API Documentation

API Documentation

September 22, 2016

Contents

Contents **1**

1 Script script-BAMmaker **2**
1.1 Functions 2
1.2 Variables 4

2 Script script-SNP_INDEL_merge **5**
2.1 Functions 5
2.2 Variables 5

3 Script script-VARCall **6**
3.1 Functions 6
3.2 Variables 7

4 Script script-VCFilter **8**
4.1 Functions 8
4.2 Variables 8

5 Script script-VCFmaker_INDEL **9**
5.1 Functions 9
5.2 Variables 9

6 Script script-VCFmaker_SNP **10**
6.1 Functions 10
6.2 Variables 10

7 Script script-VCFtoFASTA **11**
7.1 Functions 11
7.2 Variables 11

8 Script script-VCFtoMATRIX **12**
8.1 Functions 12
8.2 Variables 12

9 Script script-VCFtoPseudoGenome **13**
9.1 Functions 13
9.2 Variables 13

10 Script script-reportMaker **15**
10.1 Functions 15
10.2 Variables 16

1 Script script-BAMmaker

Make a BAM filtered and sorted from FASTQ and a reference FASTA File. This script was designed for the VARCall workflow.

Requires:

- BWA
- samtools
- picard-tools
- Trimmomatic
- GATK

1.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

exec_commands(*cmds*, *nbThreads*)

```
Exec commands in parallel in multiple process (as much as we have CPU)
@param cmds: list of commands
    @type cmds: list
    @param nbThreads: number of threads to use
    @type nbThreads: integer
```

exist(*fname*)

Check the existence of a file.

Parameters

`fname`: file name

(type=string)

Return Value

1 if the file is present, 0 otherwise

(type=integer)

lenGenome(*fname*)

Return the length of a sequence in a FASTA file (1 contig).

Parameters

`fname`: FASTA file name

(type=string)

Return Value

the number of nucleotide in the sequence

(type=integer)

coverage(*fname*, *fref*)

Compute the depp coverage and breadth coverage from a BAM file.

Parameters

`fname`: BAM file name

(type=string)

`fref`: reference sequence in FASTA format

(type=string)

Return Value

breadth coverage

(type=float)

checkPredScore(*fname*)

Unstable function, not used. Check the phred score format.

Parameters

`fname`: FASTQ file name

(type=string)

Return Value

phred score format (33 or 64)

(type=int)

changePredScore(*fname*)

Unstable function, not used. Change the phred score format.

Parameters

`fname`: FASTQ file name

(type=string)

Return Value

FASTQ file name in phred 33

(type=string)

nbReads(*liste*)

Compute the number of reads in FASTQ files.

Parameters

`liste`: list of FASTQ files

(type=list)

Return Value

number of reads

(type=integer)

main()

1.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

2 Script script-SNP_INDEL_merge

Merge SNPs VCF file and InDels VCF file. This script was designed for the VARCall workflow.

Requires:

- GATK
- SnpSift

2.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

exec_commands(*cmds*, *nbThreads*)

```
Exec commands in parallel in multiple process (as much as we have CPU)
@param cmds: list of commands
    @type cmds: list
    @param nbThreads: number of threads to use
    @type nbThreads: integer
```

main()

2.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

3 Script script-VARCall

Do the SNP/Indel calling from several FASTQ files

Requires:

- BWA
- samtools
- picard-tools
- Trimmomatic
- GATK
- SnpSift
- BAMmaker, VCFmaker_SNP, VCFmaker_INDEL, VCFmaker_INDEL and VCFtoFASTA in $PATH

  Paired-end reads must be nammed as "prefix_R1.XXXX" and "prefix_R2.XXXX". Example : E1_R1.fq and E1_R2.fq

3.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

checkPE(*readsList*)

checks all paired-end reads and reordering them

Parameters

`readsList`: list of reads given by the argument parser

(type=list)

Return Value

read list

(type=list)

commandLaunched(*Arguments*, *logFile*)

Print VarCall command in log file

Parameters

`Arguments`: list of arguments

(type=parser object)

`logFile`: log file

(type=string)

checkFiles(*fileList*)

Check if all file of a list exist

Parameters

`fileList`: list file pathway

(type=list)

main()

3.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`
accept2dyear	**Value:** `1`
altzone	**Value:** `-7200`
daylight	**Value:** `1`
timezone	**Value:** `-3600`
tzname	**Value:** `('CET', 'CEST')`

4 Script script-VCFilter

Filter VCF file by deleting lines with variant only in reference, miss informations in alignment and delete INDEL when they are SNP at the same position. This script was designed for the VARCall workflow.

4.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

different_in_list(*list*)

Return True if an element in a list exist at least twice

Parameters

`list:` list of nucleic

`type:` *(type=list)*

Return Value

True if an element in a list exist at least twice, False otherwise

(type=boolean)

main()

4.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

5 Script script-VCFmaker_INDEL

Do the InDel calling from BAM files. This script was designed for the VARCall workflow.

Requires: GATK

5.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

exec_commands(*cmds*, *nbThreads*)

```
Exec commands in parallel in multiple process (as much as we have CPU)
@param cmds: list of commands
    @type cmds: list
    @param nbThreads: number of threads to use
    @type nbThreads: integer
```

main()

5.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

6 Script script-VCFmaker_SNP

Do the SNP calling from BAM files. This script was designed for the VARCall workflow.

Requires: GATK

6.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

exec_commands(*cmds*, *nbThreads*)

```
Exec commands in parallel in multiple process (as much as we have CPU)
@param cmds: list of commands
    @type cmds: list
    @param nbThreads: number of threads to use
    @type nbThreads: integer
```

main()

6.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

7 Script script-VCFtoFASTA

Create concatenate variant fasta file from VCF file. This script was designed for the VARCall workflow.

7.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

write_fasta(*outputFile*, *dicoResult*)

Write a multi FASTA file from dictionnary.

Parameters

`outputFile`: output FASTA file name

(type=string)

`dicoResult`: dictionnary with header for key and the list of nucleotide for value

(type=dictionnary)

main()

7.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

8 Script script-VCFtoMATRIX

Create distance matrix in tabular file from VCF file. This script was designed for the VARCall workflow.

8.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

write_dico(*dico*, *file*)

Write the matrix in an output file

Parameters

`dico`: dictionnary with for each genomes a dictionnary of SNP/INDEL differences between itself and other genomes

(type=dictionnary)

`file`: output file name

(type=string)

main()

8.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

9 Script script-VCFtoPseudoGenome

Make pseudo genome from a VCF file and the reference FASTA file (1 contig). This script was designed for the VARCall workflow.

9.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

write_fasta(*outputFile*, *dicoResult*)

Write a multi FASTA file from dictionnary.

Parameters

`outputFile`: output FASTA file name

(type=string)

`dicoResult`: dictionnary with header for key and the list of nucleotide for value

(type=dictionnary)

fasta_to_dico(*FASTAfile*)

Stock sequence from FASTA file to dictionnary.

Parameters

`FASTAfile`: FASTA file name

(type=string)

`record_dict`: dictionnary with header for key and sequence for value

(type=dictionnary)

check_ref(*dicoRef*)

Check if the reference FASTA file contain only one contig.

Parameters

`dicoRef`: dictionnary with header for key and sequence for value

(type=dictionnary)

main()

9.2 Variables

continued on next page

Name	**Description**

Name	**Description**
__doc__	**Value:** `...`
__package__	**Value:** `None`

10 Script script-reportMaker

Make a PDF report from the log file generated by the VarCall workflow.

10.1 Functions

get_parser()

Parse arguments

Return Value

arguments list

(type=parser object)

texHeader(*texFile*)

Write the header of the LaTeX output file

Parameters

`texFile:` path of the output LaTeX file

(type=string)

writeVarCallCommand(*lines*, *texFile*)

Write the VarCall command in the LaTeX output file

Parameters

`lines:` lines of the log file

(type=list)

`texFile:` path of the output LaTeX file

(type=string)

extractReadStatistics(*lines*)

Extract reads statistics from the log file lines.

Parameters

`lines:` lines of the log file

(type=list)

Return Value

dictionnary with sample name in key and for value an other dictionnary with the number of reads before and after trimming, the number of reads mapped and properly paired, and the deep and breadth coverage.

(type=dictionnary)

workdir()

Find working directory and transformed it for write in latex file

Return Value

working directory transformed

(type=string)

version()

Find all version of all tools used by VarCall

Return Value

dictionnary with program name as key and version as value.

(type=dictionnary)

main()

10.2 Variables

Name	Description
__doc__	**Value:** `...`
__package__	**Value:** `None`

Index

script-BAMmaker *(script)*, 2–4
- script-BAMmaker.changePredScore *(function)*, 3
- script-BAMmaker.checkPredScore *(function)*, 3
- script-BAMmaker.coverage *(function)*, 3
- script-BAMmaker.exec_commands *(function)*, 2
- script-BAMmaker.exist *(function)*, 2
- script-BAMmaker.get_parser *(function)*, 2
- script-BAMmaker.lenGenome *(function)*, 2
- script-BAMmaker.main *(function)*, 4
- script-BAMmaker.nbReads *(function)*, 3

script-reportMaker *(script)*, 15–16
- script-reportMaker.extractReadStatistics *(function)*, 15
- script-reportMaker.get_parser *(function)*, 15
- script-reportMaker.main *(function)*, 16
- script-reportMaker.texHeader *(function)*, 15
- script-reportMaker.version *(function)*, 16
- script-reportMaker.workdir *(function)*, 15
- script-reportMaker.writeVarCallCommand *(function)*, 15

script-SNP_INDEL_merge *(script)*, 5
- script-SNP_INDEL_merge.exec_commands *(function)*, 5
- script-SNP_INDEL_merge.get_parser *(function)*, 5
- script-SNP_INDEL_merge.main *(function)*, 5

script-VARCall *(script)*, 6–7
- script-VARCall.checkFiles *(function)*, 6
- script-VARCall.checkPE *(function)*, 6
- script-VARCall.commandLaunched *(function)*, 6
- script-VARCall.get_parser *(function)*, 6
- script-VARCall.main *(function)*, 7

script-VCFilter *(script)*, 8
- script-VCFilter.different_in_list *(function)*, 8
- script-VCFilter.get_parser *(function)*, 8
- script-VCFilter.main *(function)*, 8

script-VCFmaker_INDEL *(script)*, 9
- script-VCFmaker_INDEL.exec_commands *(function)*, 9
- script-VCFmaker_INDEL.get_parser *(function)*, 9
- script-VCFmaker_INDEL.main *(function)*, 9

script-VCFmaker_SNP *(script)*, 10
- script-VCFmaker_SNP.exec_commands *(function)*, 10
- script-VCFmaker_SNP.get_parser *(function)*, 10
- script-VCFmaker_SNP.main *(function)*, 10

script-VCFtoFASTA *(script)*, 11
- script-VCFtoFASTA.get_parser *(function)*, 11
- script-VCFtoFASTA.main *(function)*, 11
- script-VCFtoFASTA.write_fasta *(function)*, 11

script-VCFtoMATRIX *(script)*, 12
- script-VCFtoMATRIX.get_parser *(function)*, 12
- script-VCFtoMATRIX.main *(function)*, 12
- script-VCFtoMATRIX.write_dico *(function)*, 12

script-VCFtoPseudoGenome *(script)*, 13–14
- script-VCFtoPseudoGenome.check_ref *(function)*, 13
- script-VCFtoPseudoGenome.fasta_to_dico *(function)*, 13
- script-VCFtoPseudoGenome.get_parser *(function)*, 13
- script-VCFtoPseudoGenome.main *(function)*, 13
- script-VCFtoPseudoGenome.write_fasta *(function)*, 13