Forschungszentrum Borstel
Leibniz Lungenzentrum

A computational pipeline for whole genome sequencing analysis of Mycobacterium tuberculosis complex isolates

MTBseq

[DESCRIPTION]

MTBseq is an automated and modular bioinformatics pipeline for the analysis of next generation sequencing data of *Mycobacterium tuberculosis* complex samples by performing reference mapping, variant detection, annotation of variants for resistance association and phylogenetic information, and comparative analysis of a set of samples. The pipeline consists of open source software for read mapping (SAMTOOLS, BWA), base call recalibration and refinement (PICARD, GATK), and variant calling (SAMTOOLS), with the full workflow and additional functionality implemented in Perl.

The modular architecture allows the pipeline to be fully customizable, including swapping out or adding entire complete functional modules. Per default, the pipeline uses the genome of *M. tuberculosis* H37Rv (NC_000962.3) as basis for reference mapping, variant calling and annotation, any bacterial reference genome can be set by the user if the appropriate sequence and annotation files are provided.

In addition to the sample specific workflow, MTBseq performs a comparative variant analysis for a set of samples, determining likely related samples by pairwise distance, and constructing an alignment of informative variant positions in FASTA format, which can be used to calculate phylogenetic trees.

TABLE OF CONTENTS

[REQUIREMENTS]

Perl: perl 5, version 22, subversion 1 (v5.22.1) or higher
Java: Oracle Java 8 or OpenJDK 8

MTBseq uses the following CPAN modules:

- MCE (v1.833)
- Statistics::Basic (v1.6611)
- FindBin (v1.51)
- Cwd (v3.62)
- Getopt::Long (v2.5)
- File::Copy (v2.30)
- List::Util (v1.49)
- Exporter (v5.72)
- vars (v1.03)
- lib (v0.63)
- strict (v1.09)
- warnings (v1.34)

MTBseq uses the following third party software:
Binaries (compiled on Ubuntu 16.04) are included

- bwa (v0.7.17)
- GenomeAnalysisTK (v3.8)
- picard (v2.17.0)
- samtools (v1.6)

Hardware requirements:
MTBseq is able to run on 1 thread up to a maximum of 8 threads. RAM usage is highly dependent on the size of FASTQ files and the number of datasets chosen for a joint analysis. 20-25GB of RAM is recommended. Required hard drive space is depended on the FASTQ file size accordingly. A dataset consisting of two 250MB FASTQs will need approximately 2.5GB disk space taking into account the files produced during analysis.

[INSTALLATION]

The following installation guide explains how to install MTBseq on a computer running the Linux operating system and has been thoroughly tested for Ubuntu 16.04 LTS. If necessary, please adapt accordingly for other Linux distributions. If you want to use MTBseq with a different OS, the easiest option is to use a virtual machine running the Linux OS.

1. **Download MTBseq from github and extract:**
   https://github.com/ngs-fzb/MTBseq_source/
   Or clone the repository:
   git clone https://github.com/ngs-fzb/MTBseq_source
   Check whether MTBseq.pl and the executables from the opt/ directory are executable and change permissions with chmod **where necessary.**

2. **You can ensure that MTBseq is executable from anywhere on your system by adding the MTBseq_source folder to your PATH variable or creating a symlink of MTBseq.pl to a folder that is already in your PATH variable. Otherwise, call MTBseq.pl with its full path, e.g.:**
   perl /home/$USER/path/to/MTBseq_source/MTBseq.pl

3. **Install modules via CPAN by typing in the command-line:**
   cpan

4. **Install the modules by typing:**
   install MCE
   install Statistics::Basic
   install FindBin
   install Cwd
   install Getopt::Long
   install File::Copy
   install List::Util
   install Exporter
   install vars
   install lib
   install strict
   install warnings

5. **If third party programs (e.g. BWA and SAMTOOLS) in MTBseq/opt/ are not working correctly, try to re-compile them (this may even be necessary for Ubuntu 14.04). The re-compiled executables MUST be located within the appropriate folders:**
   For BWA:
   cd path/to/MTBseq_source/opt/bwa_0.7.17/
   make
   For SAMTOOLS:
   cd path/to/MTBseq_source/opt/samtools_1.6/
   ./configure --prefix=[PATH_TO_YOUR_MTBSEQ]/opt/samtools_1.6 /
   make
   make install

If BWA or SAMTOOLS are already installed on your system you can create a symlink to the binaries in the appropriate folders of MTBseq.

[OVERVIEW]

MTBseq is executed from the command line and upon invocation will create its own working environment consisting of dedicated folders for the different output files created during runtime of the pipeline. Next generation sequencing data is entered into the pipeline by placing FASTQ files into the folder in which MTBseq is executed. The complete workflow can be run from a single command, or individual functional modules can be invoked directly.

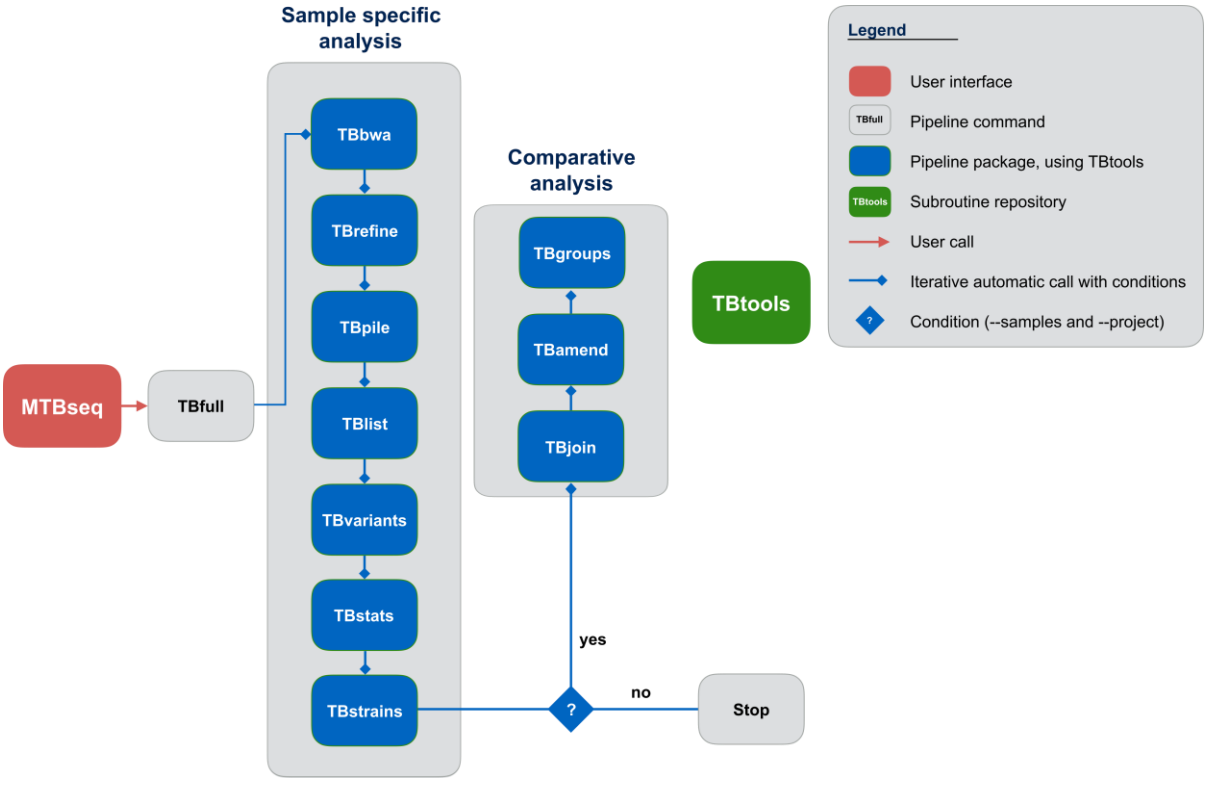
The input for MTBseq is next generation sequencing data in the form of FASTQ files from single end or paired end sequencing runs (tested for Illumina and Ion Torrent files). These files need to confirm to a certain naming scheme, in that they contain no other special characters other than hyphens and possess **at least** three fields separated by **underscores**:

[SampleID]_[LibID]_[*]_[Direction].f(ast)q.gz

example: H37Rv_library789_R1.fastq.gz + H37Rv_library789_R2.fastq.gz

Here, **[SampleID]** represents the identifier for a specific bacterial sample and **[LibID]** is an identifier for the next generation sequencing library used. **[Direction]** is an **essential** field and indicates if reads are in forward (R1) or reverse (R2) orientation in paired end data. Files for single end data have to use the **[Direction]** R1. Other than these, file names can be freely given, including further **[*]** fields.
During execution, the program will assume a certain combination of **[SampleID]** with a **[LibID]** as unique dataset identifier, and collect data for this identifier in e.g. statistics lists. Allowed FASTQ file extensions are fastq.gz or fq.gz.
The basic invocation of MTBseq on the command-line can be modified with OPTIONS and the corresponding VALUES that enable the user to call functional modules directly and set parameters:

MTBseq [OPTIONS] [VALUES]



[OPTIONS & VALUES]

Using OPTIONS and their respective VALUES, a user can modify the execution of MTBseq by directly calling functional modules and supplying parameters used in the analysis.

[OPTIONS]

[**VALUES]**

--step

The OPTION **--step** is essential and requires a user specified VALUE. MTBseq has a modular architecture with functional modules encapsulating distinct steps in the analysis pipeline. Using the **--step** OPTION, you can specify whether the full pipeline is executed, call a specific pipeline step only or as starting point for the pipeline. During execution, MTBseq will only create output files not already present in the working environment. If you choose a specific pipeline step, make sure that the output files of previous steps needed as input are present. In the following, all possible VALUES for this OPTION are described:

TBfull

VALUE of **--step** for executing the whole pipeline. If you choose this VALUE, make sure that you specify all analysis parameters for which you do not want to use the preset defaults using the appropriate OPTIONS.

TBbwa

VALUE of **--step** for mapping next generation sequencing read files to a reference genome, using BWA mem. This is the first step within the pipeline. Depending on the library type (single-end or paired-end), this module will align single end or paired end FASTQ files to a reference genome. Please ensure that FASTQ files used as input follow the required naming scheme described above.

By default, MTBseq uses the *Mycobacterium tuberculosis* H37Rv genome (NC_000962.3) as a reference and maps reads with the BWA mem program with default settings. After mapping, files (.sam) are converted into binary mapping files (.bam). The mapping is sorted, indexed and putative PCR duplicates are removed, using the program SAMTOOLS. Wherever possible, multi-threading is activated by the user provided VALUES for the **--threads** OPTION.

Input:
[SampleID]_[LibID]_[*]_[Direction].fastq.gz
Output:
Bam/[SampleID]_[LibID]_[*].bam
Bam/[SampleID]_[LibID]_[*].bai
Bam/[SampleID]_[LibID]_[*].bamlog

TBrefine

VALUE of **--step** for realignment around insertions and deletions and base call recalibration, using the program GATK. The GATK program is invoked with the parameters:

--downsample_to_coverage 10000 --defaultBaseQualities 12 --maximum_cycle_value 600 --noOriginalAlignmentTags

For base call recalibration, MTBseq employs a set of known resistance associated variants if *Mycobacterium tuberculosis* H37Rv was used as reference genome. The calibration list is stored

in the directory "var/res/MTB_Base_Calibration_List.vcf" of the package. For other reference genomes, the file needs to be specified with the **--basecalib** OPTION or this step will be skipped.

Input:
Bam/[SampleID]_[LibID]_[*].bam
Output:
GATK_Bam/[SampleID]_[LibID]_[*].gatk.bam
GATK_Bam/[SampleID]_[LibID]_[*].gatk.bai
GATK_Bam/[SampleID]_[LibID]_[*].gatk.bamlog
GATK_Bam/[SampleID]_[LibID]_[*].gatk.grp
GATK_Bam/[SampleID]_[LibID]_[*].gatk.intervals

TBpile

VALUE of **--step** for creating pileup files (.mpileup) from refined mapping files (.gatk.bam), using the program SAMTOOLS. The SAMTOOLS program is executed with the parameters -B -A

Input:
GATK_Bam/[SampleID]_[LibID]_[*].gatk.bam
Output:
Mpileup/[SampleID]_[LibID]_[*].gatk.mpileup
Mpileup/[SampleID]_[LibID]_[*].gatk.mpileuplog

TBlist

VALUE of **--step** for creating position lists from pileup files (.gatk.mpileup). Position list files capture the essential data from a mapping in a table format with 21 columns, containing the nucleotide counts in mapped reads for each position of the reference genome. This step will take into account the **--threads** and the **--minbqual** OPTIONS.

The columns in the position list file are:

Pos	Indicates the reference genome position
Insindex	An index for reporting insertion sites, with insertion positions numbered in ascending order
RefBase	The reference genome nucleotide at this position
As	Number of forward mapped reads indicating adenine
Cs	Number of forward mapped reads indicating cytosine
Gs	Number of forward mapped reads indicating guanine
Ts	Number of forward mapped reads indicating thymine
Ns	Number of forward mapped reads with ambiguous nucleotide calls
GAPs	Number of forward mapped reads indicating a GAP (a deletion)
as	Number of reverse mapped reads indicating adenine
cs	Number of reverse mapped reads indicating cytosine
gs	Number of reverse mapped reads indicating guanine
ts	Number of reverse mapped reads indicating thymine
ns	Number of reverse mapped reads with ambiguous nucleotide calls
gaps	Number of reverse mapped reads indicating a gap (a deletion)
Aqual	Number of reads indicating adenine with a phred score of at least 20
Cqual	Number of reads indicating cytosine with a phred score of at least 20
Gqual	Number of reads indicating guanine with a phred score of at least 20
Tqual	Number of reads indicating thymine with a phred score of at least 20
Nqual	Number of reads with an ambiguous nucleotide call with a phred score of at least 20
GAPqual	Number of reads indicating a deletion GAPs with a phred score of at least 20

Input:
Mpileup/[SampleID]_[LibID]_[*].gatk.mpileup
Output:
Position_Tables/[SampleID]_[LibID]_[*].gatk_position_table.tab

TBvariants

VALUE of **--step** for variant detection from position lists. Variant calling can be run in different detection modes using the OPTIONS **--all_vars**, **--snp_vars**, and **--lowfreq_vars** (with the respective changes to the detection algorithm explained in detail in the description for these OPTIONS). In the default mode, a variant is called if it is supported by a minimum number of 4 reads in both forward and reverse direction, by a minimum number of 4 reads indicating the allele with a phred score of at least 20, and if the allele is indicated by a minimum frequency of 75% in mapped reads. These thresholds can all be set using the respective modifiers (**--mincovf**, **--mincovr**, **--minphred20**, **--minfreq**). In addition, positions fulfilling these thresholds will be counted as 'unambigous' in the calculation of quality values for the dataset.

The parameters used by the pipeline will be recorded in the file names by dedicated fields, and the detection mode will be recorded in the output files as a binary string (e.g. "outmode100" indicates **--all_vars** but not **--snp_vars** and **--lowfreq_vars** active).

Input:
Position_Tables/[SampleID]_[LibID]_[*].gatk_position_table.tab
Output:
Called/[SampleID]_[LibID]_[*].gatk_position_uncovered_[mincovf]_[mincovr]_[minfreq]_[minphred20]_[all_vars][snp_vars][lowfreq_vars].tab
Called/[SampleID]_[LibID]_[*].gatk_position_variants_[mincovf]_[mincovr]_[minfreq]_[minphred20]_[all_vars][snp_vars][lowfreq_vars].tab

TBstats

VALUE of **--step** for calculating an overview of mapping quality and detected variants for a dataset, using the SAMTOOLS flagstat program. This step creates or updates a tabular delimited file "Mapping_and_Variant_Statistics.tab". This file stores all sample statistics for the analyzed datasets present in the working environment. This module employs the same thresholds to discern unambiguously covered positions as the TBvariants module (set by the **--mincovf**, **--mincovr**, **-minphred20**, **--minfreq**).

The columns of the output are:

Date	The date of MTBseq execution
SampleID	The sample ID
LibraryID	The library ID
FullID	Complete dataset name
Total Reads	The total amount of sequenced reads
Mapped Reads (%)	The percentage of reads mapped to the reference genome
Genome Size	The size of the reference genome
Genome GC	The GC content of the reference genome
(Any) Total Bases (%)	Percentage of the reference genome covered by reads
(Any) GC-Content	GC content of the reference genome covered by reads
(Any) Coverage mean	Mean coverage depth
(Any) Coverage median	Median coverage depth
(Unambiguous) Total Bases (%)	Percentage of the reference genome covered unambiguously
(Unambiguous) GC-Content	GC content of the reference genome covered unambiguously
(Unambiguous) Coverage mean	Mean coverage depth of unambiguously covered positions
(Unambiguous) Coverage median	Median coverage depth of unambiguously covered positions
SNPs	Number of detected SNPs
Deletions	Number of detected deletions
Insertions	Number of detected insertions
Uncovered	Positions of the reference genome not covered by a read
Substitutions (including Stop Codons)	Number of substitutions within genes

Input:
Bam/[SampleID]_[LibID]_[*].bam
Position_Tables/[SampleID]_[LibID]_[*].gatk_position_table.tab
Output:
Statistics/Mapping_and_Variant_Statistics.tab

TBstrains

VALUE of **--step** for lineage classification based on a set of phylogenetic SNPs (Homolka et al., 2012; Coll et al., 2014; Merker et al., 2015). This module creates a tabular delimited file within the "Classification" directory. Within this file, the majority lineage is reported for each dataset. This entry also gives an indication of the data quality for the positions used to infer the phylogenetic classification. The column indicating the quality will contain the label “good” if all phylogenetic positions contained in the set are covered at least 10fold and show a frequency of at least 75%, “bad” if any phylogenetic position does not meet these, and “ugly” if any phylogenetic position does not have a clear base call.

Input:
Position_Tables/[SampleID]_[LibID]_[*].gatk_position_table.tab
Output:
Classification/Strain_Classification.tab

TBjoin

VALUE of **--step** for creating a comparative SNP analysis of a set of samples specified by the user with the **--samples** OPTION. The comparative analysis can be run in different detection modes using the OPTIONS **--snp_vars**, and **--lowfreq_vars** (with the respective changes to the detection algorithm explained in detail in the description for these OPTIONS). For the joint analysis, first a scaffold of all variant positions is built from the individual variant files. Second, for all positions with a variant detected for any of the input samples, the allele information is recalculated from the original mappings to produce a comprehensive table. Output files will be created starting with the project name specified with the **--project** OPTION, otherwise the default file name will start with “NONE”.

The first line within the tabular delimited file joint variant table is a sample header line. The second line starts with position specific columns and continuing with sample specific entries:
Position specific columns:

#Position	Position of the reference genome
Insindex	An index for reporting insertion sites, with 0 indicating no insertion
Ref	The allele present in the reference genome at this position
Gene	Gene ID as given in the reference genome annotation
GeneName	Gene name as given in the reference genome annotation
Annotation	Gene product as given in the reference genome annotation

Sample specific fields:

Type	Type of the detected variant.
Allele	Allele detected for the respective sample at this position
CovFor	Number of forward mapped reads indicating the detected allele
CovRev	Number of reverse mapped reads indicating the detected allele
Qual20	Number of reads indicating the detected allele with a phred score of at least 20
Freq	Frequency of the detected allele
Cov	Total coverage depth at this position
Subst	Information about a possible substitution

Input:
samples.txt
--project
Called/[SampleID]_[LibID]_[*].gatk_position_variants_[mincovf]_[mincovr]_[minfreq]_[minphred20]_[all_vars][snp_vars][lowfreq_vars].tab
Position_Tables/[SampleID]_[LibID]_[*].gatk_position_table.tab
Output
Joint/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples.tab
Joint/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples.log

TBamend

VALUE of **--step** for post-processing of joint variant tables. If you call this step directly, you need to specify the correct **--project** OPTION**.** This step will produce a comprehensive variant table including calculated summary values for each position. In addition, the set of positions will be processed in consecutive filtering steps.

In the first step, positions will be excluded if less than a minimum percentage of samples have unambiguous base calls at this position, with this threshold set by the **--unambig** OPTION. In addition, all samples need to have either a SNP or wild type base at the position, and positions within repetitive regions of the reference genome or within a resistance associated genes are excluded. This filtering step results in output files carrying the "_amended_[unambig]_phylo" ending; a full table (ending in .tab), a FASTA file containing the aligned alleles of all samples for the given position (.fasta), and a corresponding FASTA file with the headers consisting solely of the respective sample ID (_plainIDs.fasta).

The second filtering step removes positions that are located within a maximum distance to each other in the same sample, with this threshold set by the **--window** OPTION. Output files from this filtering step are generated with the same naming scheme as for the first step and carry the selected window threshold as additional field. In addition, positions not passing the window criteria are reported in the output file carrying the tag "removed".

Input
samples.txt
--project
Joint/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples.tab
Output
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended.tab
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo.tab
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo.fasta
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_plainIDs.fasta
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_[window].tab
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_[window].fasta
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_[window]_plainIDs.fasta
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_[window]_removed.tab

TBgroups

VALUE of **--step** for inferring likely related isolates based on pairwise distance of distinct SNP positions. In an agglomerative process, samples are grouped together if they are within the threshold set by the **--distance** OPTION. The output files consist of a text file listing the detected groups and ungrouped isolates, and the calculated pairwise distance matrix.

Input:
Amend/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_[window].tab
Output:
Groups/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_[window].matrix
Groups/[PROJECT]_joint_[mincovf]_[mincovr]_[minfreq]_[minphred20]_samples_amended_[unambig]_phylo_[window]_[distance].groups

--continue

If a module was chosen with the **--step** OPTION, the **--continue** OPTION ensures that the pipeline will continue the analysis with downstream modules. This is automatically set if the **--step** OPTION is set to the VALUE **TBfull**.

--samples

This OPTION requires a user supplied file specifying a set of datasets (e.g. samples.txt) as VALUE. The file must be a two-column, tab-separated file. Column 1 has to be your **[SampleID]**. Column 2 has to be your **[LibID]**. **TBjoin** requires this OPTION to be set.

--project

This OPTION allows the user to set a project name for the steps **TBjoin**, **TBamend** and **TBgroups**. If you do not support a project name, **[NONE]** is used as a default value.

--ref

This OPTION sets the reference genome for the read mapping. By default, the genome of *Mycobacterium tuberculosis* H37Rv (NC_000962.3) is set as reference. User supplied FASTA files for other reference genomes should be placed in the directory /MTBseq_source/var/ref/, and the respective name given without .fasta extension. Please be aware that for other reference genomes, you need to provide the respective annotation files as well or annotations will be skipped.

--resilist

This OPTION sets a list of known variant positions associated to drug resistance for resistance prediction. Give the full path to the file. The required structure of the file can be seen here:
/MTBseq_source/var/res/MTB_Resistance_Mediating.txt

--intregions

This OPTION sets a list of interesting regions to be used for annotation of detected variants. Give the full path to the file. The required structure of the file can be seen here:
/MTBseq_source/var/res/MTB_Extended_Resistance_Mediating.txt

--categories

This OPTION specifies a gene categories file to annotate essential and non-essential genes as well as repetitive regions. SNPs in repetitive regions will be excluded for phylogenetic analysis. Give the full path to the file. The required structure of the file can be seen here:
/MTBseq_source/var/cat/MTB_Gene_Categories.txt

--basecalib

This OPTION specifies a file for base quality recalibration. The list must be in VCF format and should contain known SNPs. Give the full path to the file. The required structure of the file can be seen here:
/MTBseq_source/var/res/MTB_Base_Calibration_List.vcf

--all_vars

This OPTION is used in the modules **TBvariants**, **TBstats**, **TBjoin**, and **TBstrains**. By default, the OPTION is not active. Setting this OPTION will skip all filtering steps and report the calculated information for all positions in the input file.

--snp_vars

This OPTION is used in **TBvariants**, **TBstats**, **TBjoin**, and **TBstrains**. By default, the OPTION is not active. Setting this OPTION will add an additional filter that excludes all variants except SNPs.

--lowfreq_vars

This OPTION is used in **TBvariants**, **TBstats**, **TBjoin**, and **TBstrains**. By default, the OPTION is not active. Setting this OPTION has major implications on how the mapping data for each position is processed. By default, the majority allele is called and taken for further calculations. If the **--lowfreq_vars** OPTION is set, MTBseq will consider the majority allele distinct from wild type, if such an allele is present. This means that only in this detection mode, MTBseq will report variants present only in subpopulations, i.e. low frequency mutations. Of course, OPTIONS **--mincovf**, **--mincovr**, **--minphred20**, and **--minfreq** need to be set accordingly. Please be aware that output generated in this detection mode should not be used for phylogenetic analysis.

--minbqual

This OPTION is used in **TBlist**. By default, the OPTION is set to 13. The OPTION sets a threshold for the sequence data quality to be used for the mpileup creation.

--mincovf

This OPTION is used in **TBvariants**, **TBjoin**, **TBamend**, and **TBstrains**. By default, the OPTION is set to 4. The OPTION sets a minimum forward read coverage threshold. Alleles must have a forward coverage of this VALUE or higher to be considered.

--mincovr

This OPTION is used in **TBvariants**, **TBjoin**, **TBamend**, and **TBstrains**. By default, the OPTION is set to 4. The OPTION sets a minimum reverse read coverage threshold. Alleles must have a reverse coverage of this VALUE or higher to be considered.

--minphred20

This OPTION is used in **TBvariants**, **TBjoin**, **TBamend**, and **TBstrains**. By default, the OPTION is set to 4. The OPTION sets a minimum number of reads indicating an allele with a phred score of at least 20.

--minfreq

This OPTION is used in **TBvariants**, **TBjoin**, **TBamend**, and **TBstrains**. By default, the OPTION is set to 75. The OPTION sets a minimum frequency for an allele.

--unambig

This OPTION is used in **TBamend**. By default, the OPTION is set to 95. The option sets a minimum percentage of samples with unambiguous information for position.

--window

This OPTION is used in **TBamend**. By default, the OPTION is set to 12. The OPTION sets a window size in which the algorithm scans for the occurrence of multiple variants within the same sample. If more than one variant occurs within this window in the same sample, the positions will be excluded.

--distance

This OPTION is used in **TBgroups**. By default, the OPTION is set to 12. The OPTION sets a SNP distance that is used to classify samples into groups of samples, using agglomerative clustering. If SNP distances between samples are less or equal this VALUE, they are grouped together.

--quiet

This OPTION turns off the display logging function and will report the logging only in a file, called "MTBseq_[DATE]_[USER].log".

--threads

This OPTION is used in **TBbwa**, **TBmerge**, **TBrefine**, **TBpile** and **TBlist**. By default, the OPTION is set to 1. The OPTION sets the maximum number of CPUs to use within the pipeline. You can use more than one core in order to execute the pipeline faster. 8 is the current maximum.

--help

This OPTION will show you all available OPTIONs and corresponding VALUEs used by MTBseq.

--version

This OPTION will show you the current version of MTBseq.

[EXCHANGING THE REFERENCE GENOME]

Apart from the supplied *Mycobacterium tuberculosis* H37Rv (NC_000962.3) reference genome, any other bacterial genome can be used for MTBseq if the necessary files are provided. First, the FASTA formatted genome sequence has to be placed in the directory /MTBseq_source/var/ref/ and **–ref** has to be set during program execution to the respective file name but omitting the .fasta extension. In order for MTBseq to provide gene annotations, a respective annotation file with the extension _genes.txt needs to be placed in the same directory. For file formatting, follow the example of the existing annotation files, e.g. in the M._tuberculosis_H37Rv_2015-11-13_genes.txt file. In addition to *Mycobacterium tuberculosis* H37Rv, three other reference genomes are provided (*Mycobacterium abscessus, Mycobacterium chimaera and Mycobacterium fortuitum*).
Variant annotations regarding association to resistance, interesting regions or gene categories are only performed if the respective annotation files are provided to MTBseq with the OPTIONs **–resilist, --intregions** or **–categories**. See these OPTIONs for details and the required structure of the files.

Base calibration of the **TBrefine** module is only done for a modified reference if a list of known SNP positions is provided through the **–basecalib** OPTION, otherwise it will be skipped. This will not affect the realignment functionality of the **TBrefine** module.

[CUSTOMIZING THE CODE]

External programs used in MTBseq are implemented as SYSTEM calls and are used in **TBbwa** and **TBrefine**. To get access to the full range of parameters of included programs, the respective SYSTEM call can be modified with the desired parameters as long as the output remains in the same format. Be absolutely sure to understand the options and their impact, i.e. consult the documentation of the respective program first. This can go as far as exchanging individual programs such as the tool used for read alignment.

MTBseq is built on a modular structure, thus it is possible to write your own modules in addition to or in replacement of existing modules. Programming knowledge is required. The currently available modules can be found in the /MTBseq_source/lib/ directory and can serve as templates. The following steps will give a short introduction to build your own module:

1. Define your module and create YOURmodule.pm in MTBseq_source/lib/
2. Define the module's environment in MTBseq_source/MTBseq.pl. See code comments for the respective sections
    a. Input/Output directories ($W_dir)
    b. Program directories ($RealBin/opt/)
    c. Implement the module (use YOURmodule)
    d. Define options and parameters for the module
    e. Make the module part of the pipeline (--step)
    f. Define working variables for checkups (@ARRAYS, %HASHES)
    g. Define the module call logic (Input files, checks, hand over options, and directories)
3. Write the module
    a. Implement necessary modules
    b. Define the main function of the module and export it (@EXPORT = qw(yourmodule))
    c. Implement the main function (with the parameters handed over from MTBseq.pl)

[EXAMPLES]

MTBseq --step TBfull
Will execute the whole pipeline with default values. All log output is written into a file called "MTBseq_[DATE]_[USER].log" and on screen. The log file is located in the working environment, i.e. the folder in which you executed the pipeline.

MTBseq --step TBbwa --continue --threads 8
The pipeline starts with the read mapping module and continues after finishing this module. The system will use 8 cores whenever possible. All log output is written into a file called "MTBseq_[DATE]_[USER].log" and on screen.

MTBseq --step TBlist --threads 8
This example executes the **TBlist** module with 8 threads. All log output is written into a file called "MTBseq_[DATE]_[USER].log" and on screen. To execute this module, you need to have the respective output from the previous module **TBpile** available in the working environment.

MTBseq --step TBjoin --sample samples.txt --project TEST
This example uses the **TBjoin** module with default settings. All log output is written into a file called "MTBseq_[DATE]_[USER].log" and on screen. To execute this module, you need to have the respective output from the previous modules **TBlist** and **TBvariants** available in the working environment.

MTBseq --step TBstrains
This example uses **TBstrains** with default settings. To execute this module, you need to have the respective output from the previous module **TBlist** available in the working environment. All log output is written into a file called "MTBseq_[DATE]_[USER].log" and on screen.

MTBseq --step TBvariants --mincovf 10 --mincovr 10 --minfreq 80 --minphred20 10
This example uses **TBvariants** with a modified settings for variant calling. All log output is written into a file called "MTBseq_[DATE]_[USER].log" and on screen. To execute this module, you need to have the respective output from the previous module **TBlist** available in the working environment.

[AUTHORS]

Thomas A. Kohl (source code, core logic)
Robin Koch (source code, package building)
Christian Utpatel (source code, core logic)
Maria R. De Filippo (source code, beta test)
Viola Schleusener (performance comparison, beta test)
Patrick Beckert (annotation data, beta test)
Daniela M. Cirillo (Head)
Stefan Niemann (Head)

[COPYRIGHT AND LICENSE]

[HOMEPAGE AND SOURCE REPOSITORY]

MTBseq on github: https://github.com/ngs-fzb/MTBseq_source/
Research center Borstel: http://www.fz-borstel.de/cms/en/science/start.html

[REFERENCES]

1. Homolka, S. et al. High resolution discrimination of clinical Mycobacterium tuberculosis complex strains based on single nucleotide polymorphisms. PLoS One 7, e39855 (2012).

2. Coll, F. et al. A robust SNP barcode for typing Mycobacterium tuberculosis complex strains. Nature Communications 5, 4812 (2014).

3. Merker, M. et al. Evolutionary history and global spread of the Mycobacterium tuberculosis Beijing lineage. Nature Genetics 47(3), 242-249 (2015).