
   nsTGNNNCAC;80_CandidatesOmitted;source=CCDS
```

Sometimes the alternative amino acid can be missing

```
$ transvar panno -i ADAMTSL1:p.I396fs*30 --ccds --max-candidates 2
```

```
ADAMTSL1:p.I396fs*30 CCDS6485.1 (protein_coding)     ADAMTSL1        +
   chr9:g.18680360_18680361insG/c.1187_1188insG/p.I396fs*30  inside_[cds_in_exon_11]
```

(continues on next page)

(continued from previous page)

```
   CSQN=Frameshift;left_align_cDNA=c.1187_1188insG;left_align_gDNA=g.18680360_18
   680361insG;candidates=g.18680359dupA/c.1186dupA/g.18680358_18680359insA/c.118
   5_1186insA,g.18680359_18680360insC/c.1186_1187insC/g.18680359_18680360insC/c.
   1186_1187insC;11_CandidatesOmitted;source=CCDS
ADAMTSL1:p.I396fs*30 CCDS47954.1 (protein_coding)    ADAMTSL1        +
   chr9:g.18680360_18680361insG/c.1187_1188insG/p.I396fs*30  inside_[cds_in_exon_11]
   CSQN=Frameshift;left_align_cDNA=c.1187_1188insG;left_align_gDNA=g.18680360_18
   680361insG;candidates=g.18680359dupA/c.1186dupA/g.18680358_18680359insA/c.118
   5_1186insA,g.18680359_18680360insC/c.1186_1187insC/g.18680359_18680360insC/c.
   1186_1187insC;11_CandidatesOmitted;source=CCDS
```

TransVar can also take protein identifiers such as as input. For example,

```
$ transvar panno --refseq -i 'NP_006266:p.G240Afs*50' --idmap protein_id
```

```
NP_006266:p.G240Afs*50       NM_006275.5 (protein_coding)    SRSF6   +
   chr20:g.42089387delG/c.719delG/p.G240Afs*50       inside_[cds_in_exon_6]
   CSQN=Frameshift;left_align_cDNA=c.718delG;left_align_gDNA=g.42089386delG;cand
   idates=g.42089385delA/c.717delA/g.42089382delA/c.714delA;dbxref=GeneID:6431,H
   GNC:10788,HPRD:09054,MIM:601944;aliases=NP_006266;source=RefSeq
```

The output gives the exact details of the mutation on the DNA levels, properly right-aligned. The *candidates* fields also include other equally-likely mutation identifiers. *candidates* have the format *[right-align-gDNA]/[right-align-cDNA]/[left-align-gDNA]/[left-align-cDNA]* for each hit and , separation between hits.

Similar applies when the underlying mutation is an insertion. TransVar can infer insertion sequence of under 3 base pairs long. For example,

```
$ transvar panno -i 'AASS:p.I355Mfs*10' --ccds --max-candidates 1
```

```
AASS:p.I355Mfs*10    CCDS5783.1 (protein_coding)     AASS    -
   chr7:g.121753753_121753754insTC/c.1064_1065insGA/p.I355Mfs*10     inside_[cds_in_
↪exon_9]
   CSQN=Frameshift;left_align_cDNA=c.1064_1065insGA;left_align_gDNA=g.121753753_
   121753754insTC;candidates=g.121753753_121753754insGC/c.1064_1065insGC/g.12175
   3753_121753754insGC/c.1064_1065insGC;3_CandidatesOmitted;source=CCDS
```

When the alternative becomes a stop codon, frameshift mutation becomes a nonsense mutation:

```
$ transvar panno -i 'APC:p.I1557*fs*3' --ccds
```

returns a nonsense mutation

```
APC:p.I1557*fs*3     CCDS4107.1 (protein_coding)     APC     +
   chr5:g.112175960_112175962delATTinsTAA/c.4669_4671delATTinsTAA/p.I1557*   inside_
↪[cds_in_exon_15]
   CSQN=Nonsense;reference_codon=ATT;candidate_codons=TAA,TAG,TGA;candidate_mnv_
   variants=chr5:g.112175960_112175962delATTinsTAG,chr5:g.112175960_112175962del
   ATTinsTGA;source=CCDS
```

5.5 Whole transcript

TransVar provides an easy way to investigate a whole transcript by supplying the gene id.

```
$ transvar panno -i 'Dnmt3a' --refseq
```

outputs the basic information of transcripts of the protein, in an intuitive way,

```
Dnmt3a       XM_005264176.1 (protein_coding) DNMT3A  -
   chr2:g.25451421_25537541/c.1_2739/p.M1_*913       whole_transcript
   promoter=chr2:25537541_25538541;#exons=23;cds=chr2:25457148_25536853
Dnmt3a       XM_005264175.1 (protein_coding) DNMT3A  -
   chr2:g.25451421_25537354/c.1_2739/p.M1_*913       whole_transcript
   promoter=chr2:25537354_25538354;#exons=23;cds=chr2:25457148_25536853
Dnmt3a       XM_005264177.1 (protein_coding) DNMT3A  -
   chr2:g.25451421_25475145/c.1_2070/p.M1_*690       whole_transcript
   promoter=chr2:25475145_25476145;#exons=18;cds=chr2:25457148_25471091
Dnmt3a       NM_175629.2 (protein_coding)    DNMT3A  -
   chr2:g.25455830_25565459/c.1_2739/p.M1_*913       whole_transcript
   promoter=chr2:25565459_25566459;#exons=23;cds=chr2:25457148_25536853
Dnmt3a       NM_022552.4 (protein_coding)    DNMT3A  -
   chr2:g.25455830_25564784/c.1_2739/p.M1_*913       whole_transcript
   promoter=chr2:25564784_25565784;#exons=23;cds=chr2:25457148_25536853
Dnmt3a       NM_153759.3 (protein_coding)    DNMT3A  -
   chr2:g.25455830_25475184/c.1_2172/p.M1_*724       whole_transcript
   promoter=chr2:25475184_25476184;#exons=19;cds=chr2:25457148_25475066
Dnmt3a       NM_175630.1 (protein_coding)    DNMT3A  -
   chr2:g.25504321_25565459/c.1_501/p.M1_*167        whole_transcript
   promoter=chr2:25565459_25566459;#exons=4;cds=chr2:25505257_25536853
```

5.6 Search alternative codon identifiers

An identifier is regarded as an alternative if the underlying codon overlap with the one from the original identifier. Example: to search alternative identifiers of CDKN2A.p.58 (without knowing reference allele),

```
$ transvar codonsearch --ccds -i CDKN2A:p.58
```

```
origin_id    alt_id  chrm    codon1  codon2  transcripts_choice
CDKN2A:p.58  CDKN2A.p.73     chr9    21971184-21971185-21971186
   21971182-21971183-21971184        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
CDKN2A:p.58  CDKN2A.p.72     chr9    21971184-21971185-21971186
   21971185-21971186-21971187        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
```

The pair of transcript id listed corresponds to the transcripts based on which, the original and alternative identifiers are defined. Multiple pairs of transcript definitions are appended following a ,.

Example: to search alternative identifiers of DHODH:G152R (knowing reference allele *G*, alternative allele here will be ignored),

```
$ transvar codonsearch -i DHODH:G152R --refseq
```

outputs

```
origin_id    alt_id  chrm    codon1  codon2  transcripts_choice
DHODH:G152R  DHODH.p.G124    chr16   72050942-72050943-72050944
   72050942-72050943-72050944        NM_001361[RefSeq]/XM_005255827[RefSeq]
```

(continues on next page)

(continued from previous page)

```
DHODH:G152R  DHODH.p.G16     chr16   72050942-72050943-72050944
   72050942-72050943-72050944        NM_001361[RefSeq]/XM_005255828[RefSeq]
DHODH:G152R  DHODH.p.G9      chr16   72050942-72050943-72050944
   72050942-72050943-72050944        NM_001361[RefSeq]/XM_005255829[RefSeq]
```

TransVar outputs genomic positions of codons based on original transcript (4th column in the output) and alternative transcript (5th column in the output). The potential transcript usages are also appended.

Example: to run *transvar codonsearch* to **batch process** a list of mutation identifiers.

```
$ transvar codonsearch -l example/input_table2 --ccds -m 1 -o 1
```

Example input table

```
origin_id    alt_id  chrm    codon1  codon2  transcripts_choice
CDKN2A:p.61  CDKN2A.p.76     chr9    21971175-21971176-21971177
   21971173-21971174-21971175        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
CDKN2A:p.61  CDKN2A.p.75     chr9    21971175-21971176-21971177
   21971176-21971177-21971178        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
CDKN2A:p.69  CDKN2A.p.84     chr9    21971151-21971152-21971153
   21971149-21971150-21971151        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
CDKN2A:p.69  CDKN2A.p.83     chr9    21971151-21971152-21971153
   21971152-21971153-21971154        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
CDKN2A:p.69  CDKN2A.p.55     chr9    21971194-21971195-21971196
   21971193-21971194-21971195        CCDS6511[CCDS]/CCDS6510[CCDS],CCDS6511[CCDS]/
↪CCDS56565[CCDS]
CDKN2A:p.69  CDKN2A.p.54     chr9    21971194-21971195-21971196
   21971196-21971197-21971198        CCDS6511[CCDS]/CCDS6510[CCDS],CCDS6511[CCDS]/
↪CCDS56565[CCDS]
ERBB2:p.755  ERBB2.p.725     chr17   37880219-37880220-37880221
   37880219-37880220-37880221        CCDS32642[CCDS]/CCDS45667[CCDS]
ERBB2:p.755  ERBB2.p.785     chr17   37881024-37881025-37881026
   37881024-37881025-37881026        CCDS45667[CCDS]/CCDS32642[CCDS]
```

outputs

```
origin_id    alt_id  chrm    codon1
   codon2    transcripts_choice
CDKN2A:p.61  CDKN2A.p.76     chr9    21971175-21971176-21971177
   21971173-21971174-21971175        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
CDKN2A:p.61  CDKN2A.p.75     chr9    21971175-21971176-21971177
   21971176-21971177-21971178        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
CDKN2A:p.69  CDKN2A.p.54     chr9    21971194-21971195-21971196
   21971196-21971197-21971198        CCDS6511[CCDS]/CCDS6510[CCDS],CCDS6511[CCDS]/
↪CCDS56565[CCDS]
CDKN2A:p.69  CDKN2A.p.55     chr9    21971194-21971195-21971196
   21971193-21971194-21971195        CCDS6511[CCDS]/CCDS6510[CCDS],CCDS6511[CCDS]/
↪CCDS56565[CCDS]
CDKN2A:p.69  CDKN2A.p.83     chr9    21971151-21971152-21971153
   21971152-21971153-21971154        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
```

(continues on next page)

(continued from previous page)

```
CDKN2A:p.69  CDKN2A.p.84     chr9    21971151-21971152-21971153
   21971149-21971150-21971151        CCDS6510[CCDS]/CCDS6511[CCDS],CCDS56565[CCDS]/
↪CCDS6511[CCDS]
ERBB2:p.755  ERBB2.p.785     chr17   37881024-37881025-37881026
   37881024-37881025-37881026        CCDS45667[CCDS]/CCDS32642[CCDS]
ERBB2:p.755  ERBB2.p.725     chr17   37880219-37880220-37880221
   37880219-37880220-37880221        CCDS32642[CCDS]/CCDS45667[CCDS]
```

The third column indicates the potential transcript usage for the alternative identifier. Each transcript usage is denoted by <listing transcript>/<actual transcript>. Different potential choices are separated by ‘,’.

5.6.1 Infer potential codon identity

Example: to check if MET.p1010 and MET.p992 may be refering to one mutation due to different usage of transcripts,

```
$ transvar codonsearch --refseq -i MET:p.1010
```

gives

```
origin_id    alt_id  chrm    codon1  codon2  transcripts_choice
MET:p.1010   MET.p.973       chr7    116411932-116411933-116411934
   116411932-116411933-116411934     XM_005250353[RefSeq]/NM_000245[RefSeq]
MET:p.1010   MET.p.991       chr7    116411932-116411933-116411934
   116411932-116411933-116411934     XM_005250353[RefSeq]/NM_001127500[RefSeq]
MET:p.1010   MET.p.543       chr7    116411932-116411933-116411934
   116411932-116411933-116411934     XM_005250353[RefSeq]/XM_005250354[RefSeq]
MET:p.1010   MET.p.1029      chr7    116411989-116411990-116411991
   116411989-116411990-116411991     NM_001127500[RefSeq]/XM_005250353[RefSeq]
MET:p.1010   MET.p.992       chr7    116411989-116411990-116411991
   116411989-116411990-116411991     NM_001127500[RefSeq]/NM_000245[RefSeq]
MET:p.1010   MET.p.562       chr7    116411989-116411990-116411991
   116411989-116411990-116411991     NM_001127500[RefSeq]/XM_005250354[RefSeq]
MET:p.1010   MET.p.1047      chr7    116412043-116414935-116414936
   116412043-116414935-116414936     NM_000245[RefSeq]/XM_005250353[RefSeq]
MET:p.1010   MET.p.1028      chr7    116412043-116414935-116414936
   116412043-116414935-116414936     NM_000245[RefSeq]/NM_001127500[RefSeq]
MET:p.1010   MET.p.580       chr7    116412043-116414935-116414936
   116412043-116414935-116414936     NM_000245[RefSeq]/XM_005250354[RefSeq]
```

Since MET.p.992 is in the list, the two identifiers might be due to the same genomic mutation.

CHAPTER 6

cDNA level annotation

Annotation from cDNA level is handled by the *canno* subcommand.

6.1 cDNA region

```
$ transvar canno --ccds -i 'ABCB11:c.1198-8_1202'
```

outputs

```
ABCB11:c.1198-8_1202 CCDS46444.1 (protein_coding)    ABCB11  -
   chr2:g.169833193_169833205GGTTTCTGGAGTG/c.1198-8_1202CACTCCAGAAACC/p.400_401KP
↪from_[cds_in_exon_11]_to_[intron_between_exon_10_and_11]
   C2=acceptor_splice_site_on_exon_11_at_chr2:169833198_included;source=CCDS
```

6.2 cDNA variant

6.2.1 Single Nucleotide Variation (SNV)

TransVar infers nucleotide mutation through `PIK3CA:c.1633G>A`. Note that nucleotide identity follows the natural sequence, i.e., if transcript is interpreted on the reverse-complementary strand, the base at the site needs to be reverse-complemented too.

```
$ transvar canno --ccds -i 'PIK3CA:c.1633G>A'
```

outputs

```
PIK3CA:c.1633G>A     CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_9]
   CSQN=Missense;dbsnp=rs104886003(chr3:178936091G>A);reference_codon=GAG;altern
   ative_codon=AAG;source=CCDS
```

The SNV can be in the intronic region, e.g.,

```
$ transvar canno --ccds -i 'ABCB11:c.1198-8C>A'
```

outputs

```
ABCB11:c.1198-8C>A   CCDS46444.1 (protein_coding)    ABCB11   -
   chr2:g.169833205G>T/c.1198-8C>A/. inside_[intron_between_exon_10_and_11]
   CSQN=IntronicSNV;source=CCDS
```

Or in the 5'-UTR region, e.g.,

```
$ transvar canno -i 'KCNJ11:c.-134G>T' --ensembl
```

```
KCNJ11:c.-134G>T     ENST00000339994 (protein_coding)        KCNJ11   -
   chr11:g.17409772C>A/c.1-134G>T/.  inside_[5-UTR;noncoding_exon_1]
   CSQN=5-UTRSNV;dbsnp=rs387906398(chr11:17409772C>A);aliases=ENSP00000345708;so
   urce=Ensembl
```

Or in the 3'-UTR region, e.g.,

```
$ transvar canno -i 'MSH2:c.*95C>T' --refseq
```

```
MSH2:c.*95C>T        NM_000251.2 (protein_coding)    MSH2     +
   chr2:g.47710183C>T/c.*95C>T/.     inside_[3-UTR;noncoding_exon_16]
   CSQN=3-UTRSNV;dbsnp=rs587779062(chr2:47710183C>T);dbxref=GeneID:4436,HGNC:732
   5,HPRD:00389,MIM:609309;aliases=NP_000242;source=RefSeq
MSH2:c.*95C>T        NM_001258281.1 (protein_coding) MSH2     +
   chr2:g.47710183C>T/c.*95C>T/.     inside_[3-UTR;noncoding_exon_17]
   CSQN=3-UTRSNV;dbsnp=rs587779062(chr2:47710183C>T);dbxref=GeneID:4436,HGNC:732
   5,HPRD:00389,MIM:609309;aliases=NP_001245210;source=RefSeq
MSH2:c.*95C>T        XM_005264333.1 (protein_coding) MSH2     +
   chr2:g.47710183C>T/c.*95C>T/.     inside_[3-UTR;noncoding_exon_15]
   CSQN=3-UTRSNV;dbsnp=rs587779062(chr2:47710183C>T);dbxref=GeneID:4436,HGNC:732
   5,HPRD:00389,MIM:609309;aliases=XP_005264390;source=RefSeq
```

6.2.2 insertion

An insertion may result in: 1) a pure insertion of amino acids; 2) a block substitution of amino acids, when insertion occur after 1st or 2nd base in a codon; or 3) a frame-shift. Following HGVS nomenclature, TransVar labels the first different amino acid and the length of the peptide util stop codon, assuming no change in the splicing.

Example: to annotate an **in-frame, in-phase insertion**,

```
$ transvar canno --ccds -i 'ACIN1:c.1932_1933insATTCAC'
```

```
ACIN1:c.1932_1933insATTCAC   CCDS9587.1 (protein_coding)     ACIN1    -
   chr14:g.23548785_23548786insGTGAAT/c.1932_1933insATTCAC/p.R644_S645insIH  inside_
↪[cds_in_exon_6]
   CSQN=InFrameInsertion;left_align_gDNA=g.23548785_23548786insGTGAAT;unalign_gD
   NA=g.23548785_23548786insGTGAAT;left_align_cDNA=c.1932_1933insATTCAC;unalign_
   cDNA=c.1932_1933insATTCAC;left_align_protein=p.R644_S645insIH;unalign_protein
   =p.R644_S645insIH;phase=0;source=CCDS
ACIN1:c.1932_1933insATTCAC   CCDS53889.1 (protein_coding)    ACIN1    -
   chr14:g.23548157_23548158insGTGAAT/c.1932_1933insATTCAC/p.P644_V645insIH  inside_
↪[cds_in_exon_6]
```

(continues on next page)

(continued from previous page)

```
   CSQN=InFrameInsertion;left_align_gDNA=g.23548157_23548158insGTGAAT;unalign_gD
   NA=g.23548157_23548158insGTGAAT;left_align_cDNA=c.1932_1933insATTCAC;unalign_
   cDNA=c.1932_1933insATTCAC;left_align_protein=p.P644_V645insIH;unalign_protein
   =p.P644_V645insIH;phase=0;source=CCDS
ACIN1:c.1932_1933insATTCAC   CCDS55905.1 (protein_coding)    ACIN1   -
   chr14:g.23548785_23548786insGTGAAT/c.1932_1933insATTCAC/p.R644_S645insIH  inside_
↪[cds_in_exon_6]
   CSQN=InFrameInsertion;left_align_gDNA=g.23548785_23548786insGTGAAT;unalign_gD
   NA=g.23548785_23548786insGTGAAT;left_align_cDNA=c.1932_1933insATTCAC;unalign_
   cDNA=c.1932_1933insATTCAC;left_align_protein=p.R644_S645insIH;unalign_protein
   =p.R644_S645insIH;phase=0;source=CCDS
```

`Phase = 0,1,2` indicates whether the insertion happen after the 3rd, 1st or 2nd base of a codon, respectively. An insertion *in phase* refers to one with `Phase=0`.

Example: to annotate an **out-of-phase, in-frame insertion**,

```
$ transvar canno --ccds -i 'ACIN1:c.1930_1931insATTCAC'
```

```
ACIN1:c.1930_1931insATTCAC   CCDS9587.1 (protein_coding)    ACIN1   -
   chr14:g.23548792_23548793insTGTGAA/c.1930_1931insATTCAC/p.S643_R644insHS  inside_
↪[cds_in_exon_6]
   CSQN=InFrameInsertion;left_align_gDNA=g.23548787_23548788insGTGAAT;unalign_gD
   NA=g.23548787_23548788insGTGAAT;left_align_cDNA=c.1925_1926insTTCACA;unalign_
   cDNA=c.1930_1931insATTCAC;left_align_protein=p.R642_S643insSH;unalign_protein
   =p.S643_R644insHS;phase=1;source=CCDS
ACIN1:c.1930_1931insATTCAC   CCDS53889.1 (protein_coding)    ACIN1   -
   chr14:g.23548162_23548163insAATGTG/c.1930_1931insATTCAC/p.P643_P644insHS  inside_
↪[cds_in_exon_6]
   CSQN=InFrameInsertion;left_align_gDNA=g.23548159_23548160insGTGAAT;unalign_gD
   NA=g.23548159_23548160insGTGAAT;left_align_cDNA=c.1927_1928insCACATT;unalign_
   cDNA=c.1930_1931insATTCAC;left_align_protein=p.P643_P644insHS;unalign_protein
   =p.P643_P644insHS;phase=1;source=CCDS
ACIN1:c.1930_1931insATTCAC   CCDS55905.1 (protein_coding)    ACIN1   -
   chr14:g.23548792_23548793insTGTGAA/c.1930_1931insATTCAC/p.S643_R644insHS  inside_
↪[cds_in_exon_6]
   CSQN=InFrameInsertion;left_align_gDNA=g.23548787_23548788insGTGAAT;unalign_gD
   NA=g.23548787_23548788insGTGAAT;left_align_cDNA=c.1925_1926insTTCACA;unalign_
   cDNA=c.1930_1931insATTCAC;left_align_protein=p.R642_S643insSH;unalign_protein
   =p.S643_R644insHS;phase=1;source=CCDS
```

Reverse annotation can result in different identifiers after left/right alignments, e.g.,

```
$ transvar canno --ccds -i 'AATK:c.3976_3977insCGCCCA'
```

results in

```
AATK:c.3976_3977insCGCCCA    CCDS45807.1 (protein_coding)    AATK    -
   chr17:g.79093282_79093287dupTGGGCG/c.3988_3993dupACGCCC/p.T1330_P1331dupTP
↪inside_[cds_in_exon_13]
   CSQN=InFrameInsertion;left_align_gDNA=g.79093270_79093271insGGGCGT;unalign_gD
   NA=g.79093282_79093287dupTGGGCG;left_align_cDNA=c.3976_3977insCGCCCA;unalign_
   cDNA=c.3976_3977insCGCCCA;left_align_protein=p.A1326_P1327insPT;unalign_prote
   in=p.A1326_P1327insPT;phase=1;source=CCDS
```

Note how insertion switch to duplication when 5'flanking is identical. This conforms to HGVS recommendation to replace insertion notation with duplication when possible.

Example: to annotate a **frame-shift insertion**, frameshift mutations have not alternative alignments. Hence only cDNA and gDNA have left alignment and unalignment reports.

```
$ transvar canno --ccds -i 'AAAS:c.1225_1226insG'
```

results in

```
AAAS:c.1225_1226insG CCDS8856.1 (protein_coding)     AAAS    -
   chr12:g.53702093dupC/c.1225dupG/p.E409Gfs*17      inside_[cds_in_exon_13]
   CSQN=Frameshift;left_align_gDNA=g.53702089_53702090insC;unalign_gDNA=g.537020
   89_53702090insC;left_align_cDNA=c.1221_1222insG;unalign_cDNA=c.1225dupG;sourc
   e=CCDS
AAAS:c.1225_1226insG CCDS53797.1 (protein_coding)    AAAS    -
   chr12:g.53701842_53701843insC/c.1225_1226insG/p.L409Rfs*54        inside_[cds_in_
↪exon_13]
   CSQN=Frameshift;left_align_gDNA=g.53701842_53701843insC;unalign_gDNA=g.537018
   42_53701843insC;left_align_cDNA=c.1225_1226insG;unalign_cDNA=c.1225_1226insG;
   source=CCDS
```

Example: to annotate an **intronic insertion**,

```
$ transvar canno --ccds -i 'ADAM33:c.991-3_991-2insC'
```

outputs

```
ADAM33:c.991-3_991-2insC     CCDS13058.1 (protein_coding)    ADAM33  -
   chr20:g.3654151dupG/c.991-3dupC/. inside_[intron_between_exon_10_and_11]
   CSQN=IntronicInsertion;left_align_gDNA=g.3654145_3654146insG;unalign_gDNA=g.3
   654145_3654146insG;left_align_cDNA=c.991-9_991-8insC;unalign_cDNA=c.991-3dupC
   ;source=CCDS
```

In the case of intronic insertions, amino acid identifier is not applicable, represented in a .. But cDNA and gDNA identifier are right-aligned according to their natural order, respecting HGVS nomenclature.

Insertion could occur to *splice sites*. TransVar identifies such cases and report splice site and repress translation of protein change.

```
$ transvar canno --ccds -i 'ADAM33:c.991_992insC'
```

results in

```
ADAM33:c.991_992insC CCDS13058.1 (protein_coding)    ADAM33  -
   chr20:g.3654142_3654143insG/c.991_992insC/.       inside_[cds_in_exon_11]
   CSQN=SpliceAcceptorInsertion;left_align_gDNA=g.3654142_3654143insG;unalign_gD
   NA=g.3654142_3654143insG;left_align_cDNA=c.991_992insC;unalign_cDNA=c.991_992
   insC;C2=acceptor_splice_site_on_exon_11_at_chr20:3654144_affected;source=CCDS
```

6.2.3 deletion

Similar to insertions, deletion can be in-frame or frame-shift. The consequence of deletion to amino acid sequence may appear a simple deletion or a block substitution (in the case where in-frame deletion is out of phase, i.e., partially delete codons).

Example: to annotate an **in-frame deletion**,

```
$ transvar canno --ccds -i 'A4GNT:c.694_696delTTG'
```

```
A4GNT:c.694_696delTTG        CCDS3097.1 (protein_coding)     A4GNT   -
   chr3:g.137843435_137843437delACA/c.694_696delTTG/p.L232delL       inside_[cds_in_
↪exon_2]
   CSQN=InFrameDeletion;left_align_gDNA=g.137843433_137843435delCAA;unaligned_gD
   NA=g.137843433_137843435delCAA;left_align_cDNA=c.692_694delTGT;unalign_cDNA=c
   .694_696delTTG;left_align_protein=p.L232delL;unalign_protein=p.L232delL;sourc
   e=CCDS
```

Example: to annotate a **in-frame, out-of-phase deletion**,

```
$ transvar canno --ccds -i 'ABHD15:c.431_433delGTG'
```

```
ABHD15:c.431_433delGTG       CCDS32602.1 (protein_coding)    ABHD15  -
   chr17:g.27893552_27893554delCAC/c.431_433delGTG/p.C144_V145delinsF        inside_
↪[cds_in_exon_1]
   CSQN=MultiAAMissense;left_align_gDNA=g.27893552_27893554delCAC;unaligned_gDNA
   =g.27893552_27893554delCAC;left_align_cDNA=c.431_433delGTG;unalign_cDNA=c.431
   _433delGTG;source=CCDS
```

Example: to annotate a **frame-shift deletion**,

```
$ transvar canno --ccds -i 'AADACL3:c.374delG'
```

```
AADACL3:c.374delG    CCDS41252.1 (protein_coding)    AADACL3 +
   chr1:g.12785494delG/c.374delG/p.C125Ffs*17        inside_[cds_in_exon_3]
   CSQN=Frameshift;left_align_gDNA=g.12785494delG;unaligned_gDNA=g.12785494delG;
   left_align_cDNA=c.374delG;unalign_cDNA=c.374delG;source=CCDS
```

Example: to annotate a **deletion that span from intronic to coding region**, protein prediction is suppressed due to loss of splice site.

```
$ transvar canno --ccds -i 'ABCB11:c.1198-8_1199delcactccagAA'
```

```
ABCB11:c.1198-8_1199delcactccagAA    CCDS46444.1 (protein_coding)    ABCB11  -
   chr2:g.169833196_169833205delTTCTGGAGTG/c.1198-8_1199delCACTCCAGAA/.      from_
↪[cds_in_exon_11]_to_[intron_between_exon_10_and_11]
   CSQN=SpliceAcceptorDeletion;left_align_gDNA=g.169833196_169833205delTTCTGGAGT
   G;unaligned_gDNA=g.169833196_169833205delTTCTGGAGTG;left_align_cDNA=c.1198-8_
   1199delCACTCCAGAA;unalign_cDNA=c.1198-8_1199delCACTCCAGAA;C2=acceptor_splice_
   site_on_exon_11_at_chr2:169833198_lost;source=CCDS
```

6.2.4 block substitution

Example: to annotate a block substitution in **coding region**,

```
$ transvar canno --ccds -i 'A1CF:c.508_509delinsTT'
```

```
A1CF:c.508_509delinsTT       CCDS7241.1 (protein_coding)     A1CF    -
   chr10:g.52595929_52595930delinsAA/c.508_509delinsTT/p.P170L       inside_[cds_in_
↪exon_4]
   CSQN=Missense;codon_cDNA=508-509-510;source=CCDS
A1CF:c.508_509delinsTT       CCDS7242.1 (protein_coding)     A1CF    -
   chr10:g.52595929_52595930delinsAA/c.508_509delinsTT/p.P170L       inside_[cds_in_
↪exon_4]
```

(continues on next page)

(continued from previous page)

```
   CSQN=Missense;codon_cDNA=508-509-510;source=CCDS
A1CF:c.508_509delinsTT       CCDS7243.1 (protein_coding)     A1CF     -
   chr10:g.52595953_52595954delinsAA/c.508_509delinsTT/p.G170F       inside_[cds_in_
↪exon_4]
   CSQN=Missense;codon_cDNA=508-509-510;source=CCDS
```

When performing annotation on block substitution, the reference and alternative sequence are double trimmed so that only the minimum stretch of substitution gets annotated excluding flanking sequence that are identical between reference and alternatives. Hence block substitution does not necessarily results in block substitution annotation. For example, the following substitution results in a deletion, where protein alternative alignment should be reported.

```
$ transvar canno --ccds -i 'CSRNP1:c.1212_1224delinsGGAGGAGGAA'
```

```
CSRNP1:c.1212_1224delinsGGAGGAGGAA   CCDS2682.1 (protein_coding)     CSRNP1   -
   chr3:g.39185102_39185104delTCC/c.1221_1223delGGA/p.E411delE       inside_[cds_in_
↪exon_4]
   CSQN=InFrameDeletion;left_align_gDNA=g.39185093_39185095delTCC;unaligned_gDNA
   =g.39185093_39185095delTCC;left_align_cDNA=c.1212_1214delGGA;unalign_cDNA=c.1
   221_1223delGGA;left_align_protein=p.E405delE;unalign_protein=p.E407delE;sourc
   e=CCDS
```

The following case reduces block substitution to SNV.

```
$ transvar canno -i 'CSRNP1:c.1230_1233delinsGCAA' --ccds
```

```
CSRNP1:c.1230_1233delinsGCAA CCDS2682.1 (protein_coding)     CSRNP1   -
   chr3:g.39185085C>G/c.1231G>C/p.E411Q      inside_[cds_in_exon_4]
   CSQN=Missense;reference_codon=GAA;alternative_codon=CAA;source=CCDS
```

And the following case reduces block substitution to SNP

```
$ transvar canno -i 'CSRNP1:c.1230_1233delinsGGCAA' --ccds --suspend --gseq
```

```
CSRNP1:c.1230_1233delinsGGCAA        CCDS2682.1 (protein_coding)     CSRNP1   -
   chr3:g.39185084_39185085insG/c.1231_1232insC/p.E411Afs*17 inside_[cds_in_exon_4]
   CSQN=Frameshift;left_align_gDNA=g.39185084_39185085insG;unalign_gDNA=g.391850
   84_39185085insG;left_align_cDNA=c.1231_1232insC;unalign_cDNA=c.1231_1232insC;
   source=CCDS       chr3    39185084        T       TG
```

Likewise, block substitution could occur to **intronic region**,

```
$ transvar canno --ccds -i 'A1CF:c.1460+2_1460+3delinsCC'
```

```
A1CF:c.1460+2_1460+3delinsCC CCDS7241.1 (protein_coding)     A1CF     -
   chr10:g.52570797_52570798delinsGG/c.1460+2_1460+3delinsCC/.       inside_[intron_
↪between_exon_9_and_10]
   CSQN=IntronicBlockSubstitution;source=CCDS
```

When block substitution occurs **across splice site**, TransVar put a tag in the info fields and does not predict amino acid change.

```
$ transvar canno --ccds -i 'A1CF:c.1459_1460+3delinsCC'
```

```
A1CF:c.1459_1460+3delinsCC   CCDS7241.1 (protein_coding)     A1CF    -
   chr10:g.52570797_52570801delinsGG/c.1459_1460+3delinsCC/. from_[intron_between_
↪exon_9_and_10]_to_[cds_in_exon_9]
   CSQN=SpliceDonorBlockSubstitution;C2=donor_splice_site_on_exon_9_at_chr10:525
   70799_lost;source=CCDS
```

6.2.5 duplication

Duplication can be thought of as special insertion where the inserted sequence is identical to the sequence flanking the breakpoint. Similar to insertion, the annotation of duplication may possess alternative alignment.

Example: to annotate a duplication coding region,

```
$ transvar canno --ccds -i 'CHD7:c.1669_1674dup'
```

```
CHD7:c.1669_1674dup  CCDS47865.1 (protein_coding)    CHD7    +
   chr8:g.61693564_61693569dupCCCGTC/c.1669_1674dup/p.P558_S559dupPS inside_[cds_in_
↪exon_2]
   CSQN=InFrameInsertion;left_align_gDNA=g.61693561_61693562insTCCCCG;unalign_gD
   NA=g.61693562_61693567dupTCCCCG;left_align_cDNA=c.1668_1669insTCCCCG;unalign_
   cDNA=c.1669_1674dupTCCCCG;left_align_protein=p.H556_S557insSP;unalign_protein
   =p.S557_P558dupSP;phase=0;source=CCDS
```

Example: a duplication on the nucleotide level may lead to frame-shift or block substitution on the amino acid level,

```
$ transvar canno --ccds -i 'CHD7:c.1668_1669dup'
```

```
CHD7:c.1668_1669dup  CCDS47865.1 (protein_coding)    CHD7    +
   chr8:g.61693561_61693562dupTT/c.1668_1669dup/p.S557Ffs*8  inside_[cds_in_exon_2]
   CSQN=Frameshift;left_align_gDNA=g.61693560_61693561insTT;unalign_gDNA=g.61693
   561_61693562dupTT;left_align_cDNA=c.1667_1668insTT;unalign_cDNA=c.1668_1669du
   pTT;source=CCDS
```

Example: to annotate a duplication in intronic region,

```
$ transvar canno --ccds -i 'CHD7:c.1666-5_1666-3dup'
```

```
CHD7:c.1666-5_1666-3dup      CCDS47865.1 (protein_coding)    CHD7    +
   chr8:g.61693554_61693556dupCTC/c.1666-5_1666-3dup/.       inside_[intron_between_
↪exon_1_and_2]
   CSQN=IntronicInsertion;left_align_gDNA=g.61693553_61693554insCTC;unalign_gDNA
   =g.61693554_61693556dupCTC;left_align_cDNA=c.1666-6_1666-5insCTC;unalign_cDNA
   =c.1666-5_1666-3dupCTC;source=CCDS
```

CHAPTER 7

Interpret consequence labels (CSQN)

For each genetic variant, TransVar assigns a consequence label with *CSQN* tag. The consequence label sometimes explains the behaviour of the output, e.g., the missing of protein level representation due to the loss of splice site.

The consequence label is in the following alphabet:

7.1 General

label	interpretation
Synonymous	Variation in protein-coding sequence
	results in the same protein sequence
Missense	Single or multiple amino acid
	substitution to coding gene (1-1)
MultiAAMissense	In-frame multiple amino acid replacement
	(m to n, either m>1 or n>1)
Nonsense	Introduction of stop codon by single,
	multiple amino acid substitution or
	in-frame insertions/deletions

7.2 Coding Start/Stop

label	interpretation
CdsStartSNV	SNV at coding start
CdsStopSNV	SNV at coding stop
CdsStartDeletion	deletion of coding start
CdsStopDeletion	deletion of coding stop

7.3 Coding Insertion/Deletion

label	interpretation
Frameshift	Frameshift mutation to a coding gene
InFrameDeletion	In-frame deletion to a coding gene
InFrameInsertion	In-frame insertion to a coding gene

7.4 Intronic

label	interpretation
IntronicSNV	Intronic single nucleotide variation
IntronicDeletion	Intronic deletion
IntronicInsertion	Intronic insertion
IntronicBlockSubstitutio	Intronic block substitution

7.5 Intergenic

label	interpretation
IntergenicSNV	Intergenic single nucleotide variation
IntergenicDeletion	Intergenic deletion
IntergenicInsertion	Intergenic insertion
IntergenicBlockSubstitution	Intergenic block substitution

7.6 Splice site

label	interpretation
SpliceDonorDeletion	Deletion occurs to splice donor
SpliceAcceptorDeletion	Deletion occurs to splice acceptor
SpliceDonorSNV	Genetic variation at splice donor
SpliceAcceptorSNV	Genetic variation at splice acceptor
SpliceDonorBlockSubstitution	Block substitution occurs at splice donor
SpliceAcceptorBlockSubstitution	Block substitution occurs at splice acceptor
SpliceDonorInsertion	Insertion at splice donor
SpliceAcceptorInsertion	Insertion at splice acceptor

7.7 Others

label	interpretation
Unclassified	Unclassified

CHAPTER 8

Inspect variant sequences

The **–print-protein** and **–print-protein-pretty** options displays the full variant protein sequence in the **variant_protein_seq** field of the info when the genomic variant hits a protein-coding transcript.

8.1 Missense substitution

```
$ transvar ganno -i 'chr1:g.115256530G>A' --ensembl --print-protein
```

```
chr1:g.115256530G>A  ENST00000369535 (protein_coding)        NRAS    -
   chr1:g.115256530G>A/c.181C>T/p.Q61*       inside_[cds_in_exon_3]
   CSQN=Nonsense;variant_protein_seq=MTEYKLVVVGAGGVGKSALTIQLIQNHFVDEYDPTIEDSYRKQ
   VVIDGETCLLDILDTAG*;codon_pos=115256528-115256529-115256530;ref_codon_seq=CAA;
   aliases=ENSP00000358548;source=Ensembl
```

–print-protein-pretty output is more human-readable and highlight the mutation in brackets.

```
$ transvar ganno --ccds -i 'chr3:g.178936091G>A' --print-protein-pretty
```

```
chr3:g.178936091G>A  CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.E545K     inside_[cds_in_exon_9]
   CSQN=Missense;dbsnp=rs104886003(chr3:178936091G>A);variant_protein_seq=MPPRPS
   SGELWGIHLMPPRILVECLLPNGMIVTLECLREATLITIKHELFKEARKYPLHQLLQDESSYIFVSVTQEAEREEFF
   DETRRLCDLRLFQPFLKVIEPVGNREEKILNREIGFAIGMPVCEFDMVKDPEVQDFRRNILNVCKEAVDLRDLNSPH
   SRAMYVYPPNVESSPELPKHIYNKLDKGQIIVVIWVIVSPNNDKQKYTLKINHDCVPEQVIAEAIRKKTRSMLLSSE
   QLKLCVLEYQGKYILKVCGCDEYFLEKYPLSQYKYIRSCIMLGRMPNLMLMAKESLYSQLPMDCFTMPSYSRRISTA
   TPYMNGETSTKSLWVINSALRIKILCATYVNVNIRDIDKIYVRTGIYHGGEPLCDNVNTQRVPCSNPRWNEWLNYDI
   YIPDLPRAARLCLSICSVKGRKGAKEEHCPLAWGNINLFDYTDTLVSGKMALNLWPVPHGLEDLLNPIGVTGSNPNK
   ETPCLELEFDWFSSVVKFPDMSVIEEHANWSVSREAGFSYSHAGLSNRLARDNELRENDKEQLKAISTRDPLSEIT_
   _[E>K]__QEKDFLWSHRHYCVTIPEILPKLLLSVKWNSRDEVAQMYCLVKDWPPIKPEQAMELLDCNYPDPMVRGF
   AVRCLEKYLTDDKLSQYLIQLVQVLKYEQYLDNLLVRFLLKKALTNQRIGHFFFWHLKSEMHNKTVSQRFGLLLESY
   CRACGMYLKHLNRQVEAMEKLINLTDILKQEKKDETQKVQMKFLVEQMRRPDFMDALQGFLSPLNPAHQLGNLRLEE
   CRIMSSAKRPLWLNWENPDIMSELLFQNNEIIFKNGDDLRQDMLTLQIIRIMENIWQNQGLDLRMLPYGCLSIGDCV
```

(continues on next page)

(continued from previous page)

```
   GLIEVVRNSHTIMQIQCKGGLKGALQFNSHTLHQWLKDKNKGEIYDAAIDLFTRSCAGYCVATFILGIGDRHNSNIM
   VKDDGQLFHIDFGHFLDHKKKKFGYKRERVPFVLTQDFLIVISKGAQECTKTREFERFQEMCYKAYLAIRQHANLFI
   NLFSMMLGSGMPELQSFDDIAYIRKTLALDKTEQEALEYFMKQMNDAHHGGWTTKMDWIFHTIKQHALN*;codon_
   pos=178936091-178936092-178936093;ref_codon_seq=GAG;source=CCDS
```

The alphabet transformation option **–aa3** applies here as well.

```
$ transvar ganno -i 'chr1:g.115256530G>A' --ensembl --print-protein-pretty --aa3
```

```
chr1:g.115256530G>A  ENST00000369535 (protein_coding)        NRAS    -
   chr1:g.115256530G>A/c.181C>T/p.Gln61X     inside_[cds_in_exon_3]
   CSQN=Missense;variant_protein_seq=MetThrGluTyrLysLeuValValValGlyAlaGlyGlyValG
   lyLysSerAlaLeuThrIleGlnLeuIleGlnAsnHisPheValAspGluTyrAspProThrIleGluAspSerTyr
   ArgLysGlnValValIleAspGlyGluThrCysLeuLeuAspIleLeuAspThrAlaGly__[GluGluTyrSerAl
   aMetArgAspGlnTyrMetArgThrGlyGluGlyPheLeuCysValPheAlaIleAsnAsnSerLysSerPheAlaA
   spIleAsnLeuTyrArgGluGlnIleLysArgValLysAspSerAspAspValProMetValLeuValGlyAsnLys
   CysAspLeuProThrArgThrValAspThrLysGlnAlaHisGluLeuAlaLysSerTyrGlyIleProPheIleGl
   uThrSerAlaLysThrArgGlnGlyValGluAspAlaPheTyrThrLeuValArgGluIleArgGlnTyrArgMetL
   ysLysLeuAsnSerSerAspAspGlyThrGlnGlyCysMetGlyLeuProCysValValMet>X];codon_pos=1
   15256528-115256529-115256530;ref_codon_seq=CAA;aliases=ENSP00000358548;source
   =Ensembl
```

8.2 Deletion

```
$ transvar canno --ccds -i 'CCDS8856:c.769_771delGGG' --print-protein-pretty
```

```
CCDS8856:c.769_771delGGG     CCDS8856.1 (protein_coding)     AAAS    -
   chr12:g.53703427_53703429delCCC/c.769_771delGGG/p.G257delG        inside_[cds_in_
↪exon_8]
   CSQN=InFrameDeletion;left_align_gDNA=g.53703424_53703426delCCC;unaligned_gDNA
   =g.53703424_53703426delCCC;left_align_cDNA=c.766_768delGGG;unalign_cDNA=c.769
   _771delGGG;left_align_protein=p.G256delG;unalign_protein=p.G257delG;variant_p
   rotein_seq=MCSLGLFPPPPPRGQVTLYEHNNELVTGSSYESPPPDFRGQWINLPVLQLTKDPLKTPGRLDHGTR
   TAFIHHREQVWKRCINIWRDVGLFGVLNEIANSEEEVFEWVKTASGWALALCRWASSLHGSLFPHLSLRSEDLIAEF
   AQVTNWSSCCLRVFAWHPHTNKFAVALLDDSVRVYNASSTIVPSLKHRLQRNVASLAWKPLSASVLAVACQSCILIW
   TLDPTSLSTRPSSGCAQVLSHPGHTPVTSLAWAPSG__[G_deletion]__RLLSASPVDAAIRVWDVSTETCVPL
   PWFRGGGVTNLLWSPDGSKILATTPSAVFRVWEAQMWTCERWPTLSGRCQTGCWSPDGSRLLFTVLGEPLIYSLSFP
   ERCGEGKGCVGGAKSATIVADLSETTIQTPDGEERLGGEAHSMVWDPSGERLAVLMKGKPRVQDGKPVILLFRTRNS
   PVFELLPCGIIQGEPGAQPQLITFHPSFNKGALLSVGWSTGRIAHIPLYFVNAQFPRFSPVLGRAQEPPAGGGGSIH
   DLPLFTETSPTSAPWDPLPGPPPVLPHSPHSHL*;source=CCDS
```

8.3 Insertion

```
$ transvar ganno -i 'chr2:g.69741762_69741763insTGC' --ccds --print-protein-pretty
```

```
chr2:g.69741762_69741763insTGC       CCDS1893.2 (protein_coding)     AAK1    -
   chr2:g.69741780_69741782dupCTG/c.1614_1616dupGCA/p.Q546dupQ       inside_[cds_in_
↪exon_12]
   CSQN=InFrameInsertion;left_align_gDNA=g.69741762_69741763insTGC;unalign_gDNA=
   g.69741762_69741763insTGC;left_align_cDNA=c.1596_1597insCAG;unalign_cDNA=c.16
```

(continues on next page)

(continued from previous page)

```
   14_1616dupGCA;left_align_protein=p.Y532_Q533insQ;unalign_protein=p.Q539dupQ;v
   ariant_protein_seq=MKKFFDSRREQGGSGLGSGSSGGGGSTSGLGSGYIGRVFGIGRQQVTVDEVLAEGGFA
   IVFLVRTSNGMKCALKRMFVNNEHDLQVCKREIQIMRDLSGHKNIVGYIDSSINNVSSGDVWEVLILMDFCRGGQVV
   NLMNQRLQTGFTENEVLQIFCDTCEAVARLHQCKTPIIHRDLKVENILLHDRGHYVLCDFGSATNKFQNPQTEGVNA
   VEDEIKKYTTLSYRAPEMVNLYSGKIITTKADIWALGCLLYKLCYFTLPFGESQVAICDGNFTIPDNSRYSQDMHCL
   IRYMLEPDPDKRPDIYQVSYFSFKLLKKECPIPNVQNSPIPAKLPEPVKASEAAAKKTQPKARLTDPIPTTETSIAP
   RQRPKAGQTQPNPGILPIQPALTPRKRATVQPPPQAAGSSNQPGLLASVPQPKPQAPPSQPLPQTQAKQPQAPPTPQ
   QTPSTQAQGLPAQAQATPQHQQQLFLKQQQQQQQPPPAQQQPAGTFYQQQQAQTQQFQAVHPATQKPAIAQFPVVSQ
   GGSQQQLMQNFYQQQQQQQQQQQQQQ__[insert_Q]__LATALHQQQLMTQQAALQQKPTMAAGQQPQPQPAAAP
   QPAPAQEPAIQAPVRQQPKVQTTPPPAVQGQKVGSLTPPSSPKTQRAGHRRILSDVTHSAVFGVPASKSTQLLQAAA
   AEASLNKSKSATTTPSGSPRTSQQNVYNPSEGSTWNPFDDDNFSKLTAEELLNKDFAKLGEGKHPEKLGGSAESLIP
   GFQSTQGDAFATTSFSAGTAEKRKGGQTVDSGLPLLSVSDPFIPLQVPDAPEKLIEGLKSPDTSLLLPDLLPMTDPF
   GSTSDAVIEKADVAVESLIPGLEPPVPQRLPSQTESVTSNRTDSLTGEDSLLDCSLLSNPTTDLLEEFAPTAISAPV
   HKAAEDSNLISGFDVPEGSDKVAEDEFDPIPVLITKNPQGGHSRNSSGSSESSLPNLARSLLLVDQLIDL*;phase
   =2;source=CCDS
```

8.4 Block substitution

```
$ transvar ganno -i "chr2:g.234183372_234183383del" --ccds --print-protein-pretty
```

```
chr2:g.234183372_234183383del        CCDS2502.2 (protein_coding)     ATG16L1 +
   chr2:g.234183372_234183383del12/c.845_856del12/p.H282_G286delinsR inside_[cds_in_
↪exon_8]
   CSQN=MultiAAMissense;left_align_gDNA=g.234183372_234183383del12;unaligned_gDN
   A=g.234183372_234183383del12;left_align_cDNA=c.845_856del12;unalign_cDNA=c.84
   5_856del12;variant_protein_seq=MSSGLRAADFPRWKRHISEQLRRRDRLQRQAFEEIILQYNKLLEKS
   DLHSVLAQKLQAEKHDVPNRHEISPGHDGTWNDNQLQEMAQLRIKHQEELTELHKKRGELAQLVIDLNNQMQRKDRE
   MQMNEAKIAECLQTISDLETECLDLRTKLCDLERANQTLKDEYDALQITFTALEGKLRKTTEENQELVTRWMAEKAQ
   EANRLNAENEKDSRRRQARLQKELAEAAKEPLPVEQDDDIEVIVDETSDHTEETSPVRAISRAATRRSVSSFPVPQD
   NVDT__[HPGSG>R]__KEVRVPATALCVFDAHDGEVNAVQFSPGSRLLATGGMDRRVKLWEVFGEKCEFKGSLSGS
   NAGITSIEFDSAGSYLLAASNDFASRIWTVDDYRLRHTLTGHSGKVLSAKFLLDNARIVSGSHDRTLKLWDLRSKVC
   IKTVFAGSSCNDIVCTEQCVMSGHFDKKIRFWDIRSESIVREMELLGKITALDLNPERTELLSCSRDDLLKVIDLRT
   NAIKQTFSAPGFKCGSDWTRVVFSPDGSYVAAGSAEGSLYIWSVLTGKVEKVLSKQHSSSINAVAWSPSGSHVVSVD
   KGCKAVLWAQY*;source=CCDS
chr2:g.234183372_234183383del        CCDS2503.2 (protein_coding)     ATG16L1 +
   chr2:g.234183372_234183383del12/c.902_913del12/p.H301_G305delinsR inside_[cds_in_
↪exon_9]
   CSQN=MultiAAMissense;left_align_gDNA=g.234183372_234183383del12;unaligned_gDN
   A=g.234183372_234183383del12;left_align_cDNA=c.902_913del12;unalign_cDNA=c.90
   2_913del12;variant_protein_seq=MSSGLRAADFPRWKRHISEQLRRRDRLQRQAFEEIILQYNKLLEKS
   DLHSVLAQKLQAEKHDVPNRHEISPGHDGTWNDNQLQEMAQLRIKHQEELTELHKKRGELAQLVIDLNNQMQRKDRE
   MQMNEAKIAECLQTISDLETECLDLRTKLCDLERANQTLKDEYDALQITFTALEGKLRKTTEENQELVTRWMAEKAQ
   EANRLNAENEKDSRRRQARLQKELAEAAKEPLPVEQDDDIEVIVDETSDHTEETSPVRAISRAATKRLSQPAGGLLD
   SITNIFGRRSVSSFPVPQDNVDT__[HPGSG>R]__KEVRVPATALCVFDAHDGEVNAVQFSPGSRLLATGGMDRRV
   KLWEVFGEKCEFKGSLSGSNAGITSIEFDSAGSYLLAASNDFASRIWTVDDYRLRHTLTGHSGKVLSAKFLLDNARI
   VSGSHDRTLKLWDLRSKVCIKTVFAGSSCNDIVCTEQCVMSGHFDKKIRFWDIRSESIVREMELLGKITALDLNPER
   TELLSCSRDDLLKVIDLRTNAIKQTFSAPGFKCGSDWTRVVFSPDGSYVAAGSAEGSLYIWSVLTGKVEKVLSKQHS
   SSINAVAWSPSGSHVVSVDKGCKAVLWAQY*;source=CCDS
chr2:g.234183372_234183383del        CCDS54438.1 (protein_coding)    ATG16L1 +
   chr2:g.234183372_234183383del12/c.413_424del12/p.H138_G142delinsR inside_[cds_in_
↪exon_5]
   CSQN=MultiAAMissense;left_align_gDNA=g.234183372_234183383del12;unaligned_gDN
   A=g.234183372_234183383del12;left_align_cDNA=c.413_424del12;unalign_cDNA=c.41
```

(continues on next page)

(continued from previous page)

```
   3_424del12;variant_protein_seq=MSSGLRAADFPRWKRHISEQLRRRDRLQRQAFEEIILQYNKLLEKS
   DLHSVLAQKLQAEKHDVPNRHEIRRRQARLQKELAEAAKEPLPVEQDDDIEVIVDETSDHTEETSPVRAISRAATRR
   SVSSFPVPQDNVDT__[HPGSG>R]__KEVRVPATALCVFDAHDGEVNAVQFSPGSRLLATGGMDRRVKLWEVFGEK
   CEFKGSLSGSNAGITSIEFDSAGSYLLAASNDFASRIWTVDDYRLRHTLTGHSGKVLSAKFLLDNARIVSGSHDRTL
   KLWDLRSKVCIKTVFAGSSCNDIVCTEQCVMSGHFDKKIRFWDIRSESIVREMELLGKITALDLNPERTELLSCSRD
   DLLKVIDLRTNAIKQTFSAPGFKCGSDWTRVVFSPDGSYVAAGSAEGSLYIWSVLTGKVEKVLSKQHSSSINAVAWS
   PSGSHVVSVDKGCKAVLWAQY*;source=CCDS
```

8.5 Frameshift sequence

```
$ transvar canno --ccds -i 'CCDS8856:c.769_770delGG' --print-protein-pretty
```

```
CCDS8856:c.769_770delGG      CCDS8856.1 (protein_coding)     AAAS     -
   chr12:g.53703428_53703429delCC/c.770_771delGG/p.G257Afs*65        inside_[cds_in_
↪exon_8]
   CSQN=Frameshift;left_align_gDNA=g.53703424_53703425delCC;unaligned_gDNA=g.537
   03425_53703426delCC;left_align_cDNA=c.766_767delGG;unalign_cDNA=c.769_770delG
   G;variant_protein_seq=MCSLGLFPPPPPRGQVTLYEHNNELVTGSSYESPPPDFRGQWINLPVLQLTKDPL
   KTPGRLDHGTRTAFIHHREQVWKRCINIWRDVGLFGVLNEIANSEEEVFEWVKTASGWALALCRWASSLHGSLFPHL
   SLRSEDLIAEFAQVTNWSSCCLRVFAWHPHTNKFAVALLDDSVRVYNASSTIVPSLKHRLQRNVASLAWKPLSASVL
   AVACQSCILIWTLDPTSLSTRPSSGCAQVLSHPGHTPVTSLAWAPSG__[frameshift_GRLLSASPVDAAIRVW
   DVSTETCVPLPWFRGGGVTNLLWSPDGSKILATTPSAVFRVWEAQMWTCERWPTLSGRCQTGCWSPDGSRLLFTVLG
   EPLIYSLSFPERCGEGKGCVGGAKSATIVADLSETTIQTPDGEERLGGEAHSMVWDPSGERLAVLMKGKPRVQDGKP
   VILLFRTRNSPVFELLPCGIIQGEPGAQPQLITFHPSFNKGALLSVGWSTGRIAHIPLYFVNAQFPRFSPVLGRAQE
   PPAGGGGSIHDLPLFTETSPTSAPWDPLPGPPPVLPHSPHSHL*>AAALSFTRGCCYPGMGCLNRDLCPPSLVPRRW
   GDQPALVPRRQQNPGYHSFSCLSSLGGPDVDL*];source=CCDS
```

```
$ transvar canno -i 'CCDS54438:c.409_421del' --ccds --print-protein-pretty
```

```
CCDS54438:c.409_421del       CCDS54438.1 (protein_coding)    ATG16L1 +
   chr2:g.234183368_234183380del13/c.409_421del13/p.T137Lfs*5        inside_[cds_in_
↪exon_5]
   CSQN=Frameshift;left_align_gDNA=g.234183367_234183379del13;unaligned_gDNA=g.2
   34183368_234183380del13;left_align_cDNA=c.408_420del13;unalign_cDNA=c.409_421
   del13;variant_protein_seq=MSSGLRAADFPRWKRHISEQLRRRDRLQRQAFEEIILQYNKLLEKSDLHSV
   LAQKLQAEKHDVPNRHEIRRRQARLQKELAEAAKEPLPVEQDDDIEVIVDETSDHTEETSPVRAISRAATRRSVSSF
   PVPQDNVD__[frameshift_THPGSGKEVRVPATALCVFDAHDGEVNAVQFSPGSRLLATGGMDRRVKLWEVFGE
   KCEFKGSLSGSNAGITSIEFDSAGSYLLAASNDFASRIWTVDDYRLRHTLTGHSGKVLSAKFLLDNARIVSGSHDRT
   LKLWDLRSKVCIKTVFAGSSCNDIVCTEQCVMSGHFDKKIRFWDIRSESIVREMELLGKITALDLNPERTELLSCSR
   DDLLKVIDLRTNAIKQTFSAPGFKCGSDWTRVVFSPDGSYVAAGSAEGSLYIWSVLTGKVEKVLSKQHSSSINAVAW
   SPSGSHVVSVDKGCKAVLWAQY*>LVKK*];source=CCDS
```

CHAPTER 9

Using non-canonical IDs

TransVar provides the use of non-canonical IDs by the means of ID mapping. This is achieved by providing an ID mapping file.

9.1 Create ID Mapping File

One can create an ID Mapping file by indexing a tab-delimited file with "synonym"(noncanonical ID) in the first column and canonical ID in the second column. The content of such tab-delimited file looks like

```
MLL2    KMT2D
```

And to create a ID mapping index

```
transvar index --idmap [file_name] -o test.idmap_idx
```

Now you can use *–idmap* option in annotation to get annotation of non-canonical ID mapping

```
transvar panno -i 'MLL2:p.Asp5492Asn' --ensembl --idmap test.idmap_idx
```

```
MLL2:p.Asp5492Asn       ENST00000301067 (protein_coding)        KMT2D   -
chr12:g.49415873C>T/c.16474G>A/p.D5492N inside_[cds_in_exon_53]
CSQN=Missense;reference_codon=GAC;candidate_codons=AAC,AAT;candidate_mnv_varian
ts=chr12:g.49415871_49415873delGTCinsATT;aliases=ENSP00000301067;source=Ensembl
```

You can see now TransVar can identify MLL2 which is a noncanonical ID in addition to the standard KMT2D.

Inherently, if you name the generated ID mapping to *[path_to_transvardb].XXX.idmap_idx*. You can use the shortcut of *–idmap XXX* as long as the annotation transcript database is provided. For example, *–idmap HGNC* when used with *–ensembl path_to_ensembl.transvardb* will also look for a ID mapping file of name *path_to_ensembl.transvardb.HGNC.idmap_idx*.

CHAPTER 10

Output Options

10.1 VCF-like output

With *–gseq* transvar appends genomic sequence information as additional columns with pos, ref, alt following the VCF convention (i.e., indels are left-aligned)

```
$ transvar canno -i 'MRE11A:c.592_593delGTinsTA' --ensembl --gseq
```

```
MRE11A:c.592_593delGTinsTA   ENST00000323929 (protein_coding)        MRE11A  -
   chr11:g.94209521_94209522delinsTA/c.592_593delinsTA/p.V198*       inside_[cds_in_
↪exon_7]
   CSQN=Missense;codon_cDNA=592-593-594;aliases=ENSP00000325863;source=Ensembl       c
   hr11      94209520        TAC     TTA
MRE11A:c.592_593delGTinsTA   ENST00000323977 (protein_coding)        MRE11A  -
   chr11:g.94209521_94209522delinsTA/c.592_593delinsTA/p.V198*       inside_[cds_in_
↪exon_7]
   CSQN=Missense;codon_cDNA=592-593-594;aliases=ENSP00000326094;source=Ensembl       c
   hr11      94209520        TAC     TTA
MRE11A:c.592_593delGTinsTA   ENST00000393241 (protein_coding)        MRE11A  -
   chr11:g.94209521_94209522delinsTA/c.592_593delinsTA/p.V198*       inside_[cds_in_
↪exon_7]
   CSQN=Missense;codon_cDNA=592-593-594;aliases=ENSP00000376933;source=Ensembl       c
   hr11      94209520        TAC     TTA
MRE11A:c.592_593delGTinsTA   ENST00000540013 (protein_coding)        MRE11A  -
   chr11:g.94209521_94209522delinsTA/c.592_593delinsTA/p.V198*       inside_[cds_in_
↪exon_7]
   CSQN=Missense;codon_cDNA=592-593-594;aliases=ENSP00000440986;source=Ensembl       c
   hr11      94209520        TAC     TTA
```

Another example of deletion

```
$ transvar ganno -i "chr2:g.234183368_234183379del" --ccds --gseq --seqmax 200
```

```
chr2:g.234183368_234183379del        CCDS2502.2 (protein_coding)     ATG16L1 +
   chr2:g.234183368_234183379delACTCATCCTGGT/c.841_852delACTCATCCTGGT/p.T281_
↪G284delTHPG     inside_[cds_in_exon_8]
   CSQN=InFrameDeletion;left_align_gDNA=g.234183367_234183378delTACTCATCCTGG;una
   ligned_gDNA=g.234183368_234183379delACTCATCCTGGT;left_align_cDNA=c.840_851del
   TACTCATCCTGG;unalign_cDNA=c.841_852delACTCATCCTGGT;left_align_protein=p.T281_
   G284delTHPG;unalign_protein=p.T281_G284delTHPG;source=CCDS        chr2    ␣
↪234183366       ATA
   CTCATCCTGG        A
chr2:g.234183368_234183379del        CCDS2503.2 (protein_coding)     ATG16L1 +
   chr2:g.234183368_234183379delACTCATCCTGGT/c.898_909delACTCATCCTGGT/p.T300_
↪G303delTHPG     inside_[cds_in_exon_9]
   CSQN=InFrameDeletion;left_align_gDNA=g.234183367_234183378delTACTCATCCTGG;una
   ligned_gDNA=g.234183368_234183379delACTCATCCTGGT;left_align_cDNA=c.897_908del
   TACTCATCCTGG;unalign_cDNA=c.898_909delACTCATCCTGGT;left_align_protein=p.T300_
   G303delTHPG;unalign_protein=p.T300_G303delTHPG;source=CCDS        chr2    ␣
↪234183366       ATA
   CTCATCCTGG        A
chr2:g.234183368_234183379del        CCDS54438.1 (protein_coding)    ATG16L1 +
   chr2:g.234183368_234183379delACTCATCCTGGT/c.409_420delACTCATCCTGGT/p.T137_
↪G140delTHPG     inside_[cds_in_exon_5]
   CSQN=InFrameDeletion;left_align_gDNA=g.234183367_234183378delTACTCATCCTGG;una
   ligned_gDNA=g.234183368_234183379delACTCATCCTGGT;left_align_cDNA=c.408_419del
   TACTCATCCTGG;unalign_cDNA=c.409_420delACTCATCCTGGT;left_align_protein=p.T137_
   G140delTHPG;unalign_protein=p.T137_G140delTHPG;source=CCDS        chr2    ␣
↪234183366       ATA
   CTCATCCTGG        A
```

CHAPTER 11

FAQ

11.1 How to batch-process?

For all mutation types, one can batch process a list of mutation identifiers with optional transcript id to constraint the search. Take SNV for example,

```
transvar panno -l example/input_table -g 1 -m 5 -t 2 --ensembl -o 2,3,4
```

As suggested by the command, TransVar takes as input the 1st column as gene and 4th column as identifier. The 2nd column will be used as the transcript id from Ensembl to constrain the alternative identifier search. The 2nd, 3rd and 5th columns are chosen to be output as a validation of TransVar's performance.

Input:

```
ADAMTSL3        ENST00000286744 15:84442328     c.243G>A        p.W81*  Nonsense
ADAMTSL3        ENST00000286744 15:84442326     c.241T>C        p.W81R  Missense
ADAMTSL4        ENST00000369038 1:150530513     c.2270G>A       p.G757D Missense
ADCY2   ENST00000338316 5:7802364       c.2662G>A       p.V888I Missense
ADCY2   ENST00000338316 5:7802365       c.2663T>C       p.V888A Missense
```

Output:

```
ENST00000286744|15:84442328|c.243G>A ENST00000286744 (protein_coding)        ADAMTSL3␣
↪       +
   chr15:g.84442327G>A/c.242G>A/p.W81*       cds_in_exon_4
   reference_codon=TGG;candidate_codons=TAA,TAG,TGA;candidate_snv_variants=chr15
   :g.84442328G>A;candidate_mnv_variants=chr15:g.84442327_84442328delGGinsAA;mis
   sense;aliases=ENSP00000286744;source=Ensembl
ENST00000286744|15:84442326|c.241T>C ENST00000286744 (protein_coding)        ADAMTSL3␣
↪       +
   chr15:g.84442326T>A/c.241T>A/p.W81R       cds_in_exon_4
   reference_codon=TGG;candidate_codons=AGG,AGA,CGA,CGC,CGG,CGT;candidate_snv_va
   riants=chr15:g.84442326T>C;candidate_mnv_variants=chr15:g.84442326_84442328de
   lTGGinsAGA,chr15:g.84442326_84442328delTGGinsCGA,chr15:g.84442326_84442328del
```

(continues on next page)

(continued from previous page)

```
   TGGinsCGC,chr15:g.84442326_84442328delTGGinsCGT;missense;aliases=ENSP00000286
   744;source=Ensembl
ENST00000369038|1:150530513|c.2270G>A        ENST00000369038 (protein_coding)
↪ADAMTSL4        +
   chr1:g.150530513G>A/c.2270G>A/p.G757D     cds_in_exon_12
   reference_codon=GGT;candidate_codons=GAC,GAT;candidate_mnv_variants=chr1:g.15
   0530513_150530514delGTinsAC;missense;aliases=ENSP00000358034;source=Ensembl
ENST00000338316|5:7802364|c.2662G>A  ENST00000338316 (protein_coding)        ADCY2   +
   chr5:g.7802364G>A/c.2662G>A/p.V888I       cds_in_exon_21
   reference_codon=GTC;candidate_codons=ATC,ATA,ATT;candidate_mnv_variants=chr5:
   g.7802364_7802366delGTCinsATA,chr5:g.7802364_7802366delGTCinsATT;missense;ali
   ases=ENSP00000342952;source=Ensembl
ENST00000338316|5:7802365|c.2663T>C  ENST00000338316 (protein_coding)        ADCY2   +
   chr5:g.7802365T>C/c.2663T>C/p.V888A       cds_in_exon_21
   reference_codon=GTC;candidate_codons=GCA,GCC,GCG,GCT;candidate_mnv_variants=c
   hr5:g.7802365_7802366delTCinsCA,chr5:g.7802365_7802366delTCinsCG,chr5:g.78023
   65_7802366delTCinsCT;missense;aliases=ENSP00000342952;source=Ensembl
```

11.2 How to use VCF as input?

TransVar can take VCF as input when annotating from genomic level.

```
transvar ganno --vcf ALL.wgs.phase1_release_v3.20101123.snps_indel_sv.sites.vcf.gz --
↪ccds
# or
transvar ganno --vcf demo.1kg.vcf --ccds
```

11.3 How to automatically decompose a haplotype into multiple mutations?

TransVar performs local alignment to allow long haplotype to be decomposed into multiple mutations.

```
$ transvar ganno --ccds -i 'chr20:g.645097_645111delinsGTGCGATACCCAGGAG' --haplotype
```

leads to 2 snv and one insertion

```
chr20:g.645097_645111delinsGTGCGATACCCAGGAG  CCDS13006.1 (protein_coding)    SCRT2   -
   chr20:g.645098G>T/c.141C>A/p.A47A inside_[cds_in_exon_2]
   CSQN=Synonymous;codon_pos=645098-645099-645100;ref_codon_seq=GCC;source=CCDS
chr20:g.645097_645111delinsGTGCGATACCCAGGAG  CCDS13006.1 (protein_coding)    SCRT2   -
   chr20:g.645101_645102insA/c.137_138insT/p.A47Rfs*350      inside_[cds_in_exon_2]
   CSQN=Frameshift;left_align_gDNA=g.645101_645102insA;unalign_gDNA=g.645101_645
   102insA;left_align_cDNA=c.137_138insT;unalign_cDNA=c.137_138insT;source=CCDS
chr20:g.645097_645111delinsGTGCGATACCCAGGAG  CCDS13006.1 (protein_coding)    SCRT2   -
   chr20:g.645107T>A/c.134-2A>T/.    inside_[intron_between_exon_1_and_2]
   CSQN=SpliceAcceptorSNV;C2=SpliceAcceptorOfExon1_At_chr20:645106;source=CCDS
```

11.4 How to use 3-letter code instead of 1-letter code for protein?

TransVar automatically infer whether the input is a 3-letter code or 1-letter code. The output is default to 1-letter code. But can be switched to 3-letter code through the *–aa3* option. For example,

```
$ transvar panno --ccds -i 'PIK3CA:p.Glu545Lys' --aa3
```

```
PIK3CA:p.Glu545Lys   CCDS43171.1 (protein_coding)    PIK3CA  +
   chr3:g.178936091G>A/c.1633G>A/p.Glu545Lys inside_[cds_in_exon_9]
   CSQN=Missense;reference_codon=GAG;candidate_codons=AAG,AAA;candidate_mnv_vari
   ants=chr3:g.178936091_178936093delGAGinsAAA;dbsnp=rs104886003(chr3:178936091G
   >A);source=CCDS
```

11.5 How can I let TransVar output sequence context?

The option `--aacontext 5` output +/- 5bp protein sequence context.

```
$ transvar ganno -i 'chr17:7577124' --ccds --aacontext 5
```

```
chr17:7577124        CCDS11118.1 (protein_coding)    TP53    -
   chr17:g.7577124C>/c.814G>/p.V272  inside_[cds_in_exon_7]
   is_gene_body;aacontext=RNSFE[V]RVCAC;codon_pos=7577122-7577123-7577124;source
   =CCDS
chr17:7577124        CCDS45605.1 (protein_coding)    TP53    -
   chr17:g.7577124C>/c.814G>/p.V272  inside_[cds_in_exon_7]
   is_gene_body;aacontext=RNSFE[V]RVCAC;codon_pos=7577122-7577123-7577124;source
   =CCDS
chr17:7577124        CCDS45606.1 (protein_coding)    TP53    -
   chr17:g.7577124C>/c.814G>/p.V272  inside_[cds_in_exon_7]
   is_gene_body;aacontext=RNSFE[V]RVCAC;codon_pos=7577122-7577123-7577124;source
   =CCDS
```

shows the protein sequence context in the aacontext tag.

11.6 How to report results in one line for each query?

Use `--oneline` option. This separates the outputs from each transcript by '|||'.

11.7 I got 'gene_not_recognized', what's wrong?

Most likely you forgot to specify a transcipt definition such as `--ccds` or `--ensembl`. Sometimes there are non-canonical names for genes, this can be fixed through the `--alias` option and specify an alias table. TransVar comes with alias table from UCSC knownGene.

11.8 Does TransVar support alternative format for MNV such as c.508_509CC>TT?

Yes, but only in input. For example, **c.508_509CC>TT**

```
$ transvar canno --ccds -i 'A1CF:c.508_509CC>TT'
```

```
A1CF:c.508_509CC>TT  CCDS7241.1 (protein_coding)     A1CF    -
   chr10:g.52595929_52595930delinsAA/c.508_509delinsTT/p.P170L       inside_[cds_in_
↪exon_4]
   CSQN=Missense;codon_cDNA=508-509-510;source=CCDS
A1CF:c.508_509CC>TT  CCDS7242.1 (protein_coding)     A1CF    -
   chr10:g.52595929_52595930delinsAA/c.508_509delinsTT/p.P170L       inside_[cds_in_
↪exon_4]
   CSQN=Missense;codon_cDNA=508-509-510;source=CCDS
```

11.9 Does TransVar support relaxed input without 'g.', 'c.' and 'p.'?

Yes, the 'g.', 'c.' and 'p.' are optional in the input. For example, `12:109702119insACC` is equally acceptable as `chr12:g.109702119_109702120insACC`. TransVar also accepts '>' in denoting MNV. E.g., `c.113G>TACTAGC` can be used in place of `c.113delGinsTACTAGC`. This is common in some database such as COSMIC.

11.10 When I annotate a variant for protein identifier, why would I end up getting results in another variant type?

TransVar follows in full the HGVS nomenclature while annotating protein level mutation identifiers. For example, a out-of-phase, in frame insertion, `ACIN1:c.1930_1931insATTCAC` will be annotated with `p.S643_R644insHS` rather than `R644delinsHSR`. Protein level mutation will be generated as if no nucleotide mutation information exists.

CHAPTER 12

Features

- supports HGVS nomenclature
- supports input from gene name, transcript ID, protein ID, UniProt ID and other aliases
- supports both left-alignment and right-alignment convention in reporting indels and duplications
- supports annotation of a region based on a transcript-dependent characterization
- supports mutations at both coding region and intronic/UTR regions
- supports noncoding RNA annotation
- supports VCF inputs
- supports long haplotype decomposition
- supports single nucleotide variation (SNV), insertions and deletions (indels) and block substitutions
- supports transcript annotation from commonly-used databases such as Ensembl, NCBI RefSeq and GENCODE etc.
- supports GRCh36, 37, 38 (human), GRCm38 (mouse), NCBIM37 (mouse)
- supports >60 other genomes available from Ensembl
- functionality of forward annotation.

CHAPTER 13

Citation

Zhou et al. Nature Methods 12, 1002-1003 (2015). <http://www.nature.com/nmeth/journal/v12/n11/full/nmeth.3622.html>

CHAPTER 14

License

CHAPTER 15

Need Help

If you have trouble using TransVar, please email zhouwanding@gmail.com. Bug reports are also welcomed at the issue tracker.

If you use TransVar in your work please cite Zhou et al. Nature Methods 12, 1002-1003 (2015). Thank you.

CHAPTER 16

Indices and tables

- genindex
- modindex
- search