Getting started

Dependencies

Python (https://www.python.org/)

- version 2.7.15
- required non-standard modules: scipy, statsmodels

HISAT2 (https://ccb.jhu.edu/software/hisat2/manual.shtml)

- version 2.1.0
- HISAT2 directory must be added to the user path environment variable
- HISAT2_INDEXES environment variable must be specified
- prebuilt GRCh38 index (ex: grch38_snp_tran)

Cutadapt (https://cutadapt.readthedocs.io/en/stable/guide.html)

- version 1.18

Creating IAS sample input file

The sample input file contains the information for each sample to be analyzed. The file is a tab-delimited text file that can be initially generated as a spreadsheet. The file should not include a header and has the following structure:

column 1 =	sample_ID
column 2 =	FASTQ filename containing the IRL-tagged sequences
column 3 =	FASTQ filename containing the paired adaptor reads for the IRL-tagged sequences
column 4 =	FASTQ filename containing the IRR-tagged sequences
column 5 =	FASTQ filename containing the paired adaptor reads for the IRR-tagged sequences
column 6 =	HISAT2 index to be used for mapping (*e.g.* grch38_snp_tran)

The sample input file should be saved in the same directory as the FASTQ files to be analyzed.

Running IAS

Once all software packages are installed, the IAS can be tested using the demonstration data set provided with IAS:

Trimming and mapping of raw sequence data

The IAS_mapper.py tool uses cutadapt to remove the transposon and adaptor sequences. Trimmed sequences are then mapped using HISAT2. A SAM parser then evaluates each read and generates a .uniq output file (see "File Specifications") containing an entry for every insertion site identified in the sample.

Step 1

- generate the sample input file (see section "Creating IAS sample input file")
- place sample input file in the directory containing the raw sequence files

Step 2

- execute the following command from within the directory containing the sequence files:

```
$ python [path_to_IAS]/IAS_mapper.py       sample_info.txt
```

- A .uniq file will be generated for every sample

Data filtering to remove low quality reads

The UNIQtoGFF3.py tool is used to perform quality filtering of the initial .uniq files generated by the IAS_mapper.py tool. The UNIQtoGFF3.py can filter raw data using three different approaches, each based on the idea that the abundance of any particular transposon insertion is proportional to the number of reads that are mapped to that particular insertion site. For example, a transposon insertion present in every cell of the sample (*i.e.* clonally expanded) should generate the largest number of mapped reads. Furthermore, it is expected that both sides of the transposon insertion (IRL and IRR) will generate high read numbers. Based on this logic, transposon insertion events that generate reads in both the IRL and the IRR libraries are more likely to be clonally expanded than those that are detected in only a single library. Therefore the UNIQtoGFF3.py tool allows the user to specify different filter settings for insertion events that are detected in both libraries versus a single library. Finally, the user can also specify a minimum number of reads required for any insertion event to be included in the downstream analysis.

two_library_filter:	specifies the minimum requirement for sites detected in both IRL and IRR libraries. The value reflects the normalized abundance of the site within each library. For example, the insertion site with the highest read number in each library will have a value of 100. [value range 0 – 100]
one_library_filter:	specifies the minimum requirement for sites detected in either IRL or IRR library. [value range 0 – 100]
read_#_filter:	specifies the minimum number of reads required for any insertion site to be included in downstream analysis. Insertion sites will be removed based on this filter even if they have passed the other filters.

- execute the following command from within the directory containing the .uniq files:

```
$ python [path_to_IAS]/UNIQtoGFF3.py two_library_filter  one_library_filter  read_#_filter
```

The UNIQtoGFF3.py tool creates a new directory called "GFF3" that contains a .gff3 file corresponding to each .uniq file that met the filter criteria specified by the user. The filter settings are saved in the same directory. The GFF3 file format is a variation of the standard GFF3 file format (see "File Specifications").

Identification of candidate genes using gene-centric common insertion site analysis

We previously established a gene-centric common insertion site analysis method to identify genes that are recurrently impacted by Sleeping Beauty transposon insertion (Brett et al. 2011). The IAS includes a new version of the gCIS tool (gCIS2.py) that is specifically optimized for the analysis of data from cell-based screens (Feddersen et al. 2019)(see "IAS pipeline description" document for details).

The gCIS2.py tool requires three inputs. The first is a single GFF3 file containing all samples within the same treatment group (*i.e.* phenotypically selected vs. untreated). The second input specifies the promoter region to be included in the analysis. The allowed promoter sizes are: 0, 5000, 10000, 15000, 20000, 25000, 30000, 35000, and 40000 bp. The third input specifies the gene annotation set to be used for the analysis (ensembl, ucsc, refseq).

- execute the following command from within the directory containing the GFF3 file to be analyzed:

```
$ python [path_to_IAS]/gCIS2.py      file.gff3     promoter      gene_annotation
```

File Specifications

UNIQ file format

The IAS_mapper.py tool creates output in a custom tab-delimited file format containing 10 data columns:

1. sample ID
2. name of the chromosome or scaffold
3. position of "T" of the genomic "TA" insertion site
4. IRL read number
5. IRR read number
6. strand - defined as + (forward) or - (reverse)
7. number of IRL amplicons mapped to this site
8. number of IRR amplicons mapped to this site
9. normalized IRL abundance (*i.e.* % of maximum)
10. normalized IRR abundance (*i.e.* % of maximum)

GFF3 file format

The GFF3 file generated by the UNIQtoGFF3.py tool includes the nine standard columns as specified (https://useast.ensembl.org/info/website/upload/gff3.html). An additional four columns are included as follows:

1. seqid - name of the chromosome or scaffold
2. source
3. type - type of feature
4. start - start position of the feature
5. end - end position of the feature
6. score – maximum normalized abundance
7. strand - defined as + (forward) or - (reverse).
8. phase – [not used]
9. attributes – unique ID
10. IRL read number
11. IRR read number
12. normalized IRL abundance (*i.e.* % of maximum)
13. normalized IRR abundance (*i.e.* % of maximum)

References

Brett BT, Berquam-Vrieze KE, Nannapaneni K, Huang J, Scheetz TE, Dupuy AJ. 2011. Novel molecular and computational methods improve the accuracy of insertion site analysis in Sleeping Beauty-induced tumors. *PLoS One* **6**: e24668.

Feddersen CR, Schillo JL, Varzavand A, Vaughn HR, Wadsworth LS, Voigt AP, Zhu EY, Jennings BM, Mullen SA, Bobera J et al. 2019. Src-dependent DBL family members drive resistance to vemurafenib in human melanoma. *bioRxiv* doi:https://doi.org/10.1101/561597.