SNPdat Manual.

SNPdat: Easy and rapid annotation of results from *de novo* SNP discovery projects for model and non-model organisms.

SNPdat was written by Anthony Doran under the supervision of Christopher Creevey.

SNPdat:

- Summary

SNPdat (SNP Data Analysis Tool) is a high throughput analysis tool that can provide a comprehensive annotation of both novel and known single nucleotide polymorphisms (SNPs). It is specifically designed for use with organisms which are either not supported by other tools or have a small number of annotated SNPs available, however it can also be used to analyse datasets from organisms which are densely sampled for SNPs. It can be used for analysis of any organism with a draft sequence and annotation. SNPdat makes possible analyses involving non-model organisms that are not supported by the vast majority of SNP annotation tools currently available.

- Availability

SNPdat is freely available on the web at http://code.google.com/p/snpdat/ and runs on all systems that support recent versions of Perl, including Linux, Mac OS X and Windows. SNPdat along with the manual, sample dataset, supplementary scripts and a short tutorial can be retrieved from the downloads section of this webpage (http://code.google.com/p/snpdat/downloads).

- Disclaimer

How to use SNPdat

- Quickstart

Running SNPdat with no arguments will also print a short help menu to the screen with some basic functions for running SNPdat. To run SNPdat you will need to call perl and specify the SNPdat script (see below).

e.g.

```
user@server $HOME/directory/with_all/SNPdat/files/
$ perl SNPdat

Error no arguments supplied.


SNPdat is a high throughput analysis tool that can provide a comprehensive
annotation of both novel and known single nucleotide polymorphisms (SNPs).

SNPdat requires that each file is specified when running the program. There
are 3 mandatory file definitions.

Usage:
perl SNPdat -i Input_file -f Fasta_file -g Gene_Transfer_File

Flags:

-i      Input file (Mandatory)
-g      Gene transfer file (GTF) (Mandatory)
-f      FASTA formatted sequence file (Mandatory)
-d      a dbSNP ASN_FLAT file processed using SNPdat_parse_dbsnp.pl
(optional)
-s      a file containing a summary of the queried SNPs (optional)
-o      output_file specified by the user (optional)
        NOTE:If no output file is specified, results will be printed to
'Input_file.output'


For more instructions see the SNPdat webpage:
http://code.google.com/p/snpdat/
```

(For those of you that have your files ready to run, see section below '[Running SNPdat]')

Or you can run SNPdat with the -h or --help option to see a help page of the version of SNPdat that you are running. To see the version of SNPdat that you are using, run SNPdat with the -v or --version option.

Some Prerequisites

1. Perl

SNPdat is not operating system (OS) dependent but does require Perl to run. Perl will be included in almost all installations of MacOS, Unix and Unix-like OS. Perl is not included in the default installation of Windows but is freely available. Strawberry Perl (http://strawberryperl.com/) is a perl environment for MS Windows containing all you need to run perl applications. It is designed to be as close as possible to a perl environment on UNIX systems. Alternatively, Cygwin (http://www.cygwin.com) can be downloaded and installed for free. Cygwin is a collection of tools which provide a Linux look and feel environment for Windows.

Also, Perl for MacOS and Unix machines in available from http://www.perl.org

Mandatory Input files:

SNPdat requires three mandatory input files:

---

1. User input file

- **SNPdat** can take two file formats as user input.

- **The first user input file format** accepted by SNPdat is a simple tab-delimited text file. This file should not have any header information. Any line beginning with # will be skipped. This file should contain three columns:

```
chromosome_id            position         mutation
```

e.g. tab-delimited text file

```
chr25       1234568        A
chrX        1234568        T
chr19       1234568        G
chr1        1234568        C
```

- **The second user input file format** accepted is a Variant Calling Format (VCF) file. This file should be tab-delimited. This file must have a header line that begins with **##fileformat=vcf**. Any line after the first that begins with # will be skipped. This file should have as its first five columns:

```
Chromosome_id    position         snp_id          ref_base         mutation
```

Any column after the first five will be ignored by SNPdat

e.g. VCF file

```
##fileformat=vcfv4.0
##_This line will be ignored by SNPdat_
#_This line will also be ignored_
##_ This line will be ignored_
chr25       1234568 SNPid1  T       A
chrX        1234568 SNPid2  C       T
chr19       1234568 SNPid3  C       G
chr19       1234568 SNPid4  C       A
chr1        1234568 SNPid5  G       C
```

2. Gene annotation file (GTF)

- The GTF (General Transfer Format) is identical to GFF version 2. GTF files must have 9 fields and must be tab-separated. The first eight GTF fields are the same as GFF. The ninth field (called group field in GFF) has been expanded into a list of attributes. Each attribute consists of a type/value pair. Attributes must end in a semi-colon, and be separated from any following attribute by exactly one space.

The attribute list must begin with the two mandatory attributes:

- gene_id value - A globally unique identifier for the genomic source of the sequence.
- transcript_id value - A globally unique identifier for the predicted transcript.

A brief description of the GTF fields

1. seqname - name of the chromosome or scaffold; chromosome names cane be given with or with out the 'chr' prefix
2. source - name of the program that generated this feature, or the data source (database or project name)
3. feature - feature type name, *e.g.* Gene, Variation, Similarity
4. start - start position of the feature, with sequence numbering starting at 1
5. end - end position of the feature, with sequence numbering starting at 1
6. score - a floating point value
7. strand - defined as + (forward) or - (reverse)
8. frame - One of '0', '1', '2'. '0' indicates that the first base of the feature is the first base of a codon, '1' that the second base is the first base of a codon, and so on.
9. attribute - a semicolon-separated list of tag-value pairs, providing additional information about each feature

For additional information on GTF specification see: http://genome.ucsc.edu/FAQ/FAQformat#format4

e.g. GTF file

```
chr20   protein_coding  exon    371915  372086  .       -
  .       gene_id "ENSBTAG00000024801"; transcript_id
"ENSBTAT00000034533"; exon_number "1"; gene_name "RANBP17";
transcript_name "RANBP17";
chr20   protein_coding  CDS     371915  372086  .       -
  0       gene_id "ENSBTAG00000024801"; transcript_id
"ENSBTAT00000034533"; exon_number "1"; gene_name "RANBP17";
transcript_name "RANBP17"; protein_id "ENSBTAP00000034423";
chr20   protein_coding  start_codon     372083  372086  .
  -       0       gene_id "ENSBTAG00000024801"; transcript_id
"ENSBTAT00000034533"; exon_number "1"; gene_name "RANBP17";
transcript_name "RANBP17";
chr20   protein_coding  exon    368319  368410  .       -
  .       gene_id "ENSBTAG00000024801"; transcript_id
"ENSBTAT00000034533"; exon_number "2"; gene_name "RANBP17";
transcript_name "RANBP17";
```

3. FASTA sequence file

Fasta format is a text-based format for representing nucleotide sequences, in which nucleotides are represented using a single-letter code (A,T,C,G). A sequence in FASTA format begins with a single line description. The description line is distinguished from the sequence data by a greater than (">") symbol at the beginning of the line. The word following the ">" symbol is the identifier of the sequence. There should be no space between the ">" and the first letter of the identifier.

e.g. fasta file

```
>chr20
CCCACGTGAAACATCTGGAGGAGCACCTGGACACGGCCAGGAAGGACCTCATCAAGTCCA
AGGACATAAACAGAAAACTTGAGCGGGACGTCCGCGAAGTGAGTGACTGCAGCTGTGTCT
GTTGTCTTGAGGTGGAATGTGGGTTTCTTGTTCTCCCGGGGATCTCAGCCTTGGGATGGC
TGCGTGAGTAAGAGCAGAGCACTGGAGAGACCACAACTGGCTGCCTCCCTGCCTCCGCAT
CAAGGTCTCTGGGGTAGAAGAGGCCTGGTCCAGGCGGTCCGTTGGCAGTGGACCAACATG
AGGGCAGCCCTGACGCTGTGCCGCACCCTGGGTGTGGGCTCCTCATTTCCACAAGTTCTA
```

- Some notes on the input files

- Note that the identifier format for all input files is the same. Even if the chromosome name differs, the format must remain the same. *i.e.* They all have chr preceding the chromosome name (Similarly none of the identifiers have this).

- A list of organisms with FASTA and GTF files is available from ensembl http://www.ensembl.org/info/data/ftp/index.html). A supplementary script is available to help users retrieve FASTA and GTF files. This is called **GTF_FASTA_finder.pl** and is included on the downloads page as part of the main SNPdat package (http://code.google.com/p/snpdat/downloads/list).

- Additional script for retrieving FASTA and GTF files

- **GTF_FASTA_finder.pl** is an additional script provided with SNPdat. This is written in Perl but uses the system call cURL to retrieve the information from Ensembl. cURL is a part of most Linux distributions and Mac OS X and can also be provided for windows through cygwin, which is a collection of tools that provide a Linux-like environment for windows. This script requires an internet connection.

- This is an interactive script designed to retrieve FASTA (dna) and GTF files from Ensembl. You will need to be connected to the internet to use this script. To run this script simply type **'perl GTF_FASTA_finder.pl'** into your terminal and follow the prompts. The GTF and FASTA files for that organism will be downloaded to the directory from which the script is run. Alternatively, GTF and FASTA information can be retrieved manually via the Ensembl website.
- SNPdat also works with genomic annotations from sources other than Ensembl as long as they are provided in GTF format. This includes the results of computationally derived annotations of *de novo* genomic assemblies, or transcriptomes.

e.g.

```
user@server $HOME/directory/with_all/SNPdat/files/
$ perl GTF_FASTA_finder.pl
|36| /pub/release-56    |37| /pub/release-57    |38| /pub/release-
58
|39| /pub/release-59    |40| /pub/release-60    |41| /pub/release-
61
|42| /pub/release-62    |43| /pub/release-63    |44| /pub/release-
64
|45| /pub/release-65    |46| /pub/release-66    |47| /pub/release-
67

Please select a release to choose from by typing its number:
47
You chose release-67

|0| ailuropoda_melanoleuca
|1| ancestral_alleles
|2| anolis_carolinensis
|3| bos_taurus
.
.
.
|56| tursiops_truncatus
|57| vicugna_pacos
|58| xenopus_tropicalis
```

```
Please select an organism (enter the corresponding number) to
retrieve the FASTA file for:
46

You chose organism Saccharomyces_cerevisiae
Now retrieving the relevant FASTA file for this organism
#################################################################
###### 100.0%

|0| ailuropoda_melanoleuca
|1| anolis_carolinensis
|2| bos_taurus
.
.
.
|55| tursiops_truncatus
|56| vicugna_pacos
|57| xenopus_tropicalis

Please select an organism (enter the corresponding number) to
retrieve the GTF file for:
45

Now retrieving the selected GTF file
Retrieving GTF: Saccharomyces_cerevisiae.EF4.67.gtf.gz
#################################################################
###### 100.0%
FASTA information has been retrieved for saccharomyces_cerevisiae
from release release-67 of ensembl
GTF information has been retrieved for saccharomyces_cerevisiae
from release release-67 of ensembl

.gtf.gz and .fa.gz files need be unzipped using the command 'gzip
-d filename'

If you have any queries regarding SNPdat or the additional scripts
please consult the website:
http://code.google.com/p/snpdat/
```

For more information see the SNPdat tutorial

Optional Input File:

- ## A processed FLAT file

SNPdat can also cross reference queries against information from external databases. One such database is dbSNP (http://www.ncbi.nlm.nih.gov/snp/). **SNPdat_parse_dbsnp.pl** can be used to process a FLAT file (found in the ASN1_FLAT directory for an organism) from dbSNP and output a file in the required format for SNPdat. These files are typically '.flat.gz' and so will have to be unzipped using the 'gzip -d filename' command. You should then be left with a file that has the extension '.flat'. This file is then used as an input file for '**SNPdat_parse_dbsnp.pl**'. A list of organisms for which dbSNP FLAT files exist can be found on the

website ftp://ftp.ncbi.nih.gov/snp/organisms/. Sections 'Additional script for retrieving FLAT files from dbSNP' and 'Preparing the optional input file' contain examples for retrieving and parsing information from dbSNP.

- Additional script for retrieving FLAT files from dbSNP

'**dbSNP_finder.pl**' is supplied with the SNPdat package. This script will get dbSNP flat files for organisms contained in ftp://ftp.ncbi.nih.gov/snp/organisms/. These files may also be retrieved manually by the user from the website.

e.g.

```
user@server $HOME/directory/with_all/SNPdat/files/
$ perl dbSNP_finder.pl

|0| Alectoris_9077/
|1| Bos_29061/
|2| almond_3755/
.
.
.
|109| zebrafish_7955/
|110| zebu_9915/
|111| zostera_29655/

Please select and organism by typing its corresponding number
6
You have chosen: arabidopsis_3702/

Retrieving file: 'ds_flat_ch1.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_ch2.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_ch3.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_ch4.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_ch5.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_chMasked.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_chMulti.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_chNotOn.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_chPltd.flat.gz'
################################################################## 100.0%
Retrieving file: 'ds_flat_chUn.flat.gz'
################################################################## 100.0%

Now you will need to unzip these files.
This can be done using gzip -d
e.g.
gzip -d ds_*.gz

If you wish to use these (unzipped) files for SNPdat please run
'SNPdat_parse_dbsnp.pl' to process them first
You may want to join these files before using SNPdat_parse_dbsnp.pl
To join these files you can use the 'cat' command
e.g.
cat ds*.flat > ds_arabidopsis_3702.all.flat
```

- **The processed FLAT file**

This is the output file from **SNPdat_parse_dbsnp.pl** **SNPdat_parse_dbsnp.pl** can be used to process a FLAT file from dbSNP and output a file in the required format for SNPdat.

- Preparing the optional input file

An additional script called **SNPdat_parse_dbsnp.pl** can be used to convert a dbSNP FLAT file into a format that can be used by **SNPdat**.

This can be used as follows

```
user@server $HOME/directory/with_all/SNPdat/files/

$ perl SNPdat_parse_dbsnp.pl the_input_file.FLAT processed_output.txt
```

You will then be prompted to type the name of the assembly you want to map to:

```
This is a perl script to process a dbSNP FLAT file of SNP data and produce
an output file that can be supplied as input for the software SNPdat

Please enter the assembly you want to map RS ids to
:
```

Once you have enter this `[and hit return]` the script will continue

e.g.

```
This is a perl script to process a dbSNP FLAT file of SNP data and produce
an output file that can be supplied as input for the software SNPdat

Please enter the assembly you want to map RS ids to
:UMD3.1

The resulting output file is processed_output.txt
This file can be used as input for SNPdat using the -d switch
```

Now that the optional input file is ready you can run SNPdat with all input files

Output file:

Specifying an output file is not necessary but can be done.

- if no output file is specified the results will be printed to an output file with the same name as the input file but with the suffix **.output**
- for more information on specifying the output file see section 'Running SNPdat'

- Output file format

- SNPdat will produce a tab-delimited file as output. This file can have upto 25 columns of results.

Column Number	Description
1	The queried SNPs chromosome ID
2	The queried SNPs genomic location
3	Whether or not the queried SNP was within a feature
4	Region containing the SNP; either exonic, intronic or intergenic
5	Distance to nearest feature
6	Either the closest feature to the SNP or the feature containing the SNP
7	The number of different features that the SNP is annotated to
8	The number of annotations of the current feature
9	Start of feature (bp)
10	End of feature (bp)
11	The gene ID for the current feature
12	The gene name for the current feature
13	The transcript ID for the current feature
14	The transcript name for the current feature
15	The exon that contains the current feature and the total number of annotated

	exons for the gene containing the feature
16	The strand sense of the feature
17	The annotated reading frame (when contained in the GTF)
18	The reading frame estimated by SNPdat
19	The estimated number of stop codons in the estimated reading frame
20	The codon containing the SNP, position in the codon and reference base and mutation
21	The amino acid for the reference codon and new amino acid with the mutation in place
22	Whether or not the mutation is synonymous
23	The protein ID for the current feature
24	The RS identifier for queries that map to known SNPs
25	Error messages, warnings etc

Running SNPdat

SNPdat requires that each file is specified when running the program. There are five file definitions.

1. `-i input_file`
2. `-g GTF_file`
3. `-f FASTA_file`
4. `-d dbSNP_processed_file`
5. `-o output_file`

e.g.

```
perl SNPdat.pl -i input_file -g GTF_file -f FASTA_file -d
dbSNP_processed_file
```

Specifying an output file is optional. In the above code no output file is specified and so the results will be printed to `input_file.output`. Below is an example with the output file specified.

```
perl SNPdat.pl -i input_file -g GTF_file -f FASTA_file -d
dbSNP_processed_file -o output_file
```

Results from the above will be printed to `output_file`.

As SNPdat is running you should see progress reports as follows:

```
start time:
DAY MON  1 HOUR:MIN:SEC IST YEAR
DAY MON  1 HOUR:MIN:SEC IST YEAR
DAY MON  1 HOUR:MIN:SEC IST YEAR
GTF parsed:
DAY MON  1 HOUR:MIN:SEC IST YEAR
User input parsed:
DAY MON  1 HOUR:MIN:SEC IST YEAR
Finished analysing all SNPs with sequence information
:DAY MON  1 HOUR:MIN:SEC IST YEAR
Finished analysing all SNPs with no sequence information
:DAY MON  1 HOUR:MIN:SEC IST YEAR
SNPdat finished analysing all SNPs:
DAY MON  1 HOUR:MIN:SEC IST YEAR
view file 'output_file' for results
```

All together you should see something like this on your screen:

```
perl SNPdat.pl -i input_file -g GTF_file -f FASTA_file -d
dbSNP_processed_file -o output_file
start time:
DAY MON  1 HOUR:MIN:SEC IST YEAR
DAY MON  1 HOUR:MIN:SEC IST YEAR
DAY MON  1 HOUR:MIN:SEC IST YEAR
GTF parsed:
DAY MON  1 HOUR:MIN:SEC IST YEAR
User input parsed:
DAY MON  1 HOUR:MIN:SEC IST YEAR
Finished analysing all SNPs with sequence information
:DAY MON  1 HOUR:MIN:SEC IST YEAR
Finished analysing all SNPs with no sequence information
:DAY MON  1 HOUR:MIN:SEC IST YEAR
SNPdat finished analysing all SNPs:
DAY MON  1 HOUR:MIN:SEC IST YEAR
view file 'output_file' for results
```

Advanced options

- x ‘codons that cross feature boundaries’

Warning! This is recommended for advanced users who fully understand what the software is doing and how it will behave.

Sometimes a SNP can occur at the start or end position of a feature, or sufficiently close that the codon that it occurs in may cross the boundary of that feature. If this happens, SNPdat should return ‘?’ ‘s in the position(s) that are outside of the feature (usually intergenic or intronic). You may however, wish to retrieve information from the next/previous feature (**for the same transcript**) and use the sequence information instead of ? ‘s.

As the same applies to both ends of the feature, from this point on, I’ll only be talking about SNPs that are at the end of a feature and sequence information from the next feature.

Below is an example of a SNP that is at a feature boundary and crosses into the next feature.

Example 1
Between the brackets is the intron [GT----//----AG]. The nucleotide where the SNP occurs in 22 (T) and the mutation is to an A.

```
ATG AAT TGC TTG ATA GCT CTT T [GT----//----AG] TT TCT TGT GGG

 M   N   C   L   I   A   L                     F   S   C   G



Mutation : T -> A

ATG AAT TGC TTG ATA GCT CTT A [GT----//----AG] TT TCT TGT GGG

 M   N   C   L   I   A   L                     I   S   C   G
```

In this case the mutation occurs in the first codon position of a mutation that crosses the feature boundary. By default SNPdat (SNPdat_v1.0.5 and above) will return [T/A]??. However you may wish to use the information from the next feature (**for the same transcript**). Using the –x option will enable this.

If you use the –x option, you must specify the features that you wish to do this for using a comma separated list. Typically, you will only want to use this for the CDS feature. Not all exons are translated so using this option with features other than CDS may retrieve information that doesn’t make sense.

You can specify the features like so
-x feature1,feature2,feature3

Features that you specify must be the same case as the features that are in the GTF. In the above example, assuming that it is a CDS feature, you would specify

-x CDS
And the returned codon would be [A/T]TT or I (Isoleucine).

Note
If this feature was a CDS and you used

-x exon
You would get [A/T]??. This will only retrieve information across boundaries for the features you specify. All other features will be dealt with in the default way.

Here are some examples of the option in use:

1.
SNPdat -i INPUT_FILE -g GTF_FILE -o OUTPUT_FILE -x CDS,exon

2.
SNPdat -i INPUT_FILE -g GTF_FILE -o OUTPUT_FILE -x exon

3.
SNPdat -i INPUT_FILE -g GTF_FILE -o OUTPUT_FILE -x CDS

4.
SNPdat -i INPUT_FILE -g GTF_FILE -o OUTPUT_FILE

1. Retrieval of information will be done for both CDS and exon features should a codon cross the feature boundary
2. Retrieval of information will be done for exon features should a codon cross the feature boundary
3. Retrieval of information will be done for CDS features should a codon cross the feature boundary
4. No retrieval of information across feature boundaries will be done for any feature

This option will only retrieve information for the same feature type belonging to the same transcript. If your SNP occurs in the first or last position of a transcript, retrieval of information cannot be done so default behaviour will be used.

You can specify as many features as you want as long as they are comma separated. Use the –x option only once when specify multiple features.

Further instructions

Further instructions for using SNPdat and any of the additional scripts can be found in the short tutorial on the downloads page and on the website. Sample data to run SNPdat can also be found on the downloads page. http://code.google.com/p/snpdat/downloads/list

Authors and Contact

Anthony G Doran[1,2], Christopher J Creevey[1]

[1]Teagasc Animal and Bioscience Research Department, Animal & Grassland Research and Innovation Centre, Teagasc, Grange, Dunsany, Co. Meath, Ireland.

[2]Molecular Evolution and Bioinformatics Unit, Biology Department, NUI Maynooth, Maynooth, Co. Kildare, Ireland.

Corresponding authors:

chris.creevey@teagasc.ie and anthony.doran@teagasc.ie

Queries/bugs/suggestions can be directed to either of the above email addresses.

How to cite SNPdat

The published manuscript for SNPdat is available from BMC Bioinformatics (http://www.biomedcentral.com/1471-2105/14/45#).

SNPdat can be cited as:

Doran AG, Creevey CJ: **Snpdat: Easy and rapid annotation of results from de novo snp discovery projects for model and non-model organisms.** BMC Bioinformatics 2013, **14**(1):45.