FindPeaks 3.1.9.2 Manual

Anthony P. Fejes,
afejes@bcgsc.ca
Graduate Student
University of British Columbia

Research performed through:
BC Cancer Agency
Genome Sciences Centre,
Vancouver BC, Canada

Document last edited: May 9, 2008

Last Changed Rev: 38552
Last Changed Date: 2008-05-09 09:48:28 -0700 (Fri, 09 May 2008)

To cite this work, please contact the author for an up to date reference.

Please contact the author by email with suggestions, comments and code modifications, all of which are gratefully accepted.

INTRODUCTION:

FindPeaks was designed to identify areas of enrichment from massively parallel short-read sequencing data sets. It has since been expanded to include several new modules such that it can be useful on several new fields.

Areas of enrichment are typically identified by the height and the width of the peak observed, when sequenced fragments are mapped to the genome. However, simply looking at these two parameters fails to take into account many complexities involved. When two areas of enrichment are in close proximity, they will overlap and the relationship between the two peaks becomes more difficult to untangle.

FindPeaks does not using a sliding window algorithm. Instead, the collection of sequences for each chromosome is read and sorted by location, creating a list that is then traversed to identify regions with overlap. These regions are then independently processed, and checked against the minimum height criteria to be categorized as a peak. As each region is found, it is processed using the options that are specified on the command line, which may include the -directional, -trim and -subpeaks flags.

FONTS USED IN THIS DOCUMENT:

This document uses a common convention of marking executable commands and URLs using a fixed width monotype (FreeMono) font. Thus, commands that may be executed or used in a browser will `appear in this font`. All other text, explanations and guidance will be written in the FreeSerif font.

CURRENT SVN LOCATION (Developers only):

FindPeaks source code is currently available (only) to developers at the Genome Science Centre through the local SVN repository. The FindPeaks code is part of a larger code-base of Java programs using a common infrastructure. To obtain an SVN version of the code, check out the following location (password required.)

```
https://svn01.bcgsc.ca/svn/Illumina_Java/trunk
```

Tagged versions (from FindPeaks 2.0 to current) can be obtained from

```
https://svn01.bcgsc.ca/svn/Illumina_Java/tags
```

BUILDING FINDPEAKS (Developers only):

FindPeaks 3.1.x from SVN can be built using the ant utility (ant package required). On Debian/Ubuntu based systems, installing ant can be done with the command:

```
sudo apt-get install ant
```

Once installed, building FindPeaks can be performed by entering the root directory of the project (eg. For developers using the standard Eclipse work space structure: /home/name/workspace/Illumina_Java/) and issuing the command:

```
ant fpsuite
```

MEMORY USE:

FindPeaks version 3.1.9+ uses less than 1Gb of RAM for up to 17M sequence reads. Because memory requirements generally scale with the number of reads, your memory usage will vary.

WORK FLOW:

It is suggested that when using Eland files, they be pre-processed to remove non-unique alignents before using the FindPeaks application. This includes the removal of non-aligned reads, reads aligned to multiple locations and reads which fail QA tests. Thus, the following filtering step is suggested to assist in preparing your Eland files before use:

Retain only unique hits from an Eland file:

```
grep "U[012]" Input.eland > Input.um.eland
```

To begin the FindPeaks work flow, it is necessary to separate your reads into files, each of which contains a separate chromosome. To do this, the SeparateElandReads utility has been provided, which is able to process both .eland and .eland.gz files natively, and will produce *.part.eland.gz files, where * is the name of the individual chromosomes. An extra file, meta_info.txt, is also printed out which contains a summary of the number of reads found in each gzipped file and the total number of reads parsed.

For developers using the compiled source code:

```
cd /home/afejes/workspace/FindPeaks/classes

time java src/fileUtilities/SeparateElandReads
/path/to/files/Input.noDots.um.eland /output_dir/
```

For users of the pre-packaged jar files:

```
time java -jar SeparateElandReads.jar
/path/to/files/Input.noDots.um.eland /output_dir/
```

Once the files have been successfully separated, FindPeaks can be run on the data set.

For developers using the compiled source code:

```
time java src/projects/findPeaks/FindPeaks -name test -dist_type 1 200
-minimum 1 -eff_size 2.156E9 -output /output_dir/ -input
/path/to/files/*.part.eland.gz
```

For users of the pre-packaged jar files:

```
time java -jar FindPeaks.jar -name test -dist_type 1 200 -minimum 1 -eff_size
2.156E9 -output /output_dir/ -input /path/to/files/*.part.eland.gz
```

NOTES:

1) Chromosome naming in eland files: The naming of chromosomes varies between institutes, and is thus difficult to anticipate at run time. GSC behaviour typically involves providing the name of the chromosome as either an integer or char value (e.g. 1, 5,19, X, Y, etc), or as a NCBI name (e.g, starting with Homo_sapiens.NCBI36.42.dna.chromosome"). Where neither of these applies, Findpeaks will process the string provided in the eland file by trimming off a prefix of "chr" (if it exists), any information prior to and including a "\" character (if it exists) and any information including and following the final "." character (if it exists). This should remove any directory structure and file extension information.
2) When writing a wig file, the chromosome name will be pre-pended with a "chr" string to conform to UCSC formatting.
3) For large data sets, the standard amount of memory allocated to the java process may be insufficient. This can be managed by allocating more memory, when available, by using the "-Xmx" flag. This is done by issuing the flag directly after the java command and providing the amount of memory to be allocated. As an example, allocating 4Gb of ram would be done by writing "time java -Xmx4G FindPeaks.jar ...."

FIND PEAKS PARAMETERS:

-help

Prints an abbreviated list of available parameters

-aligner <char>

Determines which aligner input to use:

E: uses Eland input
B: Uses Bed format files. (Special version available from Anthony Fejes upon request only.)
X: uses Exonerate (Vulgar) input (Experimental – untested)
2: uses Eland Extended input mode (Experimental – expected in FindPeaks 3.2.x)

If flag is omitted: defaults to Eland mode

-directional

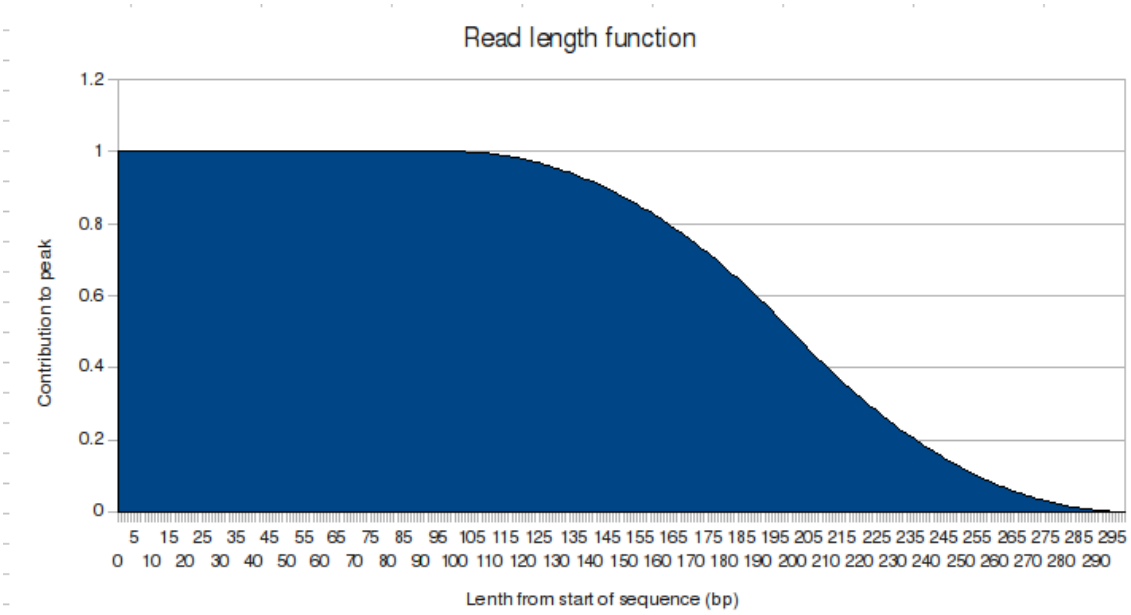
Engages directional mode, which considers directional reads for identifying the location of the maximum peak height. This may be useful for refining narrow peaks and filtering out noise.

If flag is omitted: directional mode is not engaged.

-dist_type <integer> [<integer>]

0: fixed width model: If used, it must be followed by an integer value representing the fixed with of the sequences used. All reads are then assumed to be that length. This mode is provided for backwards compatibility with FindPeaks 1 and FindPeaks 2.1.4 modes and is not suggested for production use.

1: triangle distribution: this assumes a triangle based distribution in which fragments have a minimum length of 100, a maximum length of 300, and a user supplied median size. If used, it must be followed by an Integer value representing the median value of the distribution. When used for creating the adaptive distribution (boot strap), a the median value defaults to 174. Graph below was generated with a median value of 200.



2: Adaptive (sampled) distribution: This mode will sample the reads on a full chromosome, and determine the median distance from the peak sites. From this distribution, it determines the sample fragment distribution. (It does not determine the distribution of the original fragment sizes.) (Currently disabled for non-GSC users while testing is ongoing.)

3: Native mode: This mode uses the actual length of the sequences themselves. This mode was provided

for generating wig files showing the sequence coverage across the genome of interest, however, it has undergone minimal testing, and thus the authors suggest that the ElandtoBed application, which provides very similar information, be used instead.

If flag is omitted: defaults to type 1, triangle distribution. This mode is suggested for most applications. Current recommended values are "-dist_type 1 200".

-eff_size <float>

This is the effective genome size used for the FindPeaks FDR modules, and used as the length of the chromosome generated when generating an R script for postscript preparation.

Current estimates done at the BC Genome Science Centre show that ~70% of human and mouse genomes may be mapped using ~32 base alignments, thus, recommended values for the human and mouse genomes are:

Organism	Calculation	Effective Genome Size	Database
Human	70% of 3.080 Gb	2.156e9	UCSC hg18
Mouse	70% of 2.655 Gb	1.8655e9	UCSC mm9

If flag is omitted: program will not run.

-filter

Turns on duplicate filtering. Filtering is currently only performed to remove reads in the same direction that share a start location.

If flag is omitted: duplicate filtering is off.

-hist_size <integer>

The number of cells in the output FDR histogram. The length of the histogram does not affect the running of the FindPeaks application, but only the maximum height for which data is shown in the final summary. Histogram always starts at one.

If flag is omitted: histogram size is set to 30.

-input <String> [<String> <String>...]

The set of eland files to read. A minimum of one file must be provided. A maximum of one file is used in R script mode Each file is treated as a separate chromosome. Wild card expressions are acceptable, if allowed by the Operating System in use. (e.g /path/*.part.eland.gz)

If flag is omitted: program will not run.

-mcfdr [<integer>]

Turns on the Monte Carlo simulator. This performs a user defined number of iterations, to provide a simple random MC model of expected peak heights.

Algorithm: Each iteration uses the same number of valid reads and the effective genome size of the processed data to recreate a non-enriched distribution from which a false discovery rate can be estimated. Random locations are generated for each read, which are then processed using the FindPeaks application and the same functions used to process the input files are used to count the number of peaks obtained. Note: the -filter flag is not used by the MCFDR algorithm, even if used while processing the input files, however, the post "-filter" number of reads from the input files are used.

If flag is omitted: FDR is generated using the FindPeaks 1.0 method, *which is no longer supported*
If trailing integer value is omitted: a default 3 iterations will be run.

Note: in version 3.1.9+, the MCFDR algorithm is dramatically faster than previous versions, and a minimum of 10 iterations is suggested for accuracy.

-minimum <integer>

This sets the minimum peak size to be output. All peaks below this height will not be included in the output files. This may be used with the "-subpeaks" flag, and only sub-peaks above this height will be retained.

If flag is omitted: default value is set to 1.

-name <String>

This is the name of the data set. It's used for naming output files, as well as track names for wig files.

If flag is omitted: defaults to "FP3output"

-no_peaks_header

This flag turns off a header line in the peaks file. When processing for use with a database, it is recommended to turn this off to prevent the need to strip the line out manually.

If flag is omitted: a header line is written to output peaks files.

-one_per

This file allows you to create one wig file per chromosome. Each input file is processed to a separate wig file, however only one peaks file will be created for the collection of processed chromosomes.

If flag is omitted: defaults to all wig data being placed in one file.

-output <String>

Where to put the output files. Should be an existing path. A trailing slash will be appended, if one is not provided.

If flag is omitted: program will not run.

-subpeaks <float>

Turns on the subpeaks module, to perform peak separation.

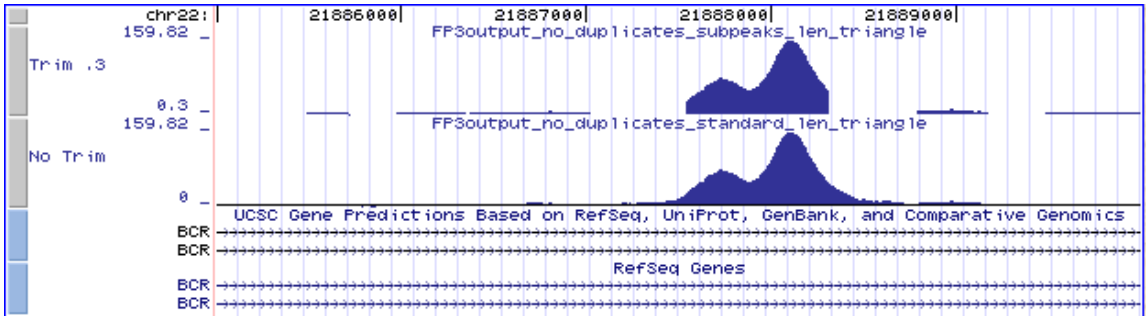
Algorithm: All sequence reads that overlap in an "area of enrichment" are collected and their weights at each position are summed. All positions which are local maxima are identified and collected into an array in sequential order. The array containing the local maxima is then inspected in a local pair-wise manner in which each set of nearest neighbors is identified. The heights of each pair of maxima are then compared, and the lowest value is taken. This value is then multiplied by the float provided with the -subpeaks flag to yield the minimum valley depth required to classify the two peaks as distinct peaks. The intervening area of enrichment between the two local maxima is then searched for values that are lower then the minimum valley depth. If found, the two peaks are then separated, with a single base pair gap, corresponding to the deepest local minima separating the two maxima. If a value lower than the minimum valley depth is not found, the lower of the two peaks is removed from the array of local maxima, and will not appear as a separate peak in the peaks file, and may not appear in the wig file.

If flag is omitted: subpeaks is not turned on and each area of enrichment will be considered as a single peak.

-trim <float>

The float value is used to determine the amount of the shoulder of each peak retained.

When used with the subpeaks algorithm, each separate peak is trimmed individually. To the fraction value provided.



Algorithm: Each local maxima located with the subpeaks method, or the global maxima for the area of enrichment is used as the focus for the trim algorithm. The local or global maxima is selected, and it's value is multiplied by the float value provided with the -trim flag at run time, to yield the shoulder trim minimum. From the location of the maximum, the application then walks one base at a time in either direction towards the "ends" of the peak, and compares the height at that position to the shoulder trim minimum. Once a value is found that falls below the shoulder trim minimum, all positions between that location and the "end" of the peak are set to zero. Note: this may "trim" off local maxima that were not identified by the subpeak algorithm. Those which were identified by the subpeak algorithm as being separate sub-peaks will not be lost.

If flag is omitted, trimming will not be engaged.

-Rmode

Turns on the R script mode. All output comes in the form of an R script, which will produce a postscript representation of a single chromosome.

If flag is omitted: R mode is not used.

EXAMPLES:

Use for generating verbose wig files

- No height cut off
- Histogram not required: set histogram size to 1.
- Use Fixed length mode: set to 200 fragment length size
- Optional: use "-minimum 2" to prevent single reads in areas without enrichment from being written out. This will reduce the size of the files created.
- Optional: use "-one_per" to split up wig files by chromosome, to prevent file size from becoming too large

For users of the pre-packaged jar files:

```
java -jar FindPeaks.jar -input /input_dir/*.part.eland.gz -name test
-dist_type 0 200 -hist_size 1 -eff_size 2.156E9 -output /output_dir/
```

Use for generating postscript files

- use -Rmode flag
- Use a minimum height threshold (improves readability of graphics and decreases processing time required by R. (set according to desired FDR cutoff)

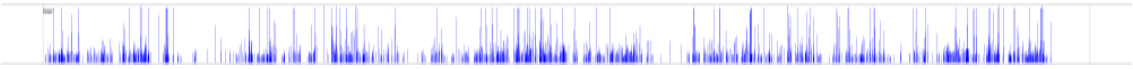
For users of the pre-packaged jar files:

```
java -jar FindPeaks.jar -input /input_dir/*.part.eland.gz
-output /output_dir/ -minimum 15 -Rmode -name test -dist_type 0 200
-eff_size 2.156E9
```

Use R to process output script:

```
source("filename")
```

Example output from the -Rmode flag (Generated from Chr 17 of STAT1 ChiIP-Seq of IFN-G stimulated Stimulated HeLa S3 cells):



Use for ChIP-Seq analysis:

- Use mode 1, median of 200.
- Use a large histogram (100)
- Use subpeaks to identify peaks maxima in overlapping areas of enrichment
- Use trim to remove shoulders.
- Use MCFDR module to estimate FDR rates
- Use minimum to retain only hits above the desired FDR.

For users of the pre-packaged jar files:

```
java -jar FindPeaks.jar -name test -dist_type 1 200 -minimum 8
-hist_size 100 -eff_size 2.156E9 -trim .2  -subpeaks .5 -output
/output_dir/ -input /input_dir/*.part.eland.gz -mcfdr 20
```

Use for transcription factors

- Use mode 1, median of 200
- Use a large histogram (50+)
- Use subpeaks to identify peaks maxima in overlapping areas of enrichment
- Use trim to remove shoulders.
- Use MCFDR module to estimate FDR rates
- Use minimum to retain only hits above the desired FDR.
- Use directional flag to use only hits contributing to a peak.
- Use filter to remove "duplicate" reads

For users of the pre-packaged jar files:

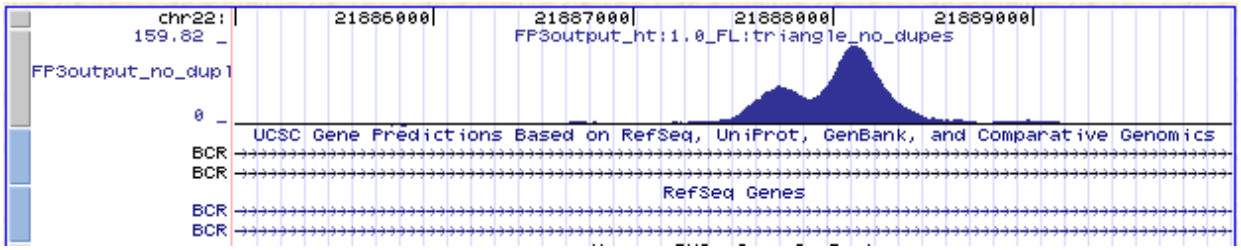
```
java -jar FindPeaks.jar -name test -dist_type 1 200 -minimum 8
-hist_size 100 -eff_size 2.156E9 -trim .4 -subpeaks .7 -output
/output_dir/ -input /path/to/files/*.part.eland.gz -mcfdr 20
-directional -filter
```

OUTPUT:

Wig File:

The wig file produced is a standard gzipped UCSC compatible wig file. For information on the wig file format, please visit http://genome.ucsc.edu/google/goldenPath/help/wiggle.html

Example (generated from test data – see below):



Peaks File:

The peaks file contains the peaks identified using the parameters supplied by the user. The columns are:

ID:	A unique identifier for each peak identified for simplified referencing.
chromosome:	The chromosome where the peak was identified.
Start location:	The start coordinate of the peak, according to the trimming algorithm requested.
End location:	The end coordinate of the peak, according to the trimming algorithm requested.
peak maximum location:	The location of the peak maximum.
maximum height:	The greatest value observed in the region specified by the start and end locations.

Output is produced using a zero based coordinate system.

Example (generated from test data – see below, "-subpeaks 0.7" on.):

id	chrom	start	end	max_coord	score
1	22	21885493	21885792	21885543	1.0
2	22	21885836	21886136	21886136	3.119
3	22	21886138	21886374	21886220	3.971
4	22	21886376	21886499	21886376	1.490
5	22	21886501	21887016	21886796	4.750
6	22	21887018	21887876	21887728	77.201
7	22	21887878	21888791	21888100	159.824
8	22	21888793	21889484	21888977	8.759
9	22	21889486	21889807	21889622	2.630
10	22	21889809	21890185	21889897	2.0

Monte Carlo False Discovery Rate:

The Monte Carlo False Discovery Rate Provides several columns of information at the final state of a run. The columns are:

Height:	The index for the FDR table, and the height (binned) for which statistics are gathered. The number of rows is determined by the -hist_size parameter.
Obs:	The number of Peaks observed while processing the experimental data that were found at this height.
Obs.sum:	The number of Peaks observed while processing the experimental data that were found at this height or greater.
Rand:	The number of peaks found during the Monte Carlo simulation at this height or higher for all iterations of the MC engine.
Rand.sum:	The average number of peaks found for each Monte Carlo simulation at this height or higher. (i.e. Normalized by the number of iterations used.)
Fraction:	The number of peaks observed during the Monte Carlos simulation(s) at this height or greater, divided by the total number of peaks observed during the Monte Carlo simulation(s).
FDR:	The number of peaks observed during the Monte Carlo simulation at this height or greater, divided by the number of peaks observed while processing the experimental data found at this height or greater, normalized by the number of

iterations used to generate the peaks in observed during the Monte Carlo simulation(s). This value can also be considered as a ratio of Noise to Signal in the Experimental data.

Example (generated from STAT1, IFN-g stimulated Hela S3 cells – Robertson et al.)

Height	Obs.	Obs.sum	Rand	Rand.sum	Fraction	FDR
0	0	2812310.0	9	60064256	1.0	21.357622
1	0	2812310.0	2298511	60064248	0.9999999	21.35762
2	1756809	2812310.0	16059673	57765736	0.9617323	20.540316
3	642969	1055501.0	22255149	41706064	0.69435745	39.51305
4	206038	412532.0	8903807	19450914	0.3238351	47.150074
5	78460	206494.0	6044178	10547107	0.17559706	51.077065
6	34861	128034.0	2966414	4502929	0.07496853	35.169792
7	19509	93173.0	1057512	1536515	0.025581188	16.49099
8	13010	73664.0	350059	479003	0.007974843	6.5025387
9	9453	60654.0	98318	128944	0.0021467677	2.1258943
10	7402	51201.0	24134	30626	5.098873E-4	0.5981524

Note: The above example was run with a "-minimum 2" flag, and thus heights below the minimum value are not collected. This is reflected in the FDR, and FDR values for heights below the supplied minimum should not be used.

TEST DATA:

FindPeaks 3.1 9+ comes with a test file (22.test.eland), which contains a 521 lines of data from Robertson et al's STAT1 experiment. (Interferon Gamma stimulated HeLa S3 cells.) When run with FindPeaks, this data set will generate one peak of height 159.824, one peak of height 1 and one peak of height 2 when run with the following options:

```
java -jar FindPeaks.jar -input 22.test.eland -output /output/directory/
-eff_size 5000 -dist_type 1 200 -directional -filter -hist_size 200
```

When run with -subpeaks turned on

```
java -jar FindPeaks.jar -input 22.test.eland -output /output/directory/
-eff_size 5000 -dist_type 1 200 -directional -filter -hist_size 200
-subpeaks .7
```

The following peaks heights are found : (See Output section above)

1.0 1.490 2.0 2.630 3.119 3.971 4.750 8.759 77.201 159.824

Other Java Applications

ALIGNSLICE:

-map

This flag engages the mappability modules, which attempt to generate a mappability track to accompany the requested slice of aligned reads from the genome. Mappability is currently only available for human, and for internal users of the GSC.

If omitted, mappability mode will not be engaged, and mappability tracks will not be generated.

-text

Uses the text file representation of the output, including the canonical sequence and mappability track.

If omitted, text mode will not be engaged. (wig files will be generated instead.)

-chr <string>

The chromosome identifier. For human chromosomes, and many mammals, this will be an integer value (e.g. 1, 3, 5, 22, X, Y, MT)

If omitted, program will not run.

-start <integer>

The start coordinate (zero based) of the slice for which the wig or text file will be generated.

If omitted, program will not run.

-end <integer>

The end coordinate (zero based) of the slice for which the wig or text file will be generated.

If omitted, program will not run.

-out <string>

The filename (and optional path) to which the slice should be written out. If the -text flag is used a .txt file extension will automatically be added. If the -text flag is not used, "._map.wig.gz" and ".wig.gz" extensions will be created for the mappability file (optional) and the wig file generated.

If omitted, program will not run.

-size <integer>

A fixed size (xset) value that must be used to determine the size of the fragment indicated.

If omitted, program will not run.

-species <string>

This string is required for determining the cannoical sequence, as well as the mappability of the organism in the selected region.

Mappability is currently only available for human (See -map).

If omitted, "human" is assumed.

-input

The source Eland or gziped eland files from which to build the list of aligned reads.

If omitted, program will not run.

ELANDTOBED:

-input <Strings>

The full list of files that should be processed using this program. Filenames are preserved from the final slash/backslash in the filename provided. The files must be separated by chromosome prior to using this program. (See SeparateElandReads)

-output <String>

The directory into which the files should be placed.

-name <String>

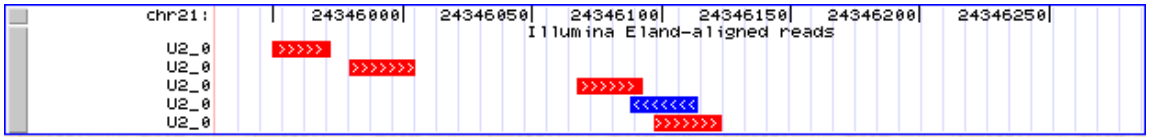
File names are created with the chromosome number, this parameter, and appended with the extension ".bed.gz".

Example:

For users of the pre-packaged jar files:

```
java -Xmx4G -jar ElandtoBed.jar -input /input_dir/*.part.eland.gz
-output /output_dir/ -name HS0419
```

Graphic example of five reads aligned to chromosome 21 using ElandtoBed:



VULGARTOBED:

-input <Strings>

The full list of files that should be processed using this program. Filenames are preserved from the final slash/backslash in the filename provided. The reads do not need to be separated by chromosome before using this program.

-output <String>

The directory into which the files should be placed.

-name <String>

File names are created with the chromosome number, this parameter, and appended with the extension ".bed.gz"

Example:

For users of the pre-packaged jar files:

```
java -Xmx4G -jar VulgartoBed.jar -input
/input_dir/*.part.eland.gz -output /output_dir/ -name HS0419
```

Additional Information

BUG REPORTS:

Bug reports can be sent to afejes@bcgsc.ca, along with a complete description of the problem, and relevant supporting information.

- For all bugs, please send the command used to execute the application, and the output provided by the program.
- If you are getting errors on input, please send the first 10 lines of your input file along with your bug report. (ie. on a linux system, use "head -10 [input_file] > [output_file]". and send the output file with your report.)
- If you have a problem with the output, please send a text file of ALL of the screen output.

FUTURE DEVELOPMENT (road map):

- See the GSC jira ticket filing system at http://gin.bcgsc.ca/jira (Internal BCGSC traffic only)

CREDITS:

FindPeaks 1.0 was written by Matthew Bainbridge, with assistance by Gordon Robertson.

FindPeaks 2 was rewritten by Anthony Fejes, with Q.A testing done by Mikhail Bilenky and Gordon Robertson

FindPeaks 3 was developed by Anthony Fejes based upon the FindPeaks 2.1.4 code.

Triangle Distribution was suggested by Mikhail Bilenky, Genome Sciences Centre, Vancouver, BC.

The author would also like to thank members of the community who have contributed their feedback to improving this document and the FindPeaks program:

Sumit Middha, Mayo Clinic, USA
Blaise T.F. Alako, Nijmegen, The Netherlands