NGSCheckMate

Version 1.0

Eunjung Lee[1,2,*,†], Sejoon Lee[3,*], Soohyun Lee[1,*], Semin Lee[1,5], Woong-Yang Park[3], Peter J. Park[1,2,4,†]

[1]Department of Biomedical Informatics, Harvard Medical School, Boston, Massachusetts 02115, USA.

[2]Department of Medicine, Division Genetics, Brigham and Women's Hospital, Boston, MA 02115, USA

[3]Samsung Genome Institute, Samsung Medical Center, Seoul, South Korea

[4]Ludwig Center at Harvard, Boston, MA 02115, USA

[5]Department of Biomedical Engineering, School of Life Sciences, Ulsan National Institute of Science and Technologies, Korea (current affiliation).

* These authors contributed equally.
[†] Correspondence should be addressed to E.L. (elee20@partners.org), P.J.P. (peter_park@hms.harvard.edu)

1. Introduction

NGSCheckMate is a software package for identifying next generation sequencing (NGS) data files from the same individual. It analyzes various types of NGS data files including (but not limited to) whole genome/exome sequencing, RNA-seq, ChIP-seq, targeted sequencing of various sequencing depths. Data types can be mixed (e.g. RNA-seq and ChIP-seq), though not every combination of types has been tested (see Lee et al. (in submission) for more details).. It takes BAM (reads aligned to the genome), VCF (variants) or FASTQ (unaligned reads) files as input. NGSCheckMate uses depth-dependent correlation models of allele fractions of known single-nucleotide polymorphisms (SNPs) to identify samples likely originating from the same individual. Our alignment-free module is fast (e.g., less than one minute for RNA-seq using a single core) and we recommend it for a quick initial quality check, before pooling / aligning the sequenced reads. The bam and vcf modules can be used after the alignment and variant calling steps, respectively, to ensure the correct sample annotation before further downstream analysis. Currently, it works only for human data.

2. Requirement

1. Software environment

i. If your input files are .bam or .vcf files
   1. Unix/Linux System
   2. Python 2.6 or above
   3. Samtools version 0.1.19

4. Bcftools version 0.1.19
5. R 3.1 or above (If you want to get a PDF version diagram)

ii. If your input files are .fastq files
1. Unix/Linux System
2. R 3.1 or above (If user want to get a PDF version diagram)

2. Reference file and other files

i. If your input is BAM / VCF
1. Hg19 / GRCh37 Reference FASTQ file (required for bam input)
2. A bed file (.bed) that lists the locations of selected SNPs (included in the package)

ii. If your input is FASTQ
1. A binary pattern file (.pt) that lists the sequences spanning selected SNPs (included in the package).

3. Installation

1. Download NGSCheckMate

```
cd installation_directory
git clone https://github.com/parklab/NGSCheckMate.git
```

2. Set paths in the configuration file (required only for BAM/VCF input)

If your input is bam/vcf, add the following lines to your ncm.conf file in the package directory. If your input is fastq, you can skip this step.

```
REF=”absolute path”    # path for the reference fastq file
SAMTOOLS=”absolute path” # path for SAMTOOLS
BCFTOOLS=”absolute path” # path for BCFTOOLS
```

3. Compilation of C codes

All the binaries are provided in the package, so in general there is no need to compile the source codes (you can skip this section). However, in case you need to, you can use make to create the necessary binaries.

ngscheckmate_fastq : the main C program used for the fastq input

```
cd ngscheckmate_fastq-0.0.9
make
```

patterngenerator (in case you want to create your own pattern file for the fastq module)

```
cd patterngenerator
```

make

4. Usage

NGSCheckMate is able to test three types of files: BAM, VCF, FASTQ. Therefore, NGSCheckMate provides three modes to test sample identity. Each mode is executed by a Python script.

1. BAM/VCF mode

Usage: python ncm.py <-B | -V> <–d INPUT_DIR | -I INPUT_LIST_FILE> -bed BED_FILE –O OUTPUT_DIR [options]

Required arguments

-B | -V	A flag that indicates the input file type (B: bam, V: vcf)
-d DIR	The directory that contains the input files (-l can be used instead)
-I FILE	A text file with paths to the input files (absolute path, one per line) (-d can be used instead)
-bed FILE	SNP bed file that lists the locations of selected SNPs (included in the package) (e.g. Distinct_features_21067_hg19.bed, if your bam file is based on hg19, Distinct_features_21067.bed, if based on GRCh37)
-O DIR	The name of the output directory

Optional arguments

-N PREFIX	The prefix of the output file (default: “output”)
-f	Use strict correlation threshold. Recommended if your data contains family members.
-t FILE	A file with test sample files

2. FASTQ mode

Usage: python ncm_fastq.py -I INPUT_LIST_FILE -pt PT_FILE –O OUTPUT_DIR [options]

Required arguments

-I FILE	A text file containing paths to the input files and sample names (see the Input file format section for more details)
-pt FILE	A binary SNP pattern file (.pt) that lists the sequences spanning selected SNPs (included in the package). (e.g. Distinct_features_21067.pt for human, regardless of your genome version)
-O DIR	The name of the output directory

Optional arguments

-N PREFIX	The name of the output file (default: “output”)
-f	Use strict correlation threshold. Recommended if your data contains family members.
-t FILE	A file with test sample files
-s FLOAT	Subsampling rate (default: 1.0)
-d INT	As an alternative to a user-defined subsampling rate, let the program compute the subsampling rate given a user-defined desired_depth and the data
-R INT	The reference length (default: 3E9) to be used for computing subsampling rate. If the data is NOT WGS from human, and if you aree using the -d option, it is highly recommended to specify the reference length. For instance, if your data is human RNA-seq, the total reference length could be about 3percent of the human genome, which can be set as 1E8

-L INT The length of the flanking sequences being used to identify SNV sites. Default is 21bp. It is recommended not to change this value, unless you have created your own pattern file with a different pattern length

-p INT Number of threads to use (default: 1)

3. FASTQ mode, alternative way

Alternatively, the C program, ngscheckmate_fastq, can be used directly to calculate VAF information. The output vaf files can be fed into another script, vaf_ncm.py to complete the downstream part. This may be useful for submitting parallel jobs. ngscheckmate_fastq takes one fastq file (or two for paired-end data) as input and streams the results to stdout.

ngscheckmate_fastq

Usage : ./ngscheckmate_fastq [options] -1 fastqfile1 [-2 fastqfile2] patternfile(.pt) > output.vaf

Input arguments (required)

patternfile : a binary file containing sequences flanking representative snv sites, along with markers indicating the snv index and whether the sequence represents reference or alternative allele.

-1, --fastq1 <fastq_file_1> : fastq file for SE data or first fastq file for a PE data. (required)

Options

-2, --fastq2 <fastq_file_2> : second fastq file for a PE data.

-s, --ss <subsampling_rate> : subsampling rate (default 1.0)

-d, --depth <desired_depth> : as an alternative to a user-defined subsampling rate, let the program compute the subsampling rate given a user-defined desired_depth and the data.

-R, --reference_length <reference_length> : The reference length (default : 3E9) to be used for computing subsampling rate. If the data is NOT WGS from human, and if you're using the -d option, it is highly recommended to specify the reference length. For instance, if your data is human RNA-seq, the total reference length could be about 3% of the human genome, which can be set as 1E8.

-L, --pattern_length <pattern_length> : The length of the flanking sequences being used to identify SNV sites. Default is 21bp. It is recommended not to change this value, unless you have created your own pattern file with a different pattern length.

-p, --maxthread <number_of_threads> : number of threads to use (default : 1)

-j, --nodeptherror : in case estimated subsampling rate is larger than 1, do not stop but reset it to 1 and continue.

vaf_ncm.py

Usage: python vaf_ncm.py -f -I <input_directory> -O <output_directory> -N output

-I : input directory that contains the output (vaf) files of ngscheckmate_fastq.

-O : output directory

-N : output_filename_tag

5. Input file format

1. BAM/VCF mode

The file that contains input bam/vcf file paths (-l) should be in the following format for example.

```
/data/LSJ.bam
/data/LSH.bam
/data/LSI.bam
```

2. FASTQ mode

The file that contains input bam/vcf file paths (-l) should be in the following format.

i. **Paired-end data**
Three columns are required for paired-end data:

R1 fastq file (tab) R2 fastq file (tab) sample name (\n)

Example:

```
/data/LSJ_R1.fastq    /data/LSJ_R2.fastq    LSJ
/data/LSH_R1.fastq    /data/LSH_R2.fastq    LSH
```

ii. **Single-end data**
Two columns are required for single-end data:

fastq file (tab) sample name (\n)

Example:

```
/data/LSJ.fastq    LSJ
/data/LSH.fastq    LSH
```

6. Examples

1. Test sample pairing using BAM input

```
python ncm.py -B -f -d /data/wgs_download/LUAD/ -O ./LUAD_WGS/ -N LUAD -
bed ./Distinct_features_21067.bed
```

2. Test sample pairing using VCF input

```
python ncm.py -V -f -d /data/wgs_download/LUAD/ -O ./LUAD_WGS/ -N LUAD -
bed ./Distinct_features_21067.bed
```

3. Test sample pairing using fastq input

```
python ncm_fastq.py -O ./output -l fastq_list.txt -N print -p 4 -pt ./Distinct_features_21067_fastq.pt
```

7. Output

1. PREFIX_all.txt

This output file lists both matched and unmatched sample pairs with VAF correlation coefficients and representative sequencing depths.

Format

```
Sample1 (tab) matched_or_unmatched (tab) Sample2 (tab) Correlation (tab) Depth
-----------------------------------------------------------------------------------------
6216-01A      matched      6216-10A        0.7957    4.9
6216-01A      unmatched    6324-10A        0.2153    15
----------------------------------------------------------------------------------------
```

2. PREFIX_matched.txt

This output file lists only matched sample.

Format

```
Sample1 (tab) matched_or_unmatched (tab) Sample2 (tab) Correlation (tab) Depth
-----------------------------------------------------------------------------------------
6216-01A      matched      6216-10A        0.7957    4.9
----------------------------------------------------------------------------------------
```

3. PREFIX.pdf

This pdf file shows a dendrogram image of hierarchical clustering of samples based on VAF correlation coefficients.

8. Supporting scripts

1. Patterngenerator

The set of scripts in the patterngenerator folder in the package generate the .pt file used by the fastq module, in case the user wants to generate a custom .pt file. It requires a bed file containing a set of SNP positions, a genome reference file (both fasta and bowtie index) and the bowtie alignment program (http://bowtie-bio.sourceforge.net/index.shtml).

```
usage: makesnvpattern.pl bedfile genomefasta genome(bowtie)index outdir
outprefix
```

2. Graph generator (Rscript)

This script with a set of xgmml templates is used for generating a graph representing matching files as connected nodes. The output format is in .xgmml, which can be read by Cytoscape.

```
source("graph/ngscheckmate2xgmml.R")
create.xgmml.from.ngscheckmateout(label.file,ngscheckmateoutput.file,output.x
gmml)
```

Label file : a tab-delimited text file containing a bam file name (1st column), an individual identifier (2nd column) and optionally, a file identifier (3rd column) for each line. An individual identifier must be unique to a subject (e.g. both tumor and normal samples from the same individual must have the same individual identifier). A file identifier must be unique to a file name.

ngscheckmateoutput.file : the output text file of NGSCheckMate. It is a tab-delimited text file containing two bam file names (1st and 2nd columns), VAF correlation (3rd column) and average depth (4th column). It may contain either all pairs or matched pairs, depending on the option used to run NGSCheckMate. Either type works.

Sample label file (sample.label.txt) and ngscheckmateouput.file (sample.input.txt) can be found in the subdirectory graph/.