README file for CONSERTING software distribution

(I) Initial setup

1. Install R (64-bit) software (http://www.r-project.org/) and the tree package.

2. Download the following script to the code directory:
   http://hgdownload.cse.ucsc.edu/admin/exe/linux.x86_64/bigWigToWig

   http://hgdownload.cse.ucsc.edu/admin/exe/linux.x86_64/wigToBigWig

3. Construct the Mapability track (Once for each window size and assembly):
   a. Download the Alignability track from UCSC Genome Browser. Choose the appropriate track according to your read length. For example, the wgEncodeCrgMapabilityAlign100mer.bw file is used for the example because the read length is 100 bp in this dataset.
      i. Hg19: http://genome.ucsc.edu/cgi-bin/hgFileUi?db=hg19&g=wgEncodeMapability
      ii. Hg18: http://hgdownload.cse.ucsc.edu/goldenPath/hg18/encodeDCC/wgEncodeMapability/

   b. Extract the downloaded track using the bigWigToWig script downloaded at Step 2.

      ```
      code/bigWigToWig DOWNLOADED_MAP_TRACK MAP_WIGGLE_FILE
      ```

      Parameters:

      `DOWNLOADED_MAP_TRACK`: The downloaded BigWig file from Step I-3a.

      `MAP_WIGGLE_FILE`: Name of the extracted wiggle file

      Example:

      ```
      code/bigWigToWig
      Mapability/hg18/wgEncodeCrgMapabilityAlign100mer.bw
      Mapability/hg18/full.wig
      ```

   c. Create the chromosome length file for the assembly (For hg18 and hg19 assembly, the file is included for convenience: Mapability/hg18/chromosome_lengths.txt and Mapability/hg19/chromosome_lengths.txt)

   d. Run the java code `ExtractMappability` to construct the mapability track for CONSERTING.

```
java -Xmx3g -cp code/ ExtractMappability DEST_DIR
MAP_WIGGLE_FILE CHR_LENGTH_FILE SIZE
```

Parameters:
`DEST_DIR`: Destination mapability directory (make sure it exists)
`MAP_WIGGLE_FILE`: Name of the extracted wiggle file from Step I-3b
`CHR_LENGTH_FILE`: The file containing the chromosomal length information (from Step I-3c)
`SIZE`: window size

Example:

```
java -Xmx3g -cp code/ ExtractMappability Mapability/hg18
Mapability/hg18/full.wig
Mapability/hg18/chromosome_lengths.txt 100
```

4. Construct the GC content track (Once for each window size and assembly).

   ```
   java –cp code/ FA2GC SIZE FA_DIR  DEST_DIR
   ```

   Parameters:
   `SIZE`: window size
   `FA_DIR`: FA_DIR: Directory for the FA files of the assembly. Expected files in the directory are 1.fa, 2.fa, …, X.fa and Y.fa.
   `DEST_DIR`: Destination GC directory (make sure it exists)

   Example:

   ```
   java –cp code/ FA2GC 100 HG18-FA-Location  GC/hg18
   ```

5. Install the CREST software (http://www.stjuderesearch.org/site/lab/zhang)

6. Download bambino_core.jar file from Bambino package[1] to the code directory, which is typically named bambino_core_X.XX.jar, e.g. bambino_core_1.04.jar (https://cgwb.nci.nih.gov/goldenPath/bamview/documentation/bam_utils.html#finney)

7. Download sam-1.27-SIGNED.jar from PICARD to the code directory (http://picard.sourceforge.net)

8. Download mysql-connector-java-5.1.10-bin.jar to the code directory (http://dev.mysql.com/downloads/connector/j)

9. Include the code directory in the PATH variable. Include the code directory and the jar files in the CLASSPATH variable (or specify the correct path in the command line)

Example:

```
CONSERTING_ROOT_DIR=`pwd`
export PATH=$PATH:$CONSERTING_ROOT_DIR/code
export CLASSPATH=$CLASSPATH:$CONSERTING_ROOT_DIR/code:$CONSERTING_ROOT_DIR/code/ bambino_core_1.04.jar:$CONSERTING_ROOT_DIR/code/sam-1.27-SIGNED.jar:$CONSERTING_ROOT_DIR/code/mysql-connector-java-5.1.10-bin.jar
```

(II) For each sample:

1. Run the coverage analysis for each BAM
   a. BAM2WIG: this substep calls a class (Ace2.SAMFinneyCoverage) from Bambino package[1].
      Note: this step requires 9 Gb of memory to ensure data integrity.

      ```
      sh code/bam2wig.sh INPUT_BAM WIG_FILE_NAME DEST_DIR
      ```

      Parameters:

      `INPUT_BAM`: Input BAM file

      `WIG_FILE_NAME`: Output filename (a gzipped file)

      `DEST_DIR`: Destination directory

      Example:

      ```
       sh code/bam2wig.sh test_data/BAM/SJSIM001_D-CREST-2011-03-nq.chr22.bam SJSIM001_D.wig.gz test_data/Coverage/
      sh code/bam2wig.sh test_data/BAM/SJSIM918_G-WUHG18-nomut-nq.chr22.bam SJSIM001_G.wig.gz test_data/Coverage/
      ```

   b. WIG2BW: this substep uses UCSC "WigToBigWig" utility to convert the wiggle format coverage to bigWiggle.
      Note: 1) This step requires 35 Gb of memory for human genome. The memory required could be different for other genomes. 2) Files hg18.size and hg19.size are provided in the distribution package for convenience,

      ```
      wigToBigWig INPUT_WIG_GZ HG_SIZE BW_FILE
      ```

      Parameters:

      `INPUT_WIG_GZ`: The output fie from Step II-1a

      `HG_SIZE`: hg18.sizes or hg19.sizes (provided in the package)

      `BW_FILE`: The output fie file name

      Example:

      ```
      wigToBigWig test_data/Coverage/SJSIM001_D.wig.gz hg18.sizes test_data/Coverage/SJSIM001_D.bw
      ```

```
wigToBigWig test_data/Coverage/SJSIM001_G.wig.gz hg18.sizes
test_data/Coverage/SJSIM001_G.bw
```

2. Run the process_coverage.sh script to create read depth input file.

   ```
   sh code/process_coverage.sh BW SAMPLE DEST_DIR TAG SIZE
   ```

   Parameters:

   `BW`: the BigWig file from Step II-1.

   `SAMPLE`: Sample name

   `DEST_DIR`: Destination directory

   `TAG`: Identification tag. Default tags are D for tumor and G for normal. Alternative tags are accepted for tumor sample.

   `SIZE`: window size

   Example:

   ```
   sh code/process_coverage.sh test_data/Coverage/SJSIM001_D.bw
   SJSIM001 SJSIM001 D 100
   sh code/process_coverage.sh test_data/Coverage/SJSIM001_G.bw
   SJSIM001 SJSIM001 G 100
   ```

   After these two steps, input files SJSIM001_D_chr*_100 and SJSIM001_G_chr*_100 files are generated in the SJSIM001 folder.

3. (RECOMMENDED but OPTIONAL, paired analysis only) Generate the LOH input file:

   Note: This step is skipped for the test dataset included since there are no SNVs in this dataset.

   a. Run putative SNV detection in the paired sample by the variation detection module of Bambino [1]:

      ```
      code/snv_consterting.sh NIB_DIR D_BAM G_BAM OUTPUT
      ```

      Parameters:

      `NIB_DIR`: NIB directory or indexed fasta file (i.e. samtools faidx) for the reference genome

      `D_BAM`: Tumor BAM

      `G_BAM`: Germline BAM

      `OUTPUT`: Output file

      Example:

      ```
      code/snv_consterting.sh PATH/NIB BAMS/SAMPLE_D.bam
      BAMS/SAMPLE_G.bam SAMPLE_SNV.out
      ```

      In this example, the code will read the reference genome information from `PATH/NIB` and the tumor BAM from `BAMS/SAMPLE_D.bam` and germline BAM from `BAMS/SAMPLE_G.bam`. The result file is saved as `SAMPLE_SNV.out`.

b. Create the LOH input file

```
java -cp code/ AllelicImbalance -d DEST_DIR SAMPLE
SNV_OUTPUT
```

Parameters:

`DEST_DIR`: Destination directory, same as the DEST_DIR folder used in Step II-2

`SAMPLE`: Sample name.

`SNV_OUTPUT`: The SNV detection output form Step II-3a

Example:

```
java -cp code/ AllelicImbalance –d SAMPLE SAMPLE
SAMPLE_SNV.out
```

The code will create a `SAMPLE.ai` file in the `SAMPLE` folder.

4. (STRONGLY RECOMMENDED but OPTIONAL) WGS CREST analysis.
Please refer to the CREST README file for Steps II-4a – II-4c. This step could run in parallel with Step II-3.

Example:

a. Extract soft-clipped reads:

```
perl  extractSClip.pl --ref_genome hg18.fa -i SJSIM001_D-
CREST-2011-03-nq.chr22.bam -p SJSIM001_D
perl  extractSClip.pl --ref_genome hg18.fa -i SJSIM918_G-
WUHG18-nomut-nq.chr22.bam -p SJSIM001_G
```

Two files are generated from this step:

SJSIM001_D.cover

SJSIM001_G.cover

b. Identify somatic soft-clip position (Skipped for Single-BAM analysis):

```
perl countDiff.pl -d SJSIM001_D.cover -g SJSIM001_G.cover
```

SJSIM001_D.cover.somatic.cover is generated. Please copy this file to the directory with RD input files from Step II-2a.

c. SV detection:

```
perl CREST.pl --ref_genome hg18.fa --rmdup --cap3 cap3 --
blatclient gfClient --blat blat  --blatport 50000 -t
hg18.2bit -f SJSIM001_D.cover.somatic.cover -d SJSIM001_D-
CREST-2011-03-nq.chr22.bam -g SJSIM918_G-WUHG18-nomut-
nq.chr22.bam -max_rep_cover 2000 --max_germline_percent 0 -
min_hit_len 15 -m 2 --min_one_side_reads 3 -p SJSIM001_D
```

A file SJSIM001_D.predSV.txt is generated.

d. Reformat the CREST result for CONSERTING usage and save the final file in the directory with RD input files. Briefly, the optional header in CREST output is removed and the first 8 columns of the CREST result are retained, which contain breakpoint information used by CONSERTING.

```
grep -v posA SJSIM001/SJSIM001_D.predSV.txt | cut -f1-8 >
SJSIM001/SJSIM001.SV.lst
```

5. Run CONSERTING :

Note: 1) Please start from the code directory; 2) Please use the absolute path or relative path to the WD; 3) This step requires approximately 24G memory for a paired human sample analysis using a window size of 100 bp; 4) Please strictly follow the R syntax.

```
cd code
R CMD BATCH --no-save --no-restore '--args wd=WD sample=SAMPLE
gc.prefix=GC_DIR map.prefix=MAP_DIR crest.cmd=CREST_CMD
norm.chr=NORM_CHR D.alter=D_ALT singleBAM=SINGLEBAM
ai.file=AI_FILE' CONSERTING.R R_logfile
```

Required Parameters:

1) `WD`: working directory, where the read depth input, LOH input and SV input files are stored. Please use quotation marks.
2) `SAMPLE`: sample name (such as `"SJSIM001"`, please use quotation marks).
3) `GC_DIR`: directory for GC content files (from Step I-4). Please use quotation marks and include the trailing "/".
4) `MAP_DIR`: folder for mapability track files (from Step I-3). Please use quotation marks and include the trailing "/".

Selected optional parameters:

1) `CREST_CMD`: part of CREST command line with basic parameters (starting from beginning, up to the BAM setting). Please use quotation marks. If omitted, CONSERTING ignores any CREST input and runs a single iteration of segmentation only.
2) `NORM_CHR`: a set of pre-determined reference chromosomes (not used in the sample) in R format (example: `c(1,3:22)`). If omitted and the LOH input file is provided , CONSERTING will infer the reference. If both are omitted, CONSERTING takes all automsomes as reference. Please refer to Guideline_Manual_Ref_Selection.pdf for useful information on manual selection of reference chromosomes based on the coverage summary information.
3) `D_ALT`: Alternative tag for the tumor sample (from Step II-2). Please use quotation marks. If omitted, the default tags used are "D" for tumor and "G" for normal sample.
4) `SINGLEBAM`: Whether run paired analysis (`F`) or single BAM analysis (`T`). Default: F (paired analysis).

5) `AI_FILE`: the LOH input file from Step II-3 (Please use quotation marks). If no reference selection is given, CONSERTING will use the information from this file to infer the reference. This information is also used to refine the final result.

Example:

```
cd code
R CMD BATCH --no-save --no-restore '--args wd="../SJSIM001"
sample="SJSIM001" gc.prefix="../GC/hg18/"
map.prefix="../Mapability/hg18/" crest.cmd="perl CREST.pl --
ref_genome hg18.fa --rmdup --cap3 cap3 --blatclient gfClient --
blat blat  --blatport 50000 -t hg18.2bit -f
SJSIM001_D.cover.somatic.cover -d SJSIM001_D-CREST-2011-03-
nq.chr22.bam -g SJSIM918_G-WUHG18-nomut-nq.chr22.bam"'
CONSERTING.R SJSIM001.out
```

6. CONSERTING Output:

   Output file is stored in the `WD` /Result folder.

   Example:

   SJSIM001_CONSERTING_Mapability_100.txt.QualityMerge: The whole genome result.
   SJSIM001_CONSERTING_Mapability_100.txt.CNAcalls: Predicted somatic CNVs.

   When CONSERTING runs without the CREST iteration, only one result file is produced:
   SJSIM001_CONSERTING_NoCREST_Mapability_100.txt

Reference:

1. Edmonson, M.N., et al., *Bambino: a variant detector and alignment viewer for next-generation sequencing data in the SAM/BAM format.* Bioinformatics, 2011. **27**(6): p. 865-6.