PASS v 2.23 user manual

Contents

Introduction

PASS applies an innovative strategy to perform fast gapped and ungapped alignment of reads onto a reference sequence. It supports several data formats and allows the user to modulate very finely the sensitivity of the alignment. The program is designed to handle huge amounts of short reads generated by Illumina, SOLiD and Roche-454 technology. The optimization of the internal data structure and a filter based on precomputed short-word alignments allow the program to skip false positives in the extension phase, thus reducing the execution time without loss of sensitivity. The final alignment is performed by dynamic programming. Pass can be used for several applicaions such as single read mapping, paired-end resequensing, small RNA discovery and RNA-seq mapping.

The main features of PASS

(1) Fast execution time
(2) Many options to modulate the sensitivity of the program
(3) Full compatibility for SOLEXA, 454 and SOLiD sequencing technologies
(4) Bases Transformations and new scoring system for color space
(5) Multi threading support
(6) Long reads alignment support
(7) Local alignment function support
(8) Paired-end managing support
(9) Spliced reads alignment support
(10) Function for cleaning reads, also for SOLiD data
(11) Bisulfite sequencing mapping support
(12) SNP calling

PASS 2.10 has been considerably improved compared with the previous versions.

(1) **New** - Memory requirement reduced by 50%
(2) **New** - 30% mapping speed increasing
(3) **New** - gaps and local alignments for Bs-seq
(4) **New** - output that includes methylation level
(5) **New** - indexing and improvements for pairing process
(6) Minor bugs fixed

These are some of the main changes and improvements of PASS 2.02:

(1) The way to define the input file with the reads has been changed; the arguments -fasta, -fastq, -csfastq, -csfasta, followed by the file names makes simpler and intuitive the definition of the input file in different formats.
(2) Automatic read trimming based on analysis of base quality and machine-learning steps.
(3) SAM format supported
(4) Automatic setting of the main parameters in several conditions (based on machine-learning)
(5) Reads cleaning for adaptors and contaminants (also for SOLiD)
(6) Bisulfite sequencing support also for SOLiD sequencing
(7) Specific allele association to Cs methylation

1 Installation

1.1 *Linux system configuration*

The Kernel should be 2.4 or later. The glibc should be 2.3.2w/NPTL or later. The CPU should be 32 or 64 bits architecture (best performance with multi-core processors). The RAM should be 2-16 GB, depending on the genome and on the parameters used. Operating systems like the Ubuntu 5.10, Suse 10.1, Fedora core 5, Linspire 5.1, Gentoo2006.0, Debian 3.1, and Red Hat 9 should be suitable for the installation of PASS without the need of any further requirement.

1.2 Required libraries

libpthread.so
libstdc++.so
libm.so
libgcc_s.so
libc.so
ld-linux.so

1.3 Compilation

The pass package comes a pre-compiled version of the program, ready to be used. However, if the user wants to re-compile it, the following simple instructions can be followed.

(1) decompress pass_vx.xx.tar.gz archive using the command: "tar -xvzf pass.tar.gz"
(2) enter into the pass directory: "cd pass"

There are several files and directories type "make" to compile the source code and see the README file for further information.

2 The algorithm

PASS builds a database (index) of the seed words occurring in the reference sequences. Then, for each seed word of the query sequence it performs 3 processes:

- Scanning the list of positions of he seed words in the genome
- Checking the flanking regions
- Refining the extension

2.1 Database building

A PASS database is constituted by a list of positions of seed words occurring in the reference sequence. More precisely, there are many lists, one for each seed word. The total number of lists is therefore 4^w where w is the length of the word and each list contains all the genomic positions where the word occurs.

In order to speed up the mapping procedure, pass is also using a pre-computed score table (PST) containing the results of all the possible combinations of short words aligned against each others. If we

call s the size of long assignment (could be 4 or 8 and depends by CPU type),*w* the length of seed words, *k* the length of the PST words and L the length of the reference genome, then the required memory M will be:

$$M = [\ (4^{w} * s) + (4^{w} * 4) + (4 * L) + L + (4^{k*2})\ ] + [\ (L / 8) * t\ ]$$

Typically, the size of the seed word w is 12 or 14, while the size of the PST words is between 6 and 8. The member of the equation within the first set of square brackets represents the amount of memory required for storing the indexed database, the reference sequence and the PST. The last part of the equation represents the memory required for threads management, where *t* is the number of threads.

For bisulfite mapping the required memory M can be calculated as follow:

$$M = (4^{w} * s) + (4^{w} * 4) + (5 * 2L) + 2L + (4^{k*2}) + (2L / 8) * t$$

For instance, the bisulfite mapping of the Human genome requires about 30 Gb of RAM if we use 12 processor cores and, "-p" parameter set to "11111111111111" .

2.2 Scanning seed list positions

In this step the program converts the query sequences to numeric data type and scans the database list for each seed words matching to the reference sequence.

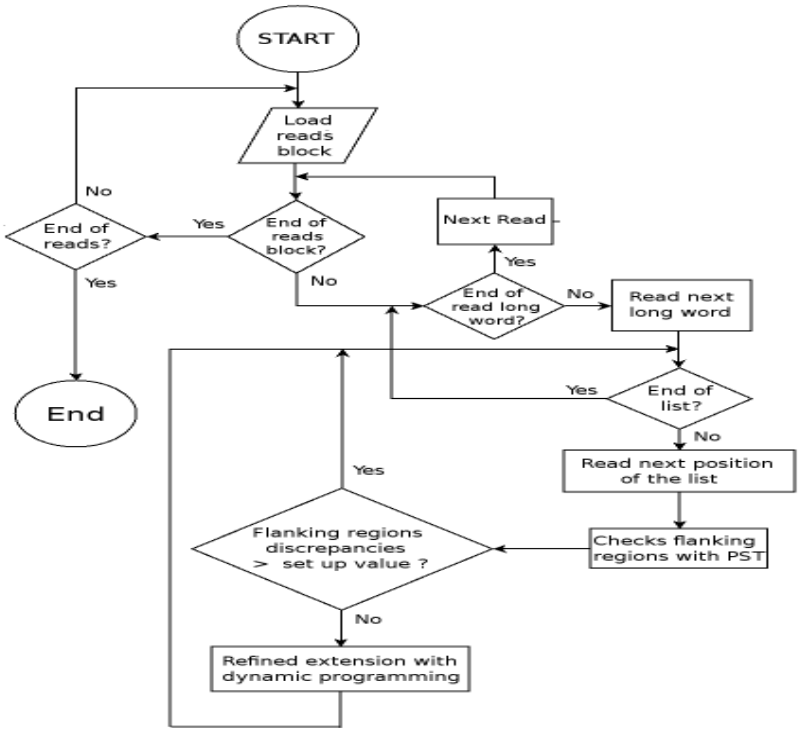


Fig. 2: Pass optimization of the extension phase. The refined extension requires longer time than the other steps. The efficiency of the program is also due to its ability to identify and skip false positives .

2.3 Flanking regions checking

The PST (precomputed score table) have been obtained by an implementation of the Needelman-Wunch algorithm, that has been modified to calculate the scores of the alignment with any combination of short words of DNA of predefined length (typically 6 or 7 bases). For every seed word matching, the program checks the flanking regions of query and reference using the PST. Only if the resulting score is higher than a given threshold then pass will enter into the refined extension procedure, otherwise it will proceed with the next seed word (see fig.2).

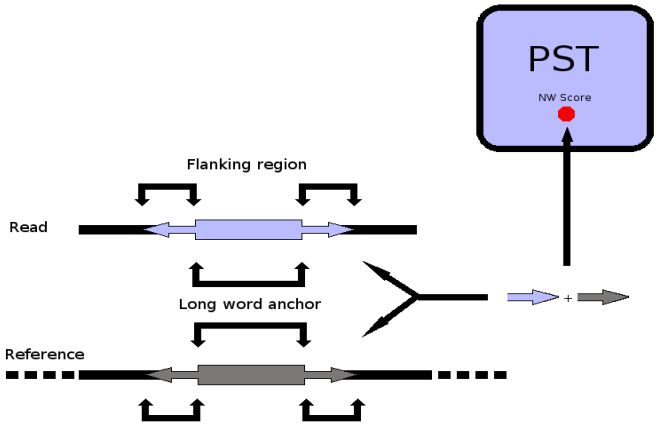


Fig. 3: this figure show how pass execute a fast check of the alignments using precomputed score matrix. The flanking regions adjacent to the long word anchor are coded to calculate the location of the Needelman-Wunch score in the PST matrix.

2.4 Refined extension

Compared to the earlier version of pass, this step is been improved to increase the speed. The algorithm is similar to Smith-Waterman but it has modified in order to reduce score matrix alignment and managing fewer gaps.

3 Starting

PASS program must be set for those indispensable parameters:

(1) **Spaced seed word.** -p parameter define at the same time structure and length of the spaced seed word
(2) **Input reads .** -fasta, -fastq, (SOLiD) -csfasta -qual or -csfastq parameters allows to load reads files
(3) **Input reference sequences.** -d parameter allows genome file loading
(4) **% base identity.** (-fid parameter)
(5) **minimal size of alignments.** -fle parameter allow to filter alignment < of set value.
(5) **gap numbers.** defined with -g parameter.
(6) **Best hits.** -b parameter enable this option.
(7) **Special filters.** -flc (linguistic complexity filter for seed word)

PASS direct to STDOUT the result alignments and to STDERR all information performed during execution. In order to save this data, '>' and '2>' could be used to redirect output; otherwise use -o parameter to save results file.

Understanding the main options

-fid n -fid option is intended as an identity % only if you set the gap panalty = 1 (default). In all cases -fid option reflects the threshold score of each alignment calculated as follow:

(fid% * alignment_length) / 100

-g n Option -g enable gaps alignment mode and determines the size (x,y) of the score matrix used to align the sequences: ((n*2+1), read_length).

PASS program maps short sequences if these two condition are true:
(a) |insertions-gaps| <= n.
(b) max gap or max insertion <= 2 * n

Those conditions will allow to recognize a number of consecutive (gaps + insertions) > n.

4 Mapping speed

The mapping speed depends by setting. A good setting allows the better compromise between sensitivity and specificity of outputs alignments. Those are some suggestions to optimize both efficiency and speed of mapping.

4.1 Structure of the seed pattern

In order to increase the speed of the mapping you could change the structure of the seed pattern:

Illumina:
-p 1111110111111 default
-p 11111110111111 increase speed about 2x
-p 111111101111111 increase speed about 3x
-p 1111111101111111 increase spedd about 4x

SOLiD:
-p 11111100111111 default
-p 111111100111111 increase speed about 2x
-p 1111111001111111 increase speed about 3x
-p 11111111001111111 increase spedd about 4x

Especially for reads size > 50 bases we recommend to increase the size of the seed pattern, however this action could affects the amount of required memory.

4.2 Number of saved seeds in the indexing process

Using the parameter -seeds_step, during the indexing process, PASS will consider only the seeds located at the seed step distance. Consequently, "-seeds_step 3" will increase the mapping speed of 3 time and will reduce the required memory of about 1/3.

4.3 Linguistic complexity of the seed

The -flc parameter allows to filter out the low complexity seeds.

-flc 1 (it filters only homopolymers) default
-flc 2 (it filters seeds that have almost 2 equal hexamers)
-flc 3 (it filters seeds that have almost 3 equal hexamers)
-flc 4 (it filters seeds that have almost 4 equal hexamers)
so -flc 2 is the stronger filter for sequence low complexity.

The best setting is a combination of those 3 parameters and should be set considering the desired sensitivity and the type of mappable data.

5 The base setting

The following examples should help the users to understand the best parameters and the commands should be issued from PASS directory.

5.0 Saving the indexed reference database

Do you need to align several input reads as small separated files? you can save time creating the indexed database of the reference genome. The binary file will be generated using the parameter "-D filename.bin" but it is important to add "-solidCS" and "-bisulfite" parameters in the case of SOLiD reads and / or bisulfite sequencing.
In order to load the generated database you should use the parameter "-R filename.bin".
Also in this case, if necessary don't remember to use the mentioned -bisulfite parameter. Follow some practical examples:

Typical setting for ILLUMINA RNA-seq and DNA-seq reads:

Generates database:

```
bin/pass  -p 111111101111111  \
       -cpu 12 -flc 1  -d genome.fa -seed_step 3  \
       -D genome.bin
```

Reads database and maps:

```
bin/pass  -R genome.bin   \
      -fastq reads.fastq \
       -cpu 12 -flc 1  -seed_step 3  \
       -fid 90 -b -fle 40 -sam > results.sam
```

Typical setting for SOLiD RNA-seq and DNA-seq reads:

Generates database:

```
bin/pass  -p 1111111001111111  \
       -cpu 12 -flc 1  -d genome.fa -seed_step 3  \
       -solidCS -D genome.bin
```

Reads database and maps:

```
bin/pass  -R genome.bin   \
      -csfastq reads.csfastq \
       -cpu 12 -flc 1  -seed_step 3  \
       -fid 90 -b -fle 40 -sam > results.sam
```

5.1 ILLUMINA global mapping setting

Source data: base space fastq format

- raw data as fastq format reads.fastq
- reference genome.fa
- quality data type phred64 (set with -phred64)
- percent identities: 90%
- cores 12
- omopolymer seed filter set (-flc 1)
- 14 bases seed pattern: 111111101111111
- best hit set with "-b"
- results sam format

Command:

```
bin/pass  -p 111111101111111 -b \
      -cpu 12 -flc 1 -fid 90 -sam -phred64 \
      -fastq example/fastq/reads.fastq \
      -d example/genomes/genome.fa -seed_step 3 \
      > result.sam
```

The results will be saved in the file result.sam

5*.2 SOLiD global mapping setting*

Source data: base space fastq format

- raw data as csfasta and qual format (reads.csfasta , reads.qual)
- reference file as fasta file genome.fa
- percent identities: 90%
- cores 12
- omopolymer seed filter set (-flc 1)
- 14 bases seed pattern: 1111111001111111
- best hit set with "-b"
- results sam format

Command:

```
bin/pass -p 1111111001111111 -b \
      -cpu 12 -flc 1 -fid 90 -sam \
      -csfasta reads.csfasta \
      -qual reads.qual \
      -d genome.fa -seed_step 3 \
      > result.sam
```

The results will be saved in the file result.sam . PASS supports also the csfastq format of the reads file, similar to the not supported BFAST format.

5*.3 L*ocal alignments

The local alignment function is important under certain particular situation. If you want to map the long reads coming from a RNA-seq experiment using the global alignment function you will worst a lot of alignments because the presence of introns. In order to enable PASS local alignment function you should add "-l" to the command line and then set a proper threshold to filter small alignments with a score less than a set threshold (set "-fle" parameter to a desired value).

For instance adding the paramenters "-l -fle 30" to the command line of a standard global alignment

setting, you will enable the local alignment function. As resulting of this changing, each read will produce several small alignments that will be filtered basing on their score value. It is important to evidence that reads coming from SOLiD and ILLUMINA platforms have the same setting and will be threated in the same way.

5.4 Spliced alignments

The spliced alignment mapping, specifically designed for RNA-seq data, allows to map the reads onto a reference genome on the splicing site junctions. The spliced alignments is particular useful to understand the transcript variants, but also to understand gene structures of new assembled genomes.

If you are mapping the reads on a **small reference genome** the base setting could be:

Command:

```
bin/pass -cpu 8 -p 1111111111 -b \
-cpu 12 -flc 1 -fid 90 -sam -phred64 \
-fastq example/fastq/reads.fastq \
-d example/genomes/genome.fa \
-spliced rna -percent_tolerance 30 -fle 10 \
> result.sam
```

you should notice the un-gapped structure of the seed pattern. The sensitivity and the specificity are affected by several parameters and you should set them according to the size of the reads. This is a brief description of the main parameters for spliced mapping:

-i_score n	to filter some class of introns basing on the conserved bases on spliced junction
-e n	e-value for spliced alignements.
-max_distance n	the tolerated max intron size
-fle n	length of the minimal alignment size
-p 11111111	size of the seed pattern structure
-percent_tolerance n	the tolerated percent of read size not covered by the alignment

5.5 Spliced alignment: additional strategy

In cases of long reference genome as the Human, the first setting is not suitable because it requires too much time. We propose another strategy to optimize spliced mapping that requires 3 steps that could be managed as the following pipeline:

(1) A step for mapping the not spliced reads to the exons. In this step you will produce global alignments and, at the same time, you will recover the not aligned reads using the "-not_aligned" parameter. This is an example:

```
bin/pass -cpu 8 -p 111111101111111 -b \
      -cpu 12 -flc 1 -fid 90 -sam -phred64 \
      -fastq example/fastq/reads.fastq \
      -d example/genomes/genome.fa \
      -not_aligned -na_file not_aligned.fastq \
      > global_result.sam
```

(2) You should generate the mapping coordinates of the "not aligned reads" using the local alignment mapping.

```
bin/pass -cpu 8 -p 1111111111 -b \
      -cpu 12 -flc 1 -fid 90 -sam -phred64 \
      -fastq not_aligned.fastq \
```

```
    -d example/genomes/genome.fa \
    -l -fle 10 -focus_check focus.fastq \
    > /dev/null
```

you can notice the parameter "-focus_check focus.fastq" that allows to save the coordinates of the mapped sequences as part of the read name. This information will be saved to "focus.fastq" file.

(3) Focusing the spliced alignments at the found coordinates.

```
bin/pass -cpu 8 -p 1111111111 -b \
    -cpu 12 -flc 1 -fid 90 -sam -phred64 \
    -fastq focus.fastq \
    -d example/genomes/genome.fa \
    -spliced rna -percent_tolerance 30 -fle 10 \
    -not_aligned \
    > spliced_result.sam
```

The 3 steps could be reiterated in a loop to increase the sensitivity step by step. However, you should remove the redundancy from the not aligned reads generated by multi map reads (by step 3) and then you should be able to restart a new loop from the step 2). For each step you should change "-p" and "-fle" parameter in order to increase the sensitivity and the specificity of the mapping.

The setting of the SOLiD data is the same as other sequencing platforms but, the "-p" parameter and the input format of the reads must be properly set. If you are interested on the pipeline please contact the authors at the email address: pass@cribi.unipd.it. The next release of PASS package will include our testing pipeline.

5.6 Reads cleaning

There are several reasons for reads cleaning. For instance, in the sequencing of miRNA the adaptors could be systematically sequenced because the long read size. PASS allows to recognize and remove the adaptors from each read using the local alignment function. It maps the reads onto a reference adaptors (specifically designed for each sequencing platform) and under certain mapping conditions, it executes the cleaning. If the conditions are not verified the reads will be discarded.

This is an example of SOLiD RNA-seq data cleaning. The same strategy could be used also for ILLUMINA data, but you need to change the adaptors sequences used as reference data for the specified sequencing platform. The following parameters could be optimized: percent identity (-fid), length of the alignments (-fle) and size of the seed pattern (-p).

```
bin./pass \
    -csfasta file.csfasta \
    -qual file.qual \
    -p 11111111 -int_limit 4 -ext_limit 3 \
    -check_block 1000 \
    -cpu 12 \
    -block 1000000 \
    -d barcoded_adaptors.fa \
    -l -fle 12 -fid 90 -sam -b -no_pst_auto -flc 0 \
    -cleaned_3end 20 -cleaned_all > cleaned.csfastq 2>cleaned.log
```

You can notice the parameter "-cleaned_3end" set to 20. Under this set all cleaned reads that have a size < 20 bp will be discarded and the good reads will be output to the STDOUT (always fastq format). In this example the input reads are in csfasta/qual format but you can change properly another input

format.

You need to create the "barcoded_adaptors.fa" file that contains the suitable adaptors for the SOLiD 5500XL sequencer (barcoded RNA-seq runs only).

>P1
CCACTACGCCTCCGCTTTCCTCTCTATGGGCAGTCGGTGAT
>IA
CTGCTGTACGGCCAAGGCG

5.7 Bisulfite sequencing

Bisulfite sequencing allows to determine the methylation pattern of the DNA. Treatment of DNA with bisulfite converts cytosine residues to uracil, but leaves 5-methylcytosine residues unaffected. Thus, bisulfite treatment introduces specific changes in the DNA sequence that depend on the methylation status of individual cytosine residues, yielding single-nucleotide resolution information about the methylation status of a segment of DNA.

A detailed description of the method implemented in PASS has been submitted for publication.
A test data set, which consists of 1 million of simulated color-space reads to be map onto the Human genome could be downloaded at: "http://pass.cribi.unipd.it" referring to the file "programs_comparison_test_set.tar.gz". Inside the compressed file there are a step by step setting information that constitutes an integral part of the implemented method submitted for publication.

In order to efficiently map bisulfite threated reads, PASS uses two steps based on different strategies: the first (3-base conversion) is efficient in terms of accuracy and resources, but does not perform well with poor quality reads. The second strategy (combinatorial C/T substitution) is more demanding in terms of resources, but allows to re-analyze the reads that did not align in the first step, and to have a higher yield of mapped reads.

1) The first strategy is very fast. The reads and the genome are converted to three-base space by changing all the Cs into Ts. The reads are then mapped and the methylation status can be inferred by comparing the alignments with the original 4-bases sequences.

2) The second strategy regards only the color space reads, that have sequencing errors at the 5'-end. The sequencing error at the 5'-end don't allow a correct color space conversion to base space so, the reads will not map with the first strategy. To solve this problem PASS considers a great number of seed combinations in the indexing step, where Cs could be methylated or not methylated. As a result a large fraction of the reads that did not map in the first step can be recovered.

Base space mapping
The base-space pipeline is based only on the first strategy because there is not the problem of color to base-space conversion. The following parameters can be used for base-space reads:

First strategy:
```
bin/pass -p 11111111111111 -d  GENOME/hg19.fasta \
      -fastq filename .fastq \
      -flc 4 -sam  \
      -seeds_step 3 -check_block 1000 \
      -bisulfite -fid 90 -b -cpu 12 \
      >REAL_TEST_SET/ filename.sam   2>REAL_TEST_SET/filename.log
```

Color space mapping

The color-space mapping can either be done using only the first strategy, or using both strategies. To use both strategies the program must be run twice, the first time on the full set of reads and the second time on the reads that did not align on the first step. These are the parameters that could be used:

First strategy (ILLUMINA):

```
bin/pass -p 11111111111111 -d  GENOME/hg19.fasta \
      -csfastq filename.csfastq \
      -flc 4 -sam  \
      -seeds_step 8 -check_block 10000 \
      -bisulfite -fid 90 -b -cpu 12 \
      >filename.sam   2>filename.log
```

First strategy (SOLiD):

```
bin/pass -p 11111111111111 -d  GENOME/hg19.fasta \
      -csfastq filename.csfastq \
      -flc 4 -sam  \
      -seeds_step 3 -check_block 1000 \
      -bisulfite -fid 90 -b -cpu 12 \
      -original -not_aligned -na_file filename.NA.csfastq \
      >filename.sam   2>filename.log
```

Second strategy (SOLiD):

```
bin/pass -p 11111111111111 -d  GENOME/hg19.fasta \
      -csfastq filename.NA.csfastq \
      -flc 4 -sam  \
      -seeds_step 3 -check_block 1000 \
      -bisulfite -fid 90 -b -cpu 12 \
      -max_combinations 8 \
      >filename.NA.sam   2>filename.NA.log
```

The line starting with “-original” is used to save the not-aligned reads to be used with the second strategy, therefore it can be omitted if the quality of the reads is good and the second strategy is not required.

Memory requirement

The required memory M can be calculated as follow:

$$M = (4^w * s) + (4^w * 4) + (5 * 2L) + 2L + (4^{k*2}) + (2L / 8) * t$$

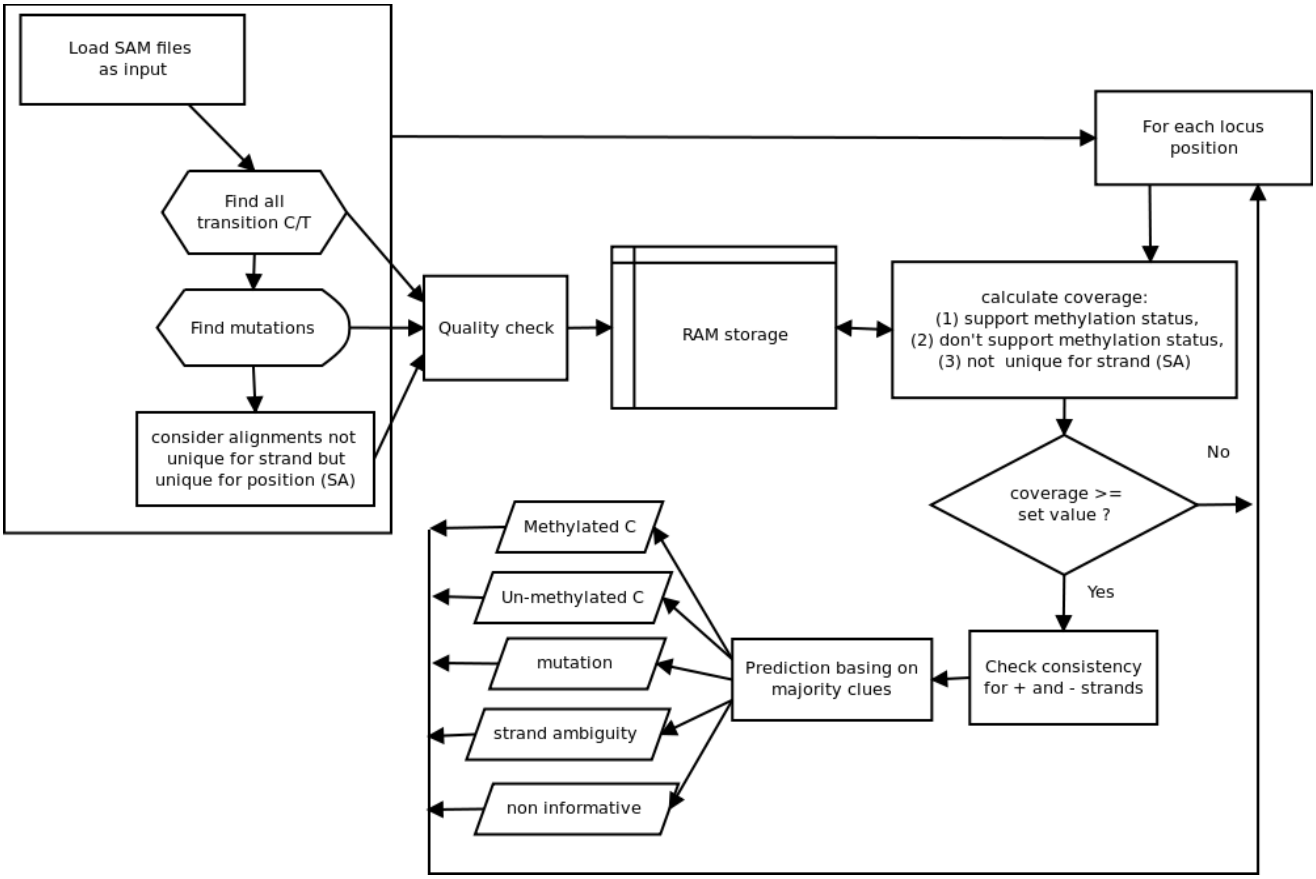
s represents the size of long assignment (could be 4 or 8 and depends by CPU type), w the size of the seed word defined by "-p" parameter, L is the length of the reference sequence, k is the size of the short word (see PST) and t the number of used threads.

For instance, the bisulfite mapping of the Human genome requires about 30 Gb of RAM if we use 12 processor cores and, "-p" parameter set to "11111111111111" .

5.7.1 Methylation calling

The methylation calling program uses the SAM files produced by bisulfite mapping to generate a list of

methylated and un-methylated cytosines. The implemented algorithm is able to manage both unique alignments and the alignments unique for position but not for strand (typically mapped into regions where Cs are totally methylated if libraries are not directed) . This is the schematic flow chart of the program.



Main setting for methylation calling:

```
bin/pass \
    -program genotype \
    -sam SAM/CS-strategy1-A.sam \
    -sam SAM/CS-strategy2-A.sam \
    -sam SAM/CS-strategy1-B.sam \
    -sam SAM/CS-strategy2-B.sam \
    -d GENOME/1M_hg19.fasta \
    -bisulfite -p 0.001  -q 9 -hits 2 -flank 0 \
    > RESULTS/results.list 2>LOG/results.log
```

"-program genotype" enable the base calling program
"-bisulfite" enable the bisulfite analysis
“-allele” It works only with “-bisulfite” option and it requires a file which contains the polymorphic position of the genome. This option enables the allele association with specific methylation. The input file must contains the information about polymorphisms with the following format:

Chromosome name<tab>chromosome position

For instance this is an example:
chr10 10010
chr10 10200
....

"-sam file" set the input file of the alignments in SAM format
"-q 9" consider only bases with quality >= 9
"-hits 2" consider reads that produce a maximum of two alignments. This set is indispensable to analyze the alignments that are unique. for position but not for strand. If you have directed libraries set this parameter to 1.
"-flank 0" Don't trim the alignment at the 3' end.
"-p 0.01" Consider the 99.99 % of the results basing on coverage distribution. This set implies to consider a minimal coverage auto-calculated.

For particular purpose you can consider to select only the alignments that map onto a single strand. For instance some bs-seq kits are strand specific. The parameter "-bis-strand" should be set as follow:

-bis-strand This is a strand filter for bisulfite alignments. Default [0].
0 it considers all strands.
Set to 1 for strands ++ and +-.
Set to 2 for strands -- and -+

Output format:
(1) Reference Chromosome
(2) reference position
(3) reference base
(4) A counts at reference position for strand (+) / counts at reference position for strand (-)
(5) C counts at reference position for strand (+) / counts at reference position for strand (-)
(6) G counts at reference position for strand (+) / counts at reference position for strand (-)
(7) T counts at reference position for strand (+) / counts at reference position for strand (-)
(8) Methylation level (-1 if it is not possible to calculate a value)
(9) Prediction of methylation status: M - methylated, U - not methylated, MP - a mutation produce a methylated C, UP - a mutation produce a not methylated C, P - a cytosine is mutated so it is not recognized for methylation , ? - non informative data for a correct prediction, SA - strand ambiguity (the alignments indicate that the Cs in the closest region are totally methylated on single or both strands but no information support a correct prediction)
(10) Only if -allele option is enabled. List of allele associated with a methylated cytosine Each allele is separated by a ';'. For instance 100/G/20 means (position=100, allele=G, counts=20)
(11) Only if -allele option is enabled. List of allele associated with a not methylated cytosine Each allele is separated by a ';'. For instance 300/T/10 means (position=300, allele=T, counts=10)

The Cs methylated in the (-) strand must be intended as Gs in the reference genome.

5.7.2 Evaluation of mapping performance using SOLiD data

You can download the SOLiD runs from the NCBI SRA archive (http://www.ncbi.nlm.nih.gov/sra). The following runs was considered:

1) SRR391055

2) SRR391056
3) SRR391057
4) SRR391058
5) SRR391059
6) SRR391060
7) SRR391061
8) SRR391062

this is the setting for the first strategy:

```
bin/pass_v1.7_I+ -p 11111111111111 -d  GENOME/hg19.fasta \
        -csfastq REAL_TEST_SET/SRRxxx.csfastq \
        -flc 4 -sam  \
        -seeds_step 3 -check_block 1000 \
        -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30 \
        -original -not_aligned -na_file REAL_TEST_SET/SRRxxx.NA.csfastq \
        >REAL_TEST_SET/SRRxxx.sam   2>REAL_TEST_SET/SRRxxx.log
```

this is the setting for the second strategy:

```
bin/pass_v1.7_I+ -p 11111111111111 -d  GENOME/hg19.fasta \
         -csfastq REAL_TEST_SET/SRRxxx.NA.csfastq \
         -flc 4 -sam  \
         -seeds_step 3 -check_block 1000 \
         -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30 \
         -max_combinations 8 \
         >REAL_TEST_SET/SRRxxx.NA.sam   2>REAL_TEST_SET/SRRxxx.NA.log
```

Mapping results:

Run	Strategy	U	RM	RC	%M	FL
SRR391055	1st strategy	4946895	6856216 / 9484529	(72.29 %)	3807626
	2nd strategy	468933	554798 / 2628266	(21.11 %)	47
SRR391056	1st strategy	6111596	8410908 / 12936642	(65.02 %)	4491416
	2nd strategy	750163	1185252 / 1845558	(64.22 %)	2680176
SRR391057	1st strategy	6028296	8020517 / 12186173	(65.82 %)	3080701
	2nd strategy	939928	1420597 / 2772224	(51.24 %)	1393432
SRR391058	1st strategy	4139055	5364105 / 9197705	(58.32 %)	1169432
	2nd strategy	788204	1212893 / 2405083	(50.43 %)	1428517
SRR391059	1st strategy	8535935	11526940 / 19962342	(57.74 %)	6593026
	2nd strategy	1798433	2649777 / 6460668	(41.01 %)	1974734
SRR391060	1st strategy	8274027	10839236 / 19585480	(55.34 %)	5159208
	2nd strategy	1904524	2767811 / 7209324	(38.39 %)	1536920
SRR391061	1st strategy	7832244	11093819 / 17532471	(63.28 %)	7992732
	2nd strategy	1510987	2192145 / 4484279	(48.89 %)	1954373
SRR391062	1st strategy	8347689	11232496 / 18784328	(59.80 %)	6078777
	2nd strategy	1506672	2293345 / 4965585	(46.18 %)	2586247

PASS mapping result of a SOLiD run. The column "Run" represents the SRR code of the run, "Strategy" the type of mapping strategy, "U" the number of reads that produce

unique alignments for strand and position, "RM" the number of mapped reads, "RC" the considered reads that have passed the quality check, "%M" the percent of mapped reads compared to the considered one and finally, "FL" the reads filtered because low quality.

Typically for a SOLiD run, the second strategy allows to recover about 50% of the reads not mapped in the first strategy and about 20% of the total reads.

For SOLiD data the recovered alignments in the second step is higher than the which one observed for the simulated data. This result evidencing a quality distribution difference of the two compared test sets.

5.7.3 Setting parameters for the first strategy

In order to understand the best setting for the first strategy we have tested the program using different setting and the same simulated test set.

SIMULATED TEST SET

The human hg19 genome was modified simulating C methylation at different levels. The modified reference was used as input for the program dwgsim to generate one million of simulated reads.
Used parameters: **[-y 0 -z 0 -d 100 -S 2 -c 0 or 1 (for Illumina or SOLiD data) -1 50 -2 50 -C -1 -N 1000000]**

The per base/color/flow error rate and the rate of mutation is set to the default values (respectively: 0.02 and 0.001). All simulated test sets were produced using the same seed, so they are comparable for number of reads, position and strand.

PASS program is evaluated for some critical parameters: the linguistic complexity (-flc), the pattern length (-p), the precomputed score table (-pst_word_range), the fraction of considered seeds (-seeds_step). For each of them the mapping process is repeated changing the value step by step. The percent of mapped reads and the elaboration time are reported as information to understand the best setting.

linguistic complexity setting (-flc parameter) (**colorspace**)

base setting: -p 111111111111111 -sam -seeds_step 3 -check_block 1000 -no_trim_auto -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30

flc	%MR	T (seconds)
2	79.38 %	169 s (more than 2 esamers)
3	95.54 %	1717 s (more than 3 esamers)
4	96.68 %	2251 s (more than 4 esamers)
1	97.52 %	4220 s (filter omopolymers)
0	97.54 %	4305 s (no filter)

low complexity seeds filter parameter effects on mapping. The "flc" indicates the number of set parameter, "%MR" the reads mapped for each analysis and T the time required to end mapping.

pattern length setting (-p parameter) (**colorspace**)

base setting: -sam -seeds_step 3 -check_block 1000 -no_trim_auto -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30 -flc 1

pattern	%MR	T (seconds)
111111111111111	97.52 %	4254 s
11111111111111	97.76 %	8052 s
1111111111111	97.97 %	15913 s
111111111111	98.17 %	31771 s

seed pattern length effects on colorspace mapping. The "pattern" indicates the structure of the pattern set, "%MR" the reads mapped for each analysis and T the time required to end mapping.

pattern length setting (-p parameter) (**basespace**)

base setting: -sam -seeds_step 3 -check_block 1000 -no_trim_auto -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30 -flc 1

pattern	%MR	T (seconds)
111111111111111	99.81 %	6471 s
11111111111111	99.81 %	12401 s
1111111111111	99.81 %	22906 s
111111111111	99.81 %	43158 s

seed pattern length effects on basespace mapping. The "pattern" indicates the structure of the pattern set, "%MR" the reads mapped for each analysis and T the time required to end mapping.

PST setting (-pst_word_range parameter) (**colorspace**)

base setting: -p 111111111111111 -sam -seeds_step 3 -check_block 1000 -no_trim_auto -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30 -flc 1

PST	%MR	T (seconds)
no pst used	98.04 %	16034 s
4	97.52 %	4278 s
5	96.71 %	3730 s
6	96.96 %	4034 s
7	95.92 %	3444 s

PST length effects on colorspace mapping. The PST is a matrix containing the precomputed score of each combination of short word alignment with a fixed size. The "PST" indicates the word size of the used PST, "%MR" the reads mapped for each analysis and T the time required to end mapping.

fraction of considered seeds (-seeds_step parameter) (**colorspace**)

base setting: -p 111111111111111 -sam -check_block 1000 -no_trim_auto -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30 -flc 1

fraction / param.value	%MR	T (seconds)
0.5 / 2	97.77 %	6303 s
0.33 / 3	97.52 %	4235 s
0.25 / 4	97.19 %	3154 s
0.20 / 5	96.86 %	2505 s
0.17 / 6	96.41 %	2152 s
0.14 / 7	95.96 %	1811 s
0.13 / 8	95.44 %	1623 s
0.11 / 9	95.12 %	1440 s
0.1 / 10	94.30 %	1264 s

Fraction of considered seeds effects on colorspace mapping. The "fraction / param.value" indicates the fraction of the considered seeds on mapping and the set of the seed step parameter to obtain this fraction, "%MR" the reads mapped for each analysis and T the time required to end mapping.

5.7.4 Setting parameters for the additional strategy

This analysis is referred to the mapping of the not aligned reads recovered from the first step (first strategy). Using a number of combination per seed of 5 we could recover about 40% of the alignments lost in the first step. An analysis of the alignments confirms that the mapped reads of the second step have the major sequencing errors located at the 5' end.

seed combinations setting (-max_combinations parameter) (**colorspace**)

base setting: -p 111111111111111 -flc 4 -sam -seeds_step 3 -check_block 1000 -no_trim_auto -bisulfite -fid 90 -b -cpu 12 -auto_seeds_limit 30

Comb./seed	%MR	T / R	TS
2	31.87 %	96.50 s / 24773	3172397910
3	34.90 %	96.46 s / 24773	4507568573
4	39.26 %	6.20 s / 24773	5860152276
5	40.08 %	7.27 s / 24773	6838247811
6	41.65 %	7.93 s / 24773	7839812660
7	42.59 %	7.99 s / 24773	8843818697
8	44.63 %	8.27 s / 24773	9853814411
9	44.85 %	10.16 s / 24773	10469605715
10	45.22 %	11.06 s / 24773	11108351607

Seed combinations effects for colorspace mapping. The "Comb./seed" field indicates the max number of combinations per seed set for each analysis, "%MR" the reads mapped, "T / R" the time required to end mapping and the considered reads to map and finally, "TS" the total number of considered seed load in RAM by the program for a specified set.

6 SNP calling

Recent advances in next-generation sequencing (NGS) technology now provide the potential to detect all single nucleotide polymorphisms (SNPs) including rare ones in a genomic region. The power of NGS based SNP detection is critically dependent upon the accuracy of base calling. The proposed

program take into account the base quality and the coverage analysis to minimize false positives of the base calls.

6.1 SNP calling program

This program allows the genotype analysis to detect SNPs, in/del or methylated Cs.
The input of the program consists on mapped data produced by PASS into SAM format and the reference genome fasta file. For bisulfite calling please see the chapter 5.

6.2 Setting parameters

***Example**: pass_vx.x -program genotype -sam sam_file1 -sam sam_file2 -ref genome.fasta -hits 2 > result.snp*

-sam	*(string)*	*sam file as input (you can add other SAM files using this options)*
-d	*(string)*	*Reference fasta file*
-bisulfite		*Enable methylation calling*
-bis-strand		This is a strand filter for bisulfite alignments. Default [0]. 0 it considers all strands. Set to 1 for strands ++ and +-. *Set to 2 for strands -- and -+*
-phred64		*PHRED64 quality format.*
-f	*(float)*	*This is the threshold frequency referred to the evidences that confirm a reference base. It allow to filter the noise data generated by sequencing error[0.2]*
-q	*(int)*	*Skipping the bases transition having quality less then this threshold value [10]*
-c	*(int,int)*	*Min. and Max numbers of evidences to report putative SNPs or in/dels (default auto calculated)*
-p	*(float)*	*This parameter considers the cumulative probability of a base transition belong the coverage distribution. In other word it allows to select the base transitions inside the coverage distribution in a range of cumulated probability between p to 1 [0.01].*
-hits	*(int)*	*The alignments with hits number <= set value, will be included in the analysis. [enabled to 1 for DNA-seq data; enabled to 2 for bisulfite sequencing data].*
-flank	*(int)*	*it not considers the variants occurred on the 3' flanking region of each alignment [4]*

Output format:
field n. / description for SNPs
(1) Reference Chromosome
(2) reference position
(3) reference base
(4) A counts at reference position
(5) C counts at reference position
(6) G counts at reference position
(7) T counts at reference position
(8) allele > set frequency [-f]
(9) format: [N/XX; ...] N= Number of insertions / XX = Inserted bases
(10) format: [N/XX; ...] N= Number of deletions / XX = Deleted bases

7 Pairing

Paired-end tags are the short sequences at the 5' and 3' ends of the DNA fragment of interest, which can be a piece of genomic DNA or cDNA. In theory, they should contain enough sequence information to be uniquely mapped to the genome and thus represent the whole DNA fragment of interest. The Mate-pair tags are used for deep sequencing of a whole genome. They may also be useful for some other applications, such as sequencing a selected region of the genome, where detection of structural genomic changes is desired. Point mutations and small insertions and deletions can be detected in the

resulting sequences. Moreover, sequences that are not located within an expected distance from each other alert us to the presence of significant structural aberrations, including translocations, inversions, deletions and insertions, in specific regions of the genome. A refined pairing process has great importance to detect structural aberration but also to do the scaffolding of new genome assembly.

7.1 Pairing program

This program executes a refined paired-end and mate-pair pairing for Sanger, SOLiD, and ILLUMINA platforms. It produces several classes of paired-end or mate-pair useful to make possible the above analysis. There are two ways to set the SAM files as input. The simple one, considers the two paired-end files merged to a single file (for instance you can use the command "cat file1 > file" and "cat file2 >> file") that you can pass to the program using "-sam1" or "-sam2" parameter. The second way require two files containing the aligned data of the 2 sequenced ends.

7.2 Setting parameters

***Example for SOLiD data**:*
pass -program pairing -sam1 pairF3_file -sam2 pairR3_file -range 0 3000 100000 -unique_pair 1 [-lib_type 1] [-pe_type 1] [-tags R3 F3] [-ref reference_file] -default_output -stdout > results.sam

***Example for ILLUMINA data**:*
pass -program pairing -sam1 pairF3_file -sam2 pairR3_file -range 0 600 600 -default_output -stdout -pe_type 0 -tags /1 / 2 [-ref reference_file] > results.sam

-cpu	*(int)*	*Enable multithreading function. Set the number of threads as the number of available processor cores.*
-sam1	*(string)*	*SAM file 1 as input. You can also use this option without -sam2 to load both paired-end data saved as one file on RAM. It is useful if the data are not ordered basing on the numbers of the read name but it requires great amount of memory.*
-sam2	*(string)*	*SAM file 2 as input if the paired data are split in two flat files.*
-token	*(char, int)*	*Optimization used when reads name are constituted by 3 or 4 ordered numbers separated by tokens and corresponding to plate coordinates. This option requires the data mapping organized as two files one for each end. e.g. for ILLUMINA paired-end -token : 2 -> use the number pointed after the second token ':' to generate a block of optimized data [-token _ 0 for SOLiD data]*
-range	*(int , int, int)*	*(Estimated_Min_Distance) (Estimated_Max_Distance) (Tolerated_Max_distance) [0 4000 100000](*
-pe_type	*(int)*	*if 0 [---> <---] : typically for BAC-END or FOSMID-END or ILLUMINA paired-end library if 1 [---> ---> or <--- <---] : typically for SOLiD mate-pair [default] if 2 [<--- --->] : typically for ILLUMINA paired-end library*
-lib_type	*(int)*	*1 directed library e.g: for SOLiD, 0 not directed library [1]*
-tags	*(string , string)*	*tag for each paired (tag1_end tag2_end) (example: R3 F3 default)*
-ref	*(string)*	*fasta reference sequences filename*
-o	*(string)*	*directory to save files*
-default_output		*Set default class of output:UNIQUE_PAIR, UNIQUE_SINGLE*
-stdout		*Instead to generate a file for each class of paired-ends it outputs all information to the stdout*
-append		*append output files to the old one*
-no_header		*it doesn't attach header to sam files*
-unique_pair	*(int)*	*(1) save or (0) don't save file*
-not_unique_pair	*(int)*	*(1) save or (0) don't save file*
-unique_wrong_s	*(int)*	*(1) save or (0) don't save file*
-not_unique_wrong_s	*(int)*	*(1) save or (0) don't save file*
-unique_wrong_d	*(int)*	*(1) save or (0) don't save file*
-not_unique_wrong_d	*(int)*	*(1) save or (0) don't save file*

-unique_single (int) (1) save or (0) don't save file
-not_unique_single (int) (1) save or (0) don't save file
-unique_pair_out (int) (1) save or (0) don't save file
-one_not_unique_pair_out (int) (1) save or (0) don't save file
-both_not_unique_pair_out (int) (1) save or (0) don't save file
-discarded_pair (int) (1) save or (0) don't save file n

***Output Format**:*
The output files will be organized as gff or SAM format. GFF: Paired-ends will be associated on two consecutive rows of data. The correct paired-ends have the field number 3 defined as 'match', otherwise for wrong distance or strand orientation will be defined as 'Wrong-D' or 'Wrong-S' respectively.

Meaning of the Output files

UNIQUE_PAIR	*file contains unique pair reads with correct distance and orientation*
NOT_UNIQUE_PAIR	*file contains not unique pair reads with correct distance and orientation*
UNIQUE_WRONG_S	*file contains unique pair reads with wrong strand*
NOT_UNIQUE_WRONG_S	*file contains not unique pair reads with wrong strand*
UNIQUE_WRONG_D	*file contains unique pair reads with wrong distance*
NOT_UNIQUE_WRONG_D	*file contains not unique pair reads with wrong distance*
UNIQUE_SINGLE	*file contains single unique read with no association*
NOT_UNIQUE_SINGLE	*file contains not unique read with no association*
UNIQUE_PAIR_OUT	*file contains unique paired-ends not associated to the same reference*
ONE_NOT_UNIQUE_PAIR_OUT	*file contains one unique end (pair) and the other not unique and not associated to the same reference*
BOTH_NOT_UNIQUE_PAIR_OUT	*file contains both ends (pair) not unique and not associated to the same reference*
DISCARDED_PAIR	*file contains discarded paired-ends*

Outputs paired-end combinations only for:
NOT_UNIQUE_PAIR, NOT_UNIQUE_WRONG_D , NOT_UNIQUE_WRONG_S

8 List of parameters

READS FILE SETTING

[-fasta file (string)]
Set input reads as multiple fasta file.

[-csfasta file (string)]
Set input color space reads as multiple fasta file. You must use this parameter with -qual parameter.

[-qual file (string)]
Set input quality reads as multiple fasta file (only associated with -csfasta parameter)

[-fastq file (string)]
Set input reads as fastq file.

[-csfastq file (string)]
Set input color space reads as fastq file.

[-query_size (int)]
Set the max read length: if you don't know the max reads length try to set this value to a great number (e.g. 1000). In case of many reference sequences this parameter can alter significantly the memory requirements.

[-trim w q (int,int)]
Trimming window: this function trims sequences at both ends. Set w for window length and q for
quality threshold. The trimming is executed at the last base with quality < q. This parameter is auto set using versions > 1.6.
Suggested [-trim 6 20]

[-trim_avg (int)]
If the average quality of the read is < n the sequence will be filtered [9].

[-max_trim_len (int)]
Set the max trim len of the reads accepted as alignable sequences [20].

[-check_block n (int)]
It defines the reads number loaded during the auto setting mode. [100000]
If you increase this parameter you will obtain the best settings in order to align your data.

[-no_quality_check]
With this parameter PASS not checks quality in order to filter low quality reads.
Expecially using SOLiD reads you could loss data.

[-no_trim_auto]
Disable trimming function. With this parameter PASS not checks quality in order to find best settings.
Especially using SOLiD reads you could loss data.

[-trim_mode (int)]
If 0 all reads that pass quality filter (-trim_avg) will be trimmed at both ends if necessary.
If 1 PASS selects the read regions with higher quality trimming the other. The -trim parameter has no effects.

[-Ns_percent] (int)]
Using this parameter PASS skips the reads that have a percentage of Ns > set value [10]

CLEANING SEQUENCING DATA

Please set -p to 111111 in order to map with the higher sensitivity, -fid to 90 and add the -l -b -fle (int)
parameters to a proper value [16] if you think to find only a small part of the adaptor. The processed reads will be output to the stdout. The input reads must be in csfasta/qual, csfastq or fastq files.

[-cleaned_mask]
It masks the adaptor sequence .

[-cleaned_ends n (int)]
it selects both ends removing the adaptor. n is the size threshold of the cleaned read.

[-cleaned_end n (int)]
it selects only the major piece resulting by adaptor removing. n is the size threshold of the cleaned read.

[-cleaned_3end n (int)]
it selects only the 5' sequence resulting by adaptor removing. n is the size threshold of the cleaned read.

[-cleaned_5end n (int)]
it selects only the 3' sequence resulting by adaptor removing. n is the size threshold of the cleaned read.

[-cleaned_all]
Includes also the sequences not selected with --cleaned_ends, --cleaned_end or --cleaned_mask parameters.

REFERENCE FASTA FILE AS INPUT

[-d (string)]
Set input reference fasta file. This file could be a fasta or multifasta file in base-space or double encoded format.

[-double_encoded]
Use this parameter to align double encoded reads to double encoded reference resulted by assembling of SOLiD data (for instance using Velvet assembly).

REFERENCE DATABASE SETTING

[-seeds_step (int)]
This parameter is useful to save RAM in the indexing process. PASS consider only the seeds at seeds step distance [1].

[-auto_seeds_limit (int)]
PASS limits the number of seeds becoming from each possible combination. n represent how many standard error must be considered as threshold for a typical random reference [disabled]. This parameter is useful to save RAM in the indexing process of bisulfite mapping.

[-D file (string)]
This parameter allows to store the database in a binary file. This step reduces the time required to reiterate the analysis on the same genome. You can use this parameters with -d, -p, -solidCS and -flc.

[-solidCS]
Use this parameter with -D if you have to map SOLiD data. A COLOR SPACE database will be created.

[-R file (string)]
This paramater allows to READ a database binary file to RAM. This step reduces the time required to reiterate the analysis on the same genome.

PARAMETERS THAT COULD AFFECT MAPPING

[-sensitivity (int)]
Auto set for buffer saturation. you can set to 0 for low , 1 for medium, 2 good, 3 for best sensitivity.
If you increase this parameter you will obtain the best settings in order to align your data. The desired sensitivity could change the execution time. [1]

[-flc n (int)]
Low complexity filter for seed words indexing.
Set to 0: no seed word filter;
Set to 1: it filters the seed word if there are at least two homopolymeric hexamers;
Set to >=n: it filters the seed word if there is a number of same hexamers >

[-fid n (int)]
It allows to filter alignments that have the percent identities less than this value.[90]

[-fle n (int)]
Force the filtering of the alignments with size < of n bases. This parameter must be set always together the -l parameter but it could be required also for spliced alignments for particular situations.

[-pst PST_file t (string, int)]
The PST is a matrix that contains precomputed score of each combination of short word alignment with a fixed size. It is used for the flanking regions checking. In those conditions PASS increase the performance in the extension step. The t value is a threshold required by the pst parameter strictly correlated by number of discrepancies (intended as % (mismatch + gap) set with -fid parameter. For example if the read size is 33 bases and the fid parameter is 90; PASS tolerates 0 to 3 discrepancies for each alignments (10% of 33 b). The PST threshold must be set with the same number of discrepancies, if we want to obtain 100% of hits without loss in sensitivity. With default setting this parameter is auto set.

[-no_pst_auto]

Disable auto PST processing. Alternatively, you can use -pst parameter in order to read PST as a file and to set a custom threshold . (By default, PASS will select the threshold basing on -fid parameter).

[-pst_word_range (int, int)]
The PST word range determines number and type of PST processed during the initial elaboration of the structures. With default setting this parameter is auto set.

[-max_seed_hits (int)]
This parameter must be set properly in order to have a minimal seed buffer saturation. Furthermore this set depends on several conditions e.g. the redundancy and the length of the reference. You can monitor the seed buffer saturation checking the log information. You needs to Set this parameter properly because it could affects the elaboration time and mapping efficiency. With default setting this parameter is auto set.

[-max_best_hits (int)]
This parameter must be set properly in order to have a minimal hits buffer saturation. This set depend on several conditions e.g. the redundancy and the length of the reference. You can monitor the hits buffer saturation checking the log information. You needs to Set this parameter properly because it could affects the mapping efficiency. With default setting this parameter is auto set.

OUTPUT PARAMETERS

[-sam]
Output the results as standard SAM format:

Meaning of the optional user's fields:
1) Field X1: This field reports information about trimming and specificity of alignments (if SOLiD color space).

X1:Z:1_48_S1E48A17L48_T0Q32_BS1

S1E48	Read trimmed at original positions 1 and 48
A17	Read average quality after trimming 17
L48	Read length after trimming 48 bp
T0	The first base and the first color are T and 0
Q32	Quality of the first color is 32
BS1	1st position of the alignment confirms the reference
BS0	Reference base non confirmed

2) Field X2: Intron type

X2:Z:[GT..AG]	Type of intron defined by 5' and 3' bases

3) Field X3: Intron score

X3:f:10.0000	Intron score to distinguish canonical and not canonical intron.

4) Field X4: spliced alignment probability

X4:f:0.30000	E value. It defines the probability that the small alignment is random one time the large one is mapped.

5) Field X9: strand information (only using -bisulfite parameter):

ZS:Z:+-	It defines the reference strands (Watson + or Crick -) and the orientation of the mapped read.

The SAM format is not implemented with parameter -spliced dna (only GFF) but work with -spliced rna and the other alignment functions.

-bowtie_sam
Bowtie output compatibility for strand info: it adds user field XS:A:[+|-] (deprecated)

[-gff]

Output the results as GFF3 format

[-info_gff]
Add to the gff 'note' field the gaps and mismatches position. Example: M:2 -> 12 A/T 15 T/C,G:2 -> Q18/G R40/A/T (2 mismatch, the first one at relative position 12 with 'A' for query and 'T' for reference; second one at relative position 15 T/C mismatch. There is a gap on query sequence at relative position 18 with a 'G' base on the reference and 1 gap on reference at relative position 40 with 'A' base on query and a 'T' on the reference sequence after the gap.)

[-b]
Only the Best Hits alignments

[-uniq]
Pass reports only the unique alignments

[-no_qual]
Alignments quality reports disabled.

[-seq_gff]
It adds to GFF outputs short read sequence (same orientation of the alignment).

[-original]
it is used with -aligned or/and -not_aligned parameters.
Save aligned or/and not aligned reads in the original format (not trimmed).

[-not_aligned]
It appends to not_aligned.fa file unaligned sequences in the same format as the input.

[-aligned]
It appends to aligned.fa file aligned sequences in the same format as the input.

[-a_file]
Aligned reads filename. use this parameter with -aligned.

[-na_file]
Unaligned reads filename. use this parameter with -not_aligned.

[-tmp_path]
set the temporary directory for tmp files. By default is set to the same directory used to run PASS.

MAIN SETTING

[-block n (int)]
Number of reads loaded for each block. Default [100000]

[-repeat]
It disables redundant sequences optimization (spliced alignments).

[-g n (int)]
Enable gap analysis: This value determines the size of the score matrix (reads size * (2*n+1)).
This means that the max gap recognized is: 2 * n. and |insertions - gaps| must be <= n. [0]

[-gap n (int)]
Penalty for each GAP used to calculate the score of each alignments [1]

[-match (int)]
match base score used to calculate the score of each alignment [1]

[-p (string)]

seed word structure (1 is a defined position, 0 is an undefined position). e.g. for SOLiD you could use 11111100111111 or 111111100111111 else for other platforms use 1111110111111 or 111111101111111 or 111111111111111. For cleaning data by adaptors you could use 11111111.

[-l]
Enable the local alignment function. It try to recognize local small alignments of each read.

[-ext_limit n (int)]
extern limit to set margins of local alignments: this function trims the ends of the alignment when the number of contiguous mismatches or gaps are > n. This parameter is auto set.

[-int_limit n (int)]
intern limit to set margins of local alignments: one time -ext_limit function has been defined the outer margins of the alignments, the -int_limit function adjust the ends as it finds a contiguous matches >= n. This parameter is auto set.

[-s (int)]
Increase sensitivity level for repetitive and very small local alignments.

[-S (int)]
Alignment mode [2]
0: aligning only for strand (+)
1: aligning only for strand (-)
2: aligning for both strands

Bisulfite threated reads mapping:
3: only map to 2 forward strands ++ and -+\
4: only map to 2 reverse strands +- and --
5: map to all 4 possible strands

[-diag n (int)]
Length of smallest diagonal not considered as background in the score matrix alignment.[4].

[-open_gap n (int)]
Open gap penalty [2].

[-cpu n (int)]
Number of threads for multithreading mode. n must be the number of processor cores. [1]

[-cs_error (int)]
Penalty for color space sequencing errors. [1]

[-phred33]
input qualities are Phred+33 format (default)

[-phred64]
input qualities are Phred+64 format (Typical of ILLUMINA sequencing data)

[-j]
Without this parameter PASS reports preferentially ungapped alignments jumping pedantic dynamic programming analysis if possible.

SPLICED ALIGNMENT

There are two main options for spliced alignments:

The first one enabled by "-spliced rna" parameter is useful to do the splicing site prediction using cDNA libraries. The second one, enabled by "-spliced dna" is useful to detect DNA structural variations using DNA libraries.

[-spliced Cn Sn ... (int, ..)]
It Outputs spliced alignments using the user intron scoring system. Please, set a list of preferred CODE and SCORE for cis splice sites. Cn is the cis splice site code, Sn the associated score. Without values PASS reports a list of codes for splicing sites usefull to set Cn values.

Use [-spliced default] command to use default settings:
INTRON TYPE: GT..AG CODE:178 SCORE:10
INTRON TYPE: CT..AC CODE:113 SCORE:10
INTRON TYPE: GC..AG CODE:146 SCORE:5
INTRON TYPE: CT..GC CODE:121 SCORE:5
INTRON TYPE: AT..AC CODE:49 SCORE:2
INTRON TYPE: GT..AT CODE:179 SCORE:2

[-i_score n (float)]
According to the scoring system upon described, all the spliced alignments with a score less than n will not be considered. The default value is 2

[-e n (float)]
It is not found by chance within the -max_distance sequence. If the calculated probability is > n, the alignment will not be considered. The default value is 0.5

[-logo n (int)]
Enable logo analysis to calculates and reports the frequency of each base position near spliced regions in a range of n+n bases centered on splicing sites. It must be >=2 [8]

[-report n (int)]
Splicing site frequencies reports (sars.info file). reports the best n scores.[10000]

[-overlap (int)]
the precise position of the splicing site cannot be determined based only on the alignment. If the
extent of the overlap is greater than value, the spliced alignment will not be considered.The default value is 8

[-redundancy]
It Disables redundancy optimization for spliced alignments

[-max_distance (int)]
This parameter set the maximal length of intron or the maximal distance accepted between two alignments. The default value is 100000

[-spliced_hits n (int)]
Outuputs the first n alignments of spliced reads [as max_best_hits]

[-percent_tolerance n (int)]
It defines the percent of read size tolerated as not matched for spliced alignments. The sum of the two alignments spliced must be >= (Read legth - n); where n = (Read legth * percent_tolerance)/100; otherwise will be filtered.

[-tolerance n (int)]
It defines the max tolerated not matched bases for spliced alignments. The spliced alignment length must be >= (Read legth - n); otherwise will be filtered. If you use the parameter -focus by default this parameter is set to 0 otherwise it is set to (seed pattern length).

if [-spliced dna] is enabled those conditions are checked:

1) L1+L2 > (read_size-n) if both alignments are found
2) (L1 or L2) > (min_size+n) if a single alignment is found.
L1 and L2 are the length of each alignment while min_size
the minimal length of tolerated alignment's set with -fle parameter !!!.

if [-spliced rna] is enabled those conditions are checked:

1) L1+L2 > (read_size-n) if both alignments are found.
2) or (L1 or L2) > (read_size-n) if a single alignment is found.
by default pass set -tolerance to 5 units less than the read size.

[-percent_anchor n (int)]
It defines the minimal percent of matched bases tolerated for each spliced alignments.
In case of short reads or particular situations where sequences are trimmed it could be useful. The spliced alignment length must be >= (Read legth * percent_anchor)/100; otherwise will be filtered. If you use the parameter -focus by default this parameter is set to 0 otherwise it is set to (seed pattern length).

if [-spliced dna] is enabled those conditions are checked:

1) L1+L2 > (Read legth * percent_anchor)/100 if both alignments are found
2) (L1 or L2) > (min_size+n) if a single alignment is found.
L1 and L2 are the length of each alignment while min_size
the minimal length of tolerated alignment's set with -fle parameter !!!.

if [-spliced rna] is enabled those conditions are checked:

1) L1+L2 > (read_size-n) if both alignments are found.
2) or (L1 or L2) > (read_size-n) if a single alignment is found.
by default pass set -tolerance to 5 units less than the read size.

[-anchor n (int)]
In case of short reads or particular situations could be possible to align only one alignment that define the 5' or 3' splicing site edge. This parameter represents the minimal length tolerated by the program for 'spliced and not spliced' alignment that match splicing site.[25]

[-focus_check filename (string)]
Save candidates spliced reads into 'filename' in order to map them again, using short seed words together the parameter -focus. The resulted output reads will contain information about local alignments coordinates mapped into the reference. Those coordinates will be recovered in the next mapping to search around these positions for spliced alignments.

[-focus]
-focus used with '-spliced' parameter performs a fast spliced alignment of great genomes. You need to pass the filename with the unspliced reads filename obtained in the previous analysis using -focus_check parameter. These reads contains the reference coordinates that allow PASS to use only a small fraction of seed words database. This strategy allow to save time and increase map sensitivity for spliced alignments.

NGS BISULFITE MAPPING

[-bisulfite]
It enables PASS to map the epigenetic data (base space and color space data).

[-max_combinations (int)]
This parameter changes the seeds indexing of the bisulfite mapping. Usually is not required but may be useful especially for color space sequences that don't map with the default indexing mode. This parameter enable the second strategy of the seeds word indexing that consider a large number of seeds combinations where the cytosines for each seed may be methylated or not. You should set it to a number >= 2 [not used]

[-combinations_offset (int)]
This parameter must be used according to the -max_combinations parameter in order to limit the number of seeds combinations in a range between [combinations_offset to max_combinations]. It allows to save RAM using only a small fraction of generated seeds. Using this parameter you could map the reads in more than one step recovering the not aligned reads. In other words, the last generated seeds is not considered because the program starts considering new seeds not considered in the last steps. Example: if in the previous step you have used -max_combinations 4 -combinations_offset 0; in the next mapping you could set them to -max_combinations 8 -combinations_offset 4 because the last 4 were just analyzed. [default 0].

[-only_c_context]
This parameter reduces the RAM requirement and should be used with -bisulfite parameter. It allows to consider only seed words that have at least one cytosine in the sequence. The reads out of this context will not be mapped. This function is allowed only using the "-max_combinations" parameter.