GenomeMapper Manual

version 0.1beta

Contents

GenomeMapper is a fast and sensitive alignment tool for short sequence reads generated on the Illumina Genome Analyzer platform. It maps short reads onto a reference sequence allowing for a user-defined number of mismatches and gaps with a maximum of 10. Limiting factors are time and sensitivity of the resulting mappings. By default GenomeMapper aligns with 3 mismatches including 1 gap. Based on a hash index, GenomeMapper follows a seed-and-extend approach supplemented by a k-banded alignment (similar to Needleman-Wunsch) for accurate and consistent read alignments.

GenomeMapper has been incorporated into the Illumina Sequencing project software suite SHORE (http://1001genomes.org).

GenomeMapper exhibits high versatility:

- Reads can be provided in three different file formats. FASTA, FASTQ and SHORE flat file format (see pg. **??**) will be automatically identified by GenomeMapper.
- GenomeMapper allows for an arbitrary number of mismatches and gaps.
- Supported hash index sizes range from 5 to 13. This allows for a multitude of applications, including read mapping to distantly related genomes, mapping longer reads from other Next Generation Sequencers, sRNA, mRNA and ChipSeq reads.
- GenomeMapper reports either all hits for a given read or all best hits. In the latter case, GenomeMapper allows to randomly assign or drop repetitive reads.
- GenomeMapper allows the adjustment of mismatch- and gapscores. However, suitable default values facilitate a simple usage.
- GenomeMapper supports two output file formats: BED files as well as SHORE files (pg. **??**), the latter including an intuitive human readable alignment string.

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 index size. For example, building an index of hash size 12 of the human genome needs ∼15.8 GB of disk space and ∼18.5 GB RAM during mapping (execution of `genomemapper`). With the same seed length, the index of the *Arabidopsis thaliana* genome (120 Mbp) needs ∼1.9 GB of disk space and ∼3.8 GB RAM.

Minimal RAM requirements can be calculated as

$$(768 \cdot r) + \left(\frac{g + n \cdot r \cdot 4 + l \cdot p + (7 + 2 \cdot p) \cdot c}{1.048.576} \right) \text{MB}$$

of memory for `genomemapper`, where:
r has value 2 if both indexes (forward and reverse) are used and 1 if only forward,
g is the total size of the reference sequence(s) in base pairs,
n is the number of occurrences of all seeds from the reads in the genome; it will be printed out from `mkindex` if the verbose mode is enabled,
l is the length of the longest chromosome/contig in the reference sequence [bp],
p is the size of a pointer in byte (for 32-bit machines usually 4 bytes, 64-bit 8 bytes),
c is the hit container size (see option `-c`, pg. **??**).

Installation

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

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

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

Quick Guide

To build the (forward and reverse) index of the reference file `genome.fa` with seed-length X, type:

```
$mkindex-path$\mkindex -g genome.fa -i genome.idx -m genome.meta -s X
```

To map the query file `reads.fl` against the reference `genome.fa` allowing for 4 mismatches and 2 gaps, but in total only 4 edit operations, execute the following:

```
$genomemapper-path$\genomemapper -g genome.fa -i genome.idx
-m genome.meta -q reads.fl -M 4 -G 2 -E 4
```

To print the mapped reads in `mapped_reads.fl` in SHORE's flat file output format and the unmapped ones in `unmapped_reads.fl` in the original query input format, the statement looks as follows:

```
$genomemapper-path$\genomemapper -g genome.fa -i genome.idx
-m genome.meta -q reads.fl -M 4 -G 2 -E 4 -o mapped_reads.fl
-u unmapped_reads.fl
```

To use different file formats than the SHORE formats, only the desired output file format has to be specified, the input format is automatically detected. For example, a query file in FASTA format is mapped and the output should be printed in BED format including the number of edit operations instead of the alignment scores:

```
$genomemapper-path$\genomemapper -g genome.fa -i genome.idx
-m genome.meta -q reads.fa -M 4 -G 2 -E 4 -o mapped_reads.fl -f bed
-u unmapped_reads.fa -e
```

Note that the unmapped read output format is always the read input format.

The pre-processing program `mkindex`

The program `mkindex` constitutes the pre-processing step of GenomeMapper. It translates the reference sequence into a hash index. Brief summary:

```
mkindex [options] -g <genome file> -i <index file> -m <meta index file>
```

Parameters (bold ones are mandatory):

-g <genome fasta file>	Specifies the input file containing the reference sequences in multi-FASTA format.
-i <index file>	Specifies the output file containing the reference index.
-m <meta index file>	Specifies the output file containing additional meta information about the reference index.
-s <int>	Specifies the seedlength. The allowed range is 5 to 13. Default seedlength is 12.
-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

-g <genome fasta file>

`mkindex` expects a multi-FASTA file containing the reference sequence(s). `mkindex` follows the IUPAC recommendation. The symbol standard is denoted at http://www.chem.qmul.ac.uk/iubmb/misc/naseq.html. Lower case bases are accepted.

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 `mkindex` is recommended.

Output files

-i <index file>

Binary representation of the hash index.

-m <meta index file>

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

Optional parameters

-s <seedlength>

Sensitivity of the mapping depends on the seedlength. Simple seed strategies require an *identical* sub-string between read and reference sequence of at least seedlength. 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 seedlengths is 5 to 13. Default is 12. Note that mapping millions of reads with `genomemapper` with very small seedlengths (< 8) can take days.

-r

By default, `mkindex` 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 reference. The summarized usage is as follows:

```
genomemapper [options] -g <genome file> -i <index file>
                       -m <meta index file> -q <query file>
```

Parameters (bold ones are mandatory):

-g <genome file>	Specifies the input file containing the reference sequences in multi-FASTA format. Must be the same as for `mkindex`.
-i <index file>	Specifies the input file containing the reference index which was output of `mkindex`.
-m <meta index file>	Specifies the input file containing additional meta information about the reference index which was output of `mkindex`.
-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. **??**).
-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.
-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.
-a <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.
-mismatch_score <double>	Specifies the penalty score of a mismatch in the alignment. It must be a positive floating point number. Default is 4.0.
-gap_score <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 seedlength. To speed up the mapping process the seedlength can be increased compared to the index file. Larger values descrease sensitivity, but speed up the mapping. Default is the index seedlength.
-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 hits that can be stored per read. Large values lead to extensive memory usage. Default is 15.000.000.
-v	Set to verbose mode.

Mandatory parameters

Input data

-g <genome file>

The reference file must be the same as used for `mkindex`.

-i <index file>

The index file produced by `mkindex`.

-m <meta index file>

The meta index file produced by `mkindex`. The seedlength 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[1] or SHORE flat file format, which will be described in the section "File format specifications" on page **??**. 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 **??**.

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.

---

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

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 seedlengths.

Default is best-hit strategy.

-a <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 $n < \left\lfloor \frac{\text{readlength}}{\text{seedlength}} \right\rfloor$, 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 seedlength.

-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.

-mismatch_score <double>
-gap_score <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 seedlength, which means that no hit will be discarded.

The range of values is from seedlength to readlength.

-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 hits can be stored per read. The default value of 15.000.000 works well for almost all practical cases, i.e. seedlengths bigger than 6 on moderately large reference sequences, but for small seedlengths 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 bigger container size than 15.000.000 for seedlength 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`.

SHORE output format

GenomeMapper's SHORE output format stores one mapping per row. If the input 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. Chromosome/Contig ID as specified in the input fasta file (`-g` option)
2. Left-most reference position of the alignment (positions starting from 1)
3. Alignment: 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 ('-').
4. Read ID as specified in the query file (`-q` option)
5. Strand: 'D' for the forward, 'P' for the reverse
6. Alignment score or number of edit operations (`-e` option)
7. Number of (reported) mappings of the read
8. Readlength
9. Undefined, SHORE interface.
10. Paired-End flag, SHORE interface.
11. Illumina GA2 quality, SHORE interface.
12. Sanger scaled quality, SHORE interface.
13. 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.

The following columns are printed tabulator delimited:

1. Chromosome/Contig ID as specified in the input fasta file (`-g` option)
2. Left-most reference position of the alignment (positions starting from 0)
3. The ending position of the alignment in the reference. 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, Stephan Ossowski and Jörg Hagmann at the Max Planck Institute for Developmental Biology, Tübingen, Germany in 2008. 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