GenomeMapper Manual

version 0.1g

Contents

GenomeMapper is a fast and sensitive alignment tool for short sequence reads generated on the Illumina Genome Analyzer platform. It aligns short reads against multiple genomes simultaneously while allowing for a user-defined number of mismatches and gaps. By default GenomeMapper aligns with 3 mismatches including 1 gap. Based on a hash index, GenomeMapper follows a seed-and-extend approach supplemented by k-banded alignments (similar to Needleman-Wunsch) for accurate and consistent read alignments. GenomeMapper is the only short read alignment tool performing its alignments against multiple genome sequences rather than a single reference sequence. A graph-based index structure stores conserved sequences only once whereas divergent parts are stored separately for each of the genomes. Thus, GenomeMapper can explore the most similar region for each of the reads within all known individuals of a species, rather than aligning against a single genome.

GenomeMapper has been incorporated into the Illumina short read analysis pipeline SHORE (http://1001genomes.org).

GenomeMapper exhibits high versatility:

- Alignments are performed against multiple genomes simultaneously. Alignments against conserved regions are reported only once.

- The target sequence is extendable. Each newly sequenced strain can be incorporated forming a population of genomes rather than a single reference genome.
- GenomeMapper will automatically identify three different file formats. FASTA, FASTQ and SHORE flat file format (see pg. 14).
- GenomeMapper allows for an arbitrary number of mismatches and gaps with a maximum of 10.
- Supported hash index sizes range from 5 to 13. This allows for a multitude of applications, including read mapping to distantly related genomes, mapping of longer reads (i.e. generated by different Next Generation Sequencers) or sRNA, mRNA and ChipSeq reads.
- GenomeMapper reports either all hits or only the best hits.
- GenomeMapper allows the adjustment of mismatch- and gapscores. However, suitable default values facilitate a simple usage.

System requirements

GenomeMapper is written in C and was developed on Linux. It only requires standard C libraries and the C compiler gcc for compilation.

GenomeMapper has varying memory and disk space requirements mainly depending on the reference sequence and the hash size. For example, building an index of hash size 12 of the human genome requires ∼15.8 GB of disk space and ∼18.5 GB RAM during mapping (execution of `genomemapper`). While the index of *Arabidopsis thaliana* (genome size of 120 Mbp) needs ∼1.9 GB of disk space and ∼3.8 GB RAM.

Installation

Download GenomeMapper at http://1001genomes.org/downloads/. Change your working directory to the folder and type:

for Linux:

```
tar -zxf genomemapper.tar.gz
cd genomemapper
make
```

for MAC (64-bit OS):

```
tar -zxf genomemapper.tar.gz
cd genomemapper
mv Makefile.MAC64 Makefile
make
```

The package contains two programs, `gmindex` and `genomemapper`. `gmindex` creates the hash index. `genomemapper` performs the read mapping.

Quick Guide

To build the index with a seed length of `12` of the reference file `genome.fa`, name the reference "Col-0", incorporate the polymorphisms of two strains (in the files `strain1.var.list` and `strain2.var.list`) and store the index in the files `genome.idx` and `genome.mta`, type:

```
$genomemapper-path$\gmindex -i genome.fa -x genome.idx -t genome.mta
-s 12 -d strain1.var.list,strain2.var.list -f Col-0
```

Note that no white space character is allowed within the list of strains of option `-d`.

Two genomic variations, a SNP (`G` to `A`) on position 5 on chromosome 2 and an `AG`-insertion on position 28 on chromosome 4, would result in following strain file (columns are tab-delimited, for more details see page 14):

```
ID     type chr  pos  len  seq
str1   S    2    5    1    A
str1   I    4    29   2    AG
```

In order to align the reads (listed in `reads.fl`) against the genomes type:

```
$genomemapper-path$\genomemapper -i genome.fa -x genome.idx
-t genome.mta -q reads.fl -M 4 -G 2 -E 4
```

-M , -G and -E adjust the maximal number of mismatches. In order to print the mapped reads in `mapped_reads.fl` and the unmapped ones in `unmapped_reads.fl` in the original query input format, and to report the number of edit operations instead of the alignment scores (`-e`), type:

```
$genomemapper-path$\genomemapper -i genome.fa -x genome.idx
-t genome.mta -q reads.fl -M 4 -G 2 -E 4
-o mapped_reads.fl -u unmapped_reads.fl -e
```

Example output (from `mapped_reads.fl`) slightly shortend:

```
strain chr posRef pos2 alignmentRef       alignmentStr read editOp occ..
Est-1  1   25.2   28   (-A)(-G)ATC[AG]GAT AGATC[AG]GAT r1     1    2  ..
Col-0  1   1683   1683 [GA]GATCGGAT       [GA]GATCGGAT r1     1    2  ..
```

`read1` maps against the genome of "Est-1" (chromosome 1 at position 28). Relative to the reference sequence, the read maps to the second inserted base following position 25 of the reference on (`25.2`). Alignments on both the strain and the reference are provided. Squared brackets indicate variations between the read and the index. But round brackets indicate mismatches of the read against the reference sequence, which are supported by a strain.

The pre-processing program `gmindex`

The program `gmindex` constitutes the pre-processing step of GenomeMapper. It translates the reference sequence(s) together with divergent strain sequences into a hash index and constructs the sequence graph. Brief summary:

```
gmindex [options] -i <genome file>
                  -x <index file>
                  -t <meta index file>
```

Parameters (bold ones are mandatory):

-i <genome fasta file>	Specifies the input file containing the reference sequences in multi-FASTA format.
-x <index file>	Specifies the output file containing the target sequence index.
-t <meta index file>	Specifies the output file containing the sequence graph representation and additional meta information about the reference index.
-s <int>	Specifies the seed length. The allowed range is 5 to 13. Default seed length is 12.
-d <list>	Specifies the divergent strain data files, one file for each strain separated by commata (without any whitespace)
-r	Disables the index of the reverse strand of the reference sequence. By default both indices of forward and reverse strand are generated.
-v	Set to verbose mode.

Mandatory parameters

Input data

-i <genome fasta file>

`gmindex` expects a multi-FASTA file containing the reference sequence(s). The string directly following – without any white spaces in between – the '>'-symbol will be interpreted as the header or chromosome/contig description. The maximal allowed length is 50 characters, longer strings will be cut. Moreover, the description ends at any white space character. `gmindex` follows the IUPAC recommendation[1]. Lower case bases are accepted. Any empty lines or lines containing only whitespace characters are ignored.

All IUPAC characters unequal to the four bases A, C, G and T are considered as mismatches in the alignment against every other symbol.

The accumulated genome size, i.e. the sum of the lengths of all chromosomes/contigs, must not exceed $2^{32} - n$ bp with n chromosomes/contigs, where $n < 2^{24}$.

To avoid long computation time due to repeats, repeat masking before using `gmindex` is recommended.

[1] The symbol standard is denoted at
http://www.chem.qmul.ac.uk/iubmb/misc/naseq.html

Output files

-x <index file>

Binary representation of the hash index.

-t <meta index file>

Binary representation of the sequence graph and additional meta information about the hash index needed by `genomemapper` to pre-allocate memory.

Optional parameters

-s <seed length>

Sensitivity of the mapping depends on the seed length. Simple seed strategies require an *identical* sub-string between read and reference sequence of at least seed length. The longer the seed, the lower the runtime of `genomemapper`, but also the lower the sensitivity. Future implementations will incorporate wildcards.

The range of allowed seed lengths is 5 to 13. Default is 12. Note that mapping millions of reads with `genomemapper` with very small seed lengths (< 8) can take days.

-d <strain file1,strain file 2,...>

This parameter specifies the files containing divergent strain data. The file format has to follow the specification given on page 14. For each strain, a separate input file is expected. The list must be separated by commata without any whitespace characters. The polymorphism data provided in the files will be incorporated into the sequence graph.

-f <string>

This option specifies the identifier of the reference sequence. This name will be used in the mapping output file of the program `genomemapper` to identify reads mapping optimally to the reference genome rather than to a specific strain. The identifiers of additional strains are supported in the strain input files (with option `-d`, see page 14).

-r

By default, `gmindex` builds two indexes, one for the forward and the other for the

reverse strand. Setting this flag will switch off the reverse strand of the reference sequence. `genomemapper` won't find any mappings on the reverse strand then.

-v

Set to verbose mode.

The mapping program `genomemapper`

`genomemapper` performs the mapping of a query against a sequence graph. The summarized usage is as follows:

```
genomemapper [options] -x <index file>
                       -t <meta index file>
                       -q <query file>
```

Parameters (bold ones are mandatory):

-x <index file>	Specifies the input file containing the target sequence index which was output of `gmindex`.
-t <meta index file>	Specifies the input file containing the sequence graph and additional meta information about the target sequence index which was output of `gmindex`.
-q <query file>	Specifies the input file containing the reads which should be mapped onto the reference. FASTA, FASTQ and SHORE flat file format are accepted (see pg. 14).
-o <output file>	Specifies the output file storing all reported mappings. Default is standard out.
-f <format>	Specifies the output file format. <format> has to be either "shore" or "bed". Default is "shore".
-u <unmapped reads file>	Specifies the output file containing all reads which could not be mapped on the reference.
-s <string>	Specifies the identifier of the reference sequence. Default is 'Ref'.
-r	Disables the mapping of the reads against the complementary reverse strand of the reference. Default is to map against both strands.

-a	Enables all-hit strategy. All alignments of the reads against the reference which fulfill the mapping criterias defined by the `-E`, `-M` and `-G` flags will be reported and printed to stdout or in the file specified by `-o`. Default is a best-hit strategy which reports only hits with the highest score. Caution, strongly increases computation time.
-n <int>	Specifies the maximal number of randomly selected best scored alignments per read. This option works only in combination with best-hit strategy. Reporting all best hits is the default setting.
-M <int>	Specifies the maximal number of allowed mismatches per alignment. Default is 3.
-G <int>	Specifies the maximal number of allowed gapped positions in the read or reference per alignment. Default is 1.
-E <int>	Specifies the maximal number of allowed edit operations per alignment, i.e. the number of mismatches plus the number of gaps must not exceed this value. Default is 3.
-m <double>	Specifies the penalty score of a mismatch in the alignment. It must be a positive floating point number. Default is 4.0.
-g <double>	Specifies the penalty score of a gap in the alignment process. It must be a positive floating point number. Default is 5.0.
-e	Instructs `genomemapper` to print out the number of edit operations needed to align the read instead of the alignment score.
-d	Instructs `genomemapper` to place gaps most right in gapped alignments. Default is most left.
-h	Instructs `genomemapper` to perform Needleman-Wunsch alignments on the complete read sequence. With `-h`, slightly higher sensitivity can be achieved.
-l <int>	Increases the seed length. To speed up the mapping process the seed length can be increased compared to the index file. Larger values descrease sensitivity, but speed up the mapping. Default is the index seed length.
-w	Softens stringent gap limit. If the best alignment contains more gaps than specified by `-G` or `-E`, it will be reported, nevertheless.
-c <int>	Specifies the number of seeds on the reference sequence that can be found/stored per read. Large values lead to extensive memory usage. Default and minimum value is 15.000.000. The value of `-c` will be added to the default size.
-v	Set to verbose mode.

Mandatory parameters

Input data

-x <index file>

The index file produced by `gmindex`.

-t <meta index file>

The meta index file produced by `gmindex`. The seed length will be read from this file.

-q <query file>

The query file contains the reads which will be mapped onto the reference. The format of this file can either be ordinary Multi-FASTA, FASTQ[2] or SHORE flat file format, which will be described in the section "File format specifications" on page 14. As for the genomic file, only IUPAC characters are accepted and non-base symbols count as mismatches.

To avoid long computation times due to repetitive reads, (quality and/or low complexity) pre-filtering the reads is recommended.

Any empty lines or lines containing only whitespace characters are ignored.

Optional parameters

Output files

-o <output file>

Output file in the format specified by the `-f` option.

-f shore or *-f bed*

Output file format. Either SHORE or BED format can be specified. For a detailed explanation of both formats refer to the section "File format specifications" on page 14. BED formatted files can only report one alignment (on the reference *or* strain sequence).

[2]FASTQ file format specification can be found at http://maq.sourceforge.net/fastq.shtml

Default is SHORE format including an intuitive human readable alignment string.

-u <unmapped reads file>

The reads which could not be mapped on the reference will be reported in this file in the input file format.

-s <string>

This option specifies the identifier of the reference sequence. This name will be used in the mapping output file to identify reads mapping optimally to the reference genome rather than to a specific strain. No white space characters are allowed. The identifiers of additional strains are supported in the strain input files (with option `-d`, see page 14). Default is "Ref".

Options

-r

If this flag is set, then `genomemapper` does not read in the index of the complementary reverse strand of the reference and thus, cannot report mappings onto the minus strand.

-a

This flag switches from best-hit strategy to all-hit strategy. Not only the hits with best scores are reported in the output stream, but every hit fulfilling the mapping criterias specified with `-E`, `-M` and `-G`. In this mode, `genomemapper` omits a major speedup step of the best-hit mode. Since the all-hit strategy performs more alignments than the best-hit strategy, it is slower and can be unfeasible slow for huge reference sets or small seed lengths.

Default is best-hit strategy.

-n <int>

The maximal number of randomly chosen hits being reported, works only with best-hit strategy. By default all alignments are reported.

-M <int>

This value specifies the maximal number of mismatches in the alignment [0-10]. Default

is 3.

`genomemapper` can find all n-mismatch mappings for $n < \left\lfloor \frac{\text{read length}}{\text{seed length}} \right\rfloor$, otherwise some hits can be missed where the mismatches are distributed in a way that there is no continuous stretch of identical bases which has the length of the seed length.

-G <int>

This value specifies the maximal number of gapped positions in the alignment [0-10]. Each position in a gap is counted separately. Default is 1.

Beware that the number of gaps can drastically influence the runtime.

It is recommended to set the number of gaps smaller than the number of mismatches.

-E <int>

Maximal number of edit operations per alignment [0-10]. Edit operations are mismatches plus gaps (also called Levenshtein distance). Default is 3.

`-E` controls `-M` and `-G`: If the number of edit operations is set to a smaller value than the number of mismatches and gaps, both are adjusted to the number of edit operations.

`-E` allows for complex requests: for instance, if you want to find reads with a Levenshtein distance of 4, but don't want to allow for 4 gaps, you can specify `-E 4`, `-M 4` and `-G 3` for 3 gaps.

-m <double>
-g <double>

The user can define the penalty scores of mismatches and gaps in the alignment process. They must be positive floating point numbers. The default setting is 4.0 for mismatches and 5.0 for gaps. That way an alignment with n mismatches is always preferred to an alignment with n gaps and an alignment with $n-1$ gaps is preferred to an alignment with n mismatches up to $n = 4$.

Note that the score for a match is 0. For accurate alignments, mismatch and gap penalties should be greater than 0.

-e

With this flag, `genomemapper` is instructed to print the number of edit operations instead of the alignment scores (default) in the output file.

-d

This flag causes `genomemapper` to position gaps most right in gapped alignments. Default is most left.

-h

This flag causes `genomemapper` to enforce an alignment of the whole read sequence to the reference. By default `genomemapper` aligns only sub-strings of reads which have not been used for the seed mapping. To ensure the uniqueness of the alignments gapped alignments are always performed on the whole read.

Activating `-h` will yield slightly more sensitive mappings.

-l <int>

Increase the seed length compared to the index. Higher values yield lower sensitivity, but speed up the mapping process. Default setting is the seed length, which means that no hit will be discarded.

The range of values is from seed length to read length.

-w

The gap limit `-G` is softened with this option. In seldom cases where best alignments don't fulfill the mapping criterias worse alignments are reported instead. For instance, if alignments with 2 gaps are superior to alignments with 3 or 4 mismatches, but only 1 gap and 4 mismatches were allowed, the worse alignment fulfills the mapping criteria whereas the better one not.

Default is to uphold the stringent gap limit and not report mappings with more gaps than desired at the cost of a slightly worse mapping sensitivity.

-c <int>

This value specifies how many seeds on the sequence graph can be stored per read (seed container size). The default value of 15,000,000 is concomitantly the minimum size, i.e. parameter values for `-c` will be added to this number (`-c 1000000` will result in a container size of 16,000,000). The default setting should work well for almost all practical cases, i.e. seed lengths bigger than 6 on moderately large reference sequences, but for small seed lengths and/or highly repetitive genomes it might have to be increased, otherwise not all hits can be processed and reported. In this case, a warning is printed out which indicates the reads which could not have been mapped completely. These reads can be mapped in a subsequent mapping run with a higher value for option `-c`.

Higher values of the container size cause the program to allocate more memory.

For instance, the genome of *Arabidopsis thaliana*, 120 Mb, requires a container size of

20,200,000 for seed length 5.

-v

Set to verbose mode.

File format specifications

SHORE input format

The flat file format files stores read information from Illumina GA2 sequencers and *must* be tabulator-delimited. The columns are as follows in this order:

1. Read ID.
2. Read sequence.
3. Paired-end flag. Not used by `genomemapper`, used by SHORE.
4. Quality value string 1. Assumed to have exactly the same length as the read sequence. Not used by `genomemapper`.
5. Quality value string 2. See 4.
6. Quality value string 3. See 4.

Strain file input format

The files containing the polymorphisms of strains compared to the reference sequence have to be tabulator-delimited. The entries *must* be sorted by ascending chromosome and position relative to the reference sequence. If both values are equal, insertions *must* be preferred over deletions and SNPs. The files must consist of following columns in this order:

1. Strain name.
2. Type of polymorphism (S for SNP, I for insertion, D for deletion).
3. Chromosome ID.
4. Position relative to the reference genome. Insertions will be incorporated before the specified position.

5. Length of the polymorphism.

6. Sequence of the polymorphism.

SHORE output format

GenomeMapper's SHORE output format stores one mapping per row. If the input read file is in FASTA format, only the columns 1 to 9 described below are reported, for FASTQ only columns 1 to 9 and 11 and in case of SHORE flat file format all 13 columns will be printed.

The columns are tabulator delimited.

1. Strain name

2. Chromosome/Contig ID as specified in the input fasta file of `gmindex` (`-i` option)

3. Left-most reference position of the alignment (positions starting from 1)

4. Left-most strain position of the alignment if it is on the strain or left-most reference position if it is on the reference (positions starting from 1)

5. Alignment on the reference sequence: Matching bases are reported once, whereas mismatches and gaps are represented as two characters in squared brackets. The first character refers to the reference and the second to the read. These sequences are always on the positive strand of the reference. Gaps are reported as dashes ('-'). For alignments on a divergent strain, mismatches or gaps present in this strain, but not in the reference, are represented as two characters in parentheses. The first character refers to the reference and the second to the read.

6. Alignment on the strain sequence: Matching bases are reported once, whereas mismatches and gaps are represented as two characters in squared brackets. The first character refers to the strain and the second to the read. These sequences are always on the positive strand of the strain. Gaps are reported as dashes ('-').

7. Read ID as specified in the query file (`-q` option)

8. Strand: 'D' for the forward, 'P' for the reverse

9. Alignment score or number of edit operations (`-e` option)

10. Number of (reported) mappings of the read

11. Read length

12. Undefined, SHORE interface.
13. Paired-End flag, SHORE interface.
14. Illumina GA2 quality, SHORE interface.
15. Sanger scaled quality, SHORE interface.
16. Chastity value, SHORE interface.

BED output format

GenomeMapper prints the first 6 fields of the BED file format specification which can be found at http://genome.ucsc.edu/FAQ/FAQformat#format1. Only the alignment to the specific strain or reference sequence is printed as opposed to the SHORE output format.

The following columns are printed tabulator delimited:

1. Chromosome/Contig ID as specified in the input fasta file of `gmindex` (`-i` option)
2. Left-most reference (or strain) position of the alignment (positions starting from 0)
3. The ending position of the alignment in the reference (or strain). The specified position is not included in the display of the alignment.
4. Read ID as specified in the query file (`-q` option)
5. Alignment score or number of edit operations (`-e` option)
6. Strand: '+' for the forward, '-' for the reverse

Contact

GenomeMapper was written by Korbinian Schneeberger, Jörg Hagmann and Stephan Ossowski at the Max Planck Institute for Developmental Biology, Tübingen, Germany in 2008/09. We appreciate any kind of feedback. Please do not hesitate and contact us in any case of problems or questions:

korbinian.schneeberger@tuebingen.mpg.de
stephan.ossowski@tuebingen.mpg.de

joerg.hagmann@tuebingen.mpg.de