TITAN: Subclonal copy number and LOH prediction from whole genome sequencing of tumours

Gavin Ha

April 26, 2022

Contents

1 Introduction

TITAN is a probabilistic framework for predicting regions of copy number alterations (LOH) and loss of heterozygosity (LOH) events in tumour whole genome sequencing (WGS) data. The model simultaneously estimates the cellular prevalence, proportion of tumour sample containing the event, and clonal cluster memberships. The statistical framework was designed based on the principles that the observed sequencing signal is an aggregated measure of multiple heterogeneous cellular populations including normal and tumour subpopulations and sets of genetic aberrations are observed at similar cellular prevalence if these events

co-occurred in the same clone due to punctuated clonal expansions [1]. The TITAN software is part of an accompanying manuscript where mathematical details are presented [2].

The observed measurements are modeled as generated from a composite of three cell populations [4], consisting normal cells at global proportion n; tumour cells with normal genotype at proportion $(1-n)*s_z$; and tumour cells with aberrant genotype at proportion $(1-n)*(1-s_z)$. s_z is the proportion of tumour cells that is diploid heterozygous (and therefore normal) at the locus. Thus, $(1-s_z)$ is the proportion of the tumour population containing the event, or we define as the cellular prevalence. We assume multiple somatic events share similar cellular prevalence and thus can be assigned to one of a finite number of clonal clusters, $z \in Z$.

The input data to TITAN are tumour read depth and allele counts. The read depth is corrected for GC content and mappability biases using the Bioconductor R package HMMcopy. Then, the log ratio between tumour and normal corrected depths is computed. The log ratio is model using a mixture of Gaussians were the mean parameter is a funtion of the cellular prevalence, normal proportion, and tumour ploidy; the variance is also unknown and estimated. The allele counts are transformed to allelic ratios, which is defined as the reference read count divided by depth or proportion of reference reads at a locus. Allelic ratios are modeled using a binomial distribution [3] with the success parameter being a function of cellular prevalence and normal contamination.

TITAN is implemented as a two-factor hidden Markov model (HMM) where the hidden genotypes and the hidden clonal cluster memberships are the two Markov chains. The state space is expanded as the joint genotype and clonal cluster states. The output of TITAN is a list of CNA and LOH events, with accompanying parameter estimates of normal proportion, the cellular prevalence and clonal population cluster membership for each event.

TITAN estimates the cellular prevalence (proportion of tumour cells with the tumour genotype) based on a fixed number of clonal clusters. Users are required to run TITAN on a tumour sample for a range of clonal clusters in parallel. We advise users to use a range of 1 to 5. When results for the 5 jobs are generated, one for each fixed number of clusters, each parameter output file will contain an S_Dbw Validity Index score. The run for a tumour sample that has the lowest S_Dbw score is the optimal result and the corresponding number of clonal clusters is the optimal one.

This vignette will detail an example run of TITAN for only 2 clonal clusters of chromosome 2 of a triple negative breast cancer sample [3, 2]. Once again, for real data, users should repeat the *TITAN* for additional clonal clusters and then select the optimal run using the minimum S_Dbw Validity index.

2 Analysis Workflow Overview

1. The data points of interest are the germline heterozygous SNP loci identified in the matched normal WGS sample. This is a pre-processing step that is completed by the user, and is outside of the scope of this R package.

2. At these loci, the read depth, reference read count and non-reference read count are extracted from the tumour WGS sample. This step is completed outside of the R package

3. The read depth are normalized for GC content and mappability biases using the *HMMcopy* R package. *TITAN* uses a wrapper for the function in *HMMcopy* to accomplish this.

4. *TITAN* requires as input, the read counts and normalized read depth to estimate model parameters and infer subclonal CNA/LOH.

5. For real data, repeat *TITAN* analysis for a range of clonal clusters.

3 Model training and parameter estimation

TITAN uses the Expectiation-Maximization algorithm to train the model and estimate parameters for cellular prevalence, normal proportion and tumour ploidy. In the E-step, the forwards-backwards algorithm is employed, making a call to a C implementation for faster computation. In the M-step, parameters are estimated using maximum a posteriori (MAP). The various parameters

TITAN uses the Viterbi algorithm to find the optimal state sequence path of genotype and clonal cluster membership for each data point.

4 Extracting tumour and normal read depth

TITAN requires the read depth from the tumour and normal WGS samples. This is done using a tool, outside of R, distributed as part of the *HMMcopy Suite* available at `http://compbio.bccrc.ca/software/hmmcopy/`.

In short, the suite has tools to:

- Obtain high resolution bin counts for large ($\approx$ 250GB) BAM files within a few hours.
- Obtain GC content for bins from standard FASTA files within minutes for a human genome.
- Obtain average mappability for bins from BigWig within minutes, or FASTA files within a day for a human genome.

For TITAN, the user will need to generate the read depth files for the tumour and normal samples in the form of WIG files. Also required, and is provided in the *HMMcopy suite* for the GRCh37-lite reference genome, are the GC content and mappability scores WIG files. Please refer to the instructions on the *HMMcopy suite* website for preparing these files.

5 TITAN analysis

Users should run TITAN once for each setting of the number of clonal clusters, ranging from 1 to 5. Although higher number of clonal clusters can be used, in practice, the sequencing coverage of 30-50X, analyzing more than 5 clonal clusters may not produce accurate results. For each run of TITAN on real data with approximately 2 million loci across all chromosomes and for 5 clonal clusters, the maximum memory requirement is approximately 24Gb.

5.1 Loading the data

Load the provided text file for the allele counts for chromosome 2 of a breast cancer sample [3]. The format of the file must be 6 columns: chromosome, position, reference base, reference read counts, non-reference base, non-reference read counts. A list object with 6 components of equal size/lengths is returned.

```
> library(TitanCNA)
> infile <- system.file("extdata", "test_alleleCounts_chr2.txt",
+                       package = "TitanCNA")
> data <- loadAlleleCounts(infile, genomeStyle = "NCBI")
> names(data)
```

```
[1] "chr"         "posn"        "ref"         "refOriginal" "nonRef"
[6] "tumDepth"
```

Users can specify the desired chromosome naming convention by using the `genomeStyle` argument. Using `NCBI` will use chromosome names such as 1, 2, 3, ..., X, while using `UCSC` will use names such as chr1, chr2, chr3, ..., chrX. Note that it does not matter what the chromosome convention was originally; using the `genomeStyle` argument will return the desired convention.

5.2 Correct GC content and mappability biases

Correct GC content and mappability biases using a wrapper function that calls uses a function similar to the *HMMcopy* package R function, *correctReadcount*. Here, 4 wig files are required: tumour, normal, GC content, and mappability score. The wrapper will apply the correction sub-routines and compute the log ratio as log2(tumour/normal).

```
> tumWig <- system.file("extdata", "test_tum_chr2.wig", package = "TitanCNA")
> normWig <- system.file("extdata", "test_norm_chr2.wig", package = "TitanCNA")
> gc <- system.file("extdata", "gc_chr2.wig", package = "TitanCNA")
> map <- system.file("extdata", "map_chr2.wig", package = "TitanCNA")
> cnData <- correctReadDepth(tumWig, normWig, gc, map, genomeStyle = "NCBI")
> head(cnData)
```

```
  chr start  end        logR
1   2     1 1000  1.18453687
2   2  1001 2000 -0.58498542
3   2  2001 3000 -0.10983809
4   2  3001 4000 -0.10433145
5   2  4001 5000  0.11298478
6   2  5001 6000  0.04131512
```

For real data, users can use the genome-wide GC content and mappability score for 1kb windows. The two files can be found compressed in data/GRCh37-lite.tar.gz of the git repository. These WIG files were generated from the reference GRCh37-lite.fasta; the tumour and normal BAM files used for generating the tumour and normal WIG files in the preprocessing steps MUST also use this same reference. Otherwise, new GC and map wig files will need to be created as per instructions on `http://compbio.bccrc.ca/software/hmmcopy/`

Then, find the log ratio at each position of interest (germline heterozygous SNPs) given in the object "data". Then transform the log ratios to natural logs instead of log base 2. Remove `logR` and `cnData` objects to save memory.

```
> logR <- getPositionOverlap(data$chr, data$posn, cnData)
> data$logR <- log(2^logR)  #transform the log ratio to natural logs
```

5.3 Filter the data

Filter the data for low and high depth positions and positions with NA's. For ease of analysis, TITAN converts chromosome X->23 and Y->24. Do not worry, in the final output, "X" and "Y" will be in the output. Optionally,

- `positionList`: data.frame containing a list of positions can be passed as an argument, specifying which positions to use.
- `centromeres`: data.frame with 3 columns - chr, start, stop that correspond to centromere regions. This argument can actually accept any regions that you wish to exclude.
- `centromere.flankLength`: the number of base pairs to the left and right of each region in `centromeres` that you which to further exclude.

```
> data <- filterData(data, c(1:22, "X", "Y"), minDepth = 10, maxDepth = 200,
+                    positionList = NULL, centromere = NULL, centromere.flankLength = 10000)
```

5.4 Loading the initial parameters

Load the default parameters using a specified maximum copy number *TITAN* will consider. Here, 5 is used but up to 8 copies can be specified, however, running time and memory performance hits will be incurred. Here, you will also specify the number of clonal clusters to use for the TITAN run. This is where the number of clusters will be specified when running for 1,2,3,4, or 5 clonal cluster for real data. In general, the default parameters will work well with real, full datasets of whole genome sequencing of tumour samples. Including the loaded `data` from the previous step will help inform the baseline allelic ratio parameters; this is recommended when using `symmetric` genotypes. `hetBaselineSkew` is the allelic reference skew for heterozygous states (e.g. 1:1, 2:2, 3:3). This value is the additive to baseline allelic ratios, for example, when `hetBaselineSkew=0.05`, then the heterozygous allelic ratio is expected to be 0.55. The `alleleEmissionModel` is specified as `"binomial"` or `"Gaussian"` (not fully implemented yet; please use binomial).

```
> numClusters <- 2
> params <- loadDefaultParameters(copyNumber = 5,
+                                 numberClonalClusters = numClusters,
+                                 symmetric = TRUE, hetBaselineSkew = 0,
+                                 alleleEmissionModel = "binomial", data = data)
> params

$genotypeParams
$genotypeParams$rt
 [1] 0.5000000 1.0000000 1.0000000 0.5000000 1.0000000 0.6666667 1.0000000
 [8] 0.7500000 0.5000000 1.0000000 0.8000000 0.6000000

$genotypeParams$rn
[1] 0.5

$genotypeParams$ZS
 [1]  0  1  2 -1  4  5  6  7  8  9 10 11

$genotypeParams$ct
 [1] 0 1 2 2 3 3 4 4 4 5 5 5

$genotypeParams$corRho_0
[1] -0.12784

$genotypeParams$var_0
 [1] 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05

$genotypeParams$alphaKHyper
 [1] 760.2561 760.2561 760.2561 760.2561 760.2561 760.2561 760.2561 760.2561
 [9] 760.2561 760.2561 760.2561 760.2561

$genotypeParams$betaKHyper
 [1] 25 25 25 25 25 25 25 25 25 25 25 25

$genotypeParams$alleleEmissionModel
[1] "binomial"

$genotypeParams$varR_0
 [1] 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05 0.05
```

```
$genotypeParams$alphaRHyper
 [1] 1376.962 1376.962 1376.962 1376.962 1376.962 1376.962 1376.962 1376.962
 [9] 1376.962 1376.962 1376.962 1376.962

$genotypeParams$betaRHyper
 [1] 25 25 25 25 25 25 25 25 25 25 25 25

$genotypeParams$kappaGHyper
 [1]   2 101 101 500 101 101 101 101 500 101 101 101

$genotypeParams$piG_0
 [1] 0.0005265929 0.0526592944 0.0526592944 0.2627698789 0.0526592944
 [6] 0.0526592944 0.0526592944 0.0526592944 0.2627698789 0.0526592944
[11] 0.0526592944 0.0526592944

$genotypeParams$outlierVar
[1] 10000

$genotypeParams$symmetric
[1] TRUE


$ploidyParams
$ploidyParams$phi_0
[1] 2

$ploidyParams$alphaPHyper
[1] 20

$ploidyParams$betaPHyper
[1] 42


$normalParams
$normalParams$n_0
[1] 0.5

$normalParams$alphaNHyper
[1] 2

$normalParams$betaNHyper
[1] 2


$cellPrevParams
$cellPrevParams$s_0
[1] 0.001 0.200

$cellPrevParams$alphaSHyper
[1] 2 2
```

```
$cellPrevParams$betaSHyper
[1] 2 2

$cellPrevParams$kappaZHyper
[1] 2 2

$cellPrevParams$piZ_0
[1] 0.5 0.5
```

`params` is a list object containing 4 sets of parameters, each as a component:

- `genotypeParams`: Parameters for copy number and allelic ratios genotype states
- `normalParams`: Parameters for normal contamination
- `ploidyParams`: Parameters for average tumour ploidy
- `cellPrevParams`: Parameters for modeling subclonality: clonal clusters and cellular prevalence
- `data`: Input data frame to help determine the prior diploid heterozygous baseline allelic ratio noise

Analysis of copy number by *TITAN* is subject to tumour ploidy. TITAN attempts to estimate this value globally and is accurate in most cases. However, the issue is that, from the signals observed in the data, *TITAN* cannot distinguish between diploid (2 global copies) and tetraploidy (4 global copies) because the data can explain both situations. Therefore, in general, if the user is aware that a sample is ploypoid, then TITAN should be run with the ploidy initialization of 2 and also for initialization of 4. If the user is unaware of the ploidy status of the sample, then inspection of a TITAN run at ploidy initialization of 2 can help. If this run has many large, prominent regions of inferred homozygous deletions, then it is likely that this sample is triploid or tetraploid.

```
> params$ploidyParams$phi_0 <- 2 # for diploid or
> params$ploidyParams$phi_0 <- 4 # for tetraploid/ploypoid
```

Because this vignette is an example involving only one chromosome, the fewer loci in the analysis will give different results than for a sample with all chromosomes. For this example, we will modify the Gaussian variance hyperparameter (prior used in modeling the log ratios) to a less influential setting, making the analysis more suitable for one chromosome. Also, because we know the average tumour ploidy of the genome-wide sample is slightly less than 2, we will initialize the `phi_0` to 1.5 so that it does not overfit to this one chromosome.

```
> K <- length(params$genotypeParams$alphaKHyper)
> params$genotypeParams$alphaKHyper <- rep(500, K)
> params$ploidyParams$phi_0 <- 1.5
```

5.5 Model training and parameter estimation

Parameter estimation in *TITAN* uses the Expectation Maximization algorithm (EM) and forwards-backwards algorithm. This step generally requires parallelization to increase time performance. The *TITAN* package uses the *foreach* package which conveniently enables the user to use libraries, such as *doMC* and *doMPI*, to parallelize the training by chromosome. It is up to the user to decide which is the best library to use depending on the current system. For example, if one wishes to use multiple cores via forking on a single machine,

```
> library(doMC)
> registerDoMC(cores = 4) #use 4 cores on a single machine
```

Here is a little more detail regarding some important arguments:

- `maxiter`: maximum number of EM iterations allowed. In practice, users should set it at 20 since it does not exceed 20. For this vignette to finish in a shorter time, we use 3.
- `maxiterUpdate`: maximum number of coordinate descent iterations during the M-step when parameters are estimated. In practice, 1500 iterations should be used.
- `txnExpLen`: influences prior probability of genotype transitions in the HMM. The higher, the lower tendency to change state.
- `txnZstrength`: influences prior probability of clonal cluster transitions in the HMM. Larger values means lower tendency to change clonal cluster state.
- `useOutlierState`: logical indicating whether an additional outlier state should be used. In practice, this usually is not necessary.
- `normalEstimateMethod`: specifies how to handle normal proportion estimation. To estimate, use "map" which is maximum a posteriori. If you wish to not estimate this parameter, then use "fixed". This will default the normal proportion to whatever is specified in `params$normalParams$n_0`. For example, if you know that this sample has absolutely no normal contamination, then use `params$normalParams$n_0 <- 0` and set `normalEstimateMethod="fixed"`.
- `estimateS`: logical indicating whether to account for clonality and estimate subclonal events
- `estimatePloidy`: logical indicating whether to estimate and account for tumour ploidy

```
> convergeParams <- runEMclonalCN(data, params,
+                                 maxiter = 3, maxiterUpdate = 50,
+                                 useOutlierState = FALSE, txnExpLen = 1e15,
+                                 txnZstrength = 5e5,
+                                 normalEstimateMethod = "map",
+                                 estimateS = TRUE, estimatePloidy = TRUE)
> names(convergeParams)
```

```
 [1] "n"               "s"               "var"             "varR"
 [5] "phi"             "piG"             "piZ"             "muR"
 [9] "muC"             "loglik"          "rhoG"            "rhoZ"
[13] "txnExpLen"       "txnZstrength"    "useOutlierState" "genotypeParams"
[17] "ploidyParams"    "normalParams"    "cellPrevParams"  "symmetric"
```

`convergeParams` is a list object with components containing the converged parameters from the EM training, including posterior marginal responsibilities, log likelihood, and original parameter settings.

- `n`: Converged estimate for normal contamination parameter. `numeric array` containing estimates at each EM iteration.
- `s`: Converged estimate(s) for cellular prevalence parameter(s). This value is defined as the proportion of tumour sample that does *not* contain the aberrant genotype. This will contrast what is output in `outputTitanResults`. `numeric array` containing estimates at each EM iteration. If more than one cluster is specified, then `s` is a `numeric matrix`.
- `var`: Converged estimates for variance parameter of the Gaussian mixtures used to model the log ratio data. `numeric matrix` containing estimates at each EM iteration.
- `phi`: Converged estimate for tumour ploidy parameter. `numeric array` containing estimates at each EM iteration.

- `piG`: Converged estimate for initial genotype state distribution. `numeric matrix` containing estimates at each EM iteration.
- `piZ`: Converged estimate for initial clonal cluster state distribution. `numeric matrix` containing estimates at each EM iteration.
- `muR`: Mean of binomial mixtures computed as a function of `s` and `n`. `numeric matrix` containing estimates at each EM iteration. See References for mathematical details.
- `muC`: Mean of Gaussian mixtures computed as a function of `s`, `n`, and `phi`. `numeric matrix` containing estimates at each EM iteration. See References for mathematical details.
- `loglik`: Posterior Log-likelihood that includes data likelihood and the priors. `numeric array` containing estimates at each EM iteration.

5.6 Find the genotype and clonal cluster state paths

Viterbi algorithm is used to return the optimal genotype/clonal cluster state path. After running EM, use the converge parameters and the input data to compute the optimal segmentation results.

```
> optimalPath <- viterbiClonalCN(data, convergeParams)
> head(optimalPath)

[1] 14 14 14 14 14 14
```

It is difficult to interpret the output of this function directly. The user should use the function `outputTitanResults` to format the results.

5.7 Format and print results to file

5.7.1 Position-specific Copy Number Results

Position-specific results that includes genotype, clonal cluster, and cellular prevalence estimates. The posterior probabilities at each position can optionally be returned. The results can be output to a file if a file name is specified for the `filename` argument.

```
> results <- outputTitanResults(data, convergeParams, optimalPath,
+                               filename = NULL, posteriorProbs = FALSE,
+                               subcloneProfiles = TRUE, correctResults = TRUE,
+                               proportionThreshold = 0.05,
+                               proportionThresholdClonal = 0.05,
+                               is.haplotypeData = FALSE)
> names(results)

[1] "results"        "corrResults"    "convergeParams"

> head(results$corrResults) ## corrected results

   Chr Position RefCount Depth AllelicRatio    LogRatio CopyNumber TITANstate
1:   2    55802       13    25    0.5200000  0.12709729          1          1
2:   2    56232        3    10    0.3000000  0.01820724          1          1
3:   2    56619        7    13    0.5384615  0.01820724          1          1
4:   2    57093       10    12    0.8333333 -0.41401567          1          1
5:   2    60324        6    16    0.3750000 -0.03516920          1          1
6:   2    60906        6    11    0.5454545 -0.03516920          1          1
```

```
   TITANcall ClonalCluster CellularPrevalence Subclone1.CopyNumber
1:      DLOH             2          0.5510502                    2
2:      DLOH             2          0.5510502                    2
3:      DLOH             2          0.5510502                    2
4:      DLOH             2          0.5510502                    2
5:      DLOH             2          0.5510502                    2
6:      DLOH             2          0.5510502                    2
   Subclone1.TITANcall Subclone1.Prevalence Subclone2.CopyNumber
1:                 HET            0.4489498                    1
2:                 HET            0.4489498                    1
3:                 HET            0.4489498                    1
4:                 HET            0.4489498                    1
5:                 HET            0.4489498                    1
6:                 HET            0.4489498                    1
   Subclone2.TITANcall Subclone2.Prevalence
1:                DLOH            0.5510502
2:                DLOH            0.5510502
3:                DLOH            0.5510502
4:                DLOH            0.5510502
5:                DLOH            0.5510502
6:                DLOH            0.5510502
```

```
> convergeParams <- results$convergeParam ## use corrected parameters
> results <- results$corrResults ## use corrected results
```

The elements of this list contains the following:

1. `results`: TITAN results, uncorrected for cluster number and parameters
2. `corrResults`: TITAN results, corrected by removing empty clusters and parameters adjusted accordingly.
3. `convergeParams`: Corrected parameter object

When `correctResults=TRUE`, the results will be post-processed to exclude clonal clusters that have fewer genomic alterations. Users can specify the minimum proportion of the genome altered for any subclonal cluster using `proportionThreshold` and the minimum proportion of clonal events for the clonal (highest cellular prevalence) cluster using `proportionThresholdClonal`. The function returns the post-processed `convergeParams` and `corrResults` objects in a 2 element list.

`results` and `CorrResults` have the following format:

1. `Chr`
2. `Position`
3. `RefCount`: number of reads matching the reference base
4. `NRefCount`: number of reads matching the non-reference base
5. `Depth`: total read depth at the position
6. `AllelicRatio`: RefCount/Depth
7. `LogRatio`: log2 ratio between normalized tumour and normal read depths
8. `CopyNumber`: predicted TITAN copy number

9. `TITANstate`: internal state number used by TITAN; see supplementary table 2 in manuscript
10. `TITANcall`: interpretable TITAN state; string {HOMD,DLOH,HET,NLOH,ALOH,ASCNA,BCNA,UBCNA}, see supplementary table 2 in manuscript
11. `ClonalCluster`: predicted TITAN clonal cluster; lower cluster numbers represent clusters with higher cellular prevalence
12. `CellularPrevalence`: proportion of tumour cells containing event; not to be mistaken as proportion of sample (including normal)
13. `Subclone1.CopyNumber`: Copy number profile for Subclone 1
14. `Subclone1.TITANcall`: TITAN state for Subclone 1
15. `Subclone1.Prevalence`: Subclonal prevalence for Subclone 1
16. `Subclone2.CopyNumber`: Copy number profile for Subclone 2
17. `Subclone2.TITANcall`: TITAN state for Subclone 2
18. `Subclone2.Prevalence`: Subclonal prevalence for Subclone 2

5.7.2 Copy number segments

Since version 1.10.1, users can directly generate segments and output these to files. There are two output files: 1) Segments with detailed information, and 2) Segments compatible for loading into the Integrative Genomics Viewer (IGV) application. These filenames can be specified in arguments `filename` and `igvfilename`

```
> segs <- outputTitanSegments(results, id = "test", convergeParams,
+                             filename = NULL, igvfilename = NULL)
> head(segs)
```

```
   Sample Chromosome Start_Position.bp. End_Position.bp. Length.snp.
1:   test          2              55802         37796810        2617
2:   test          2           37801146         37801146           1
3:   test          2           37811950         37875052          74
4:   test          2           37887950         37887950           1
5:   test          2           37893503         59989919        1789
6:   test          2           60088960         72371178         903
   Median_Ratio Median_logR TITAN_state TITAN_call Copy_Number MinorCN MajorCN
1:     0.666667   -0.019281           1       DLOH           1       0       1
2:     0.684211    0.075778           3        HET           2       1       1
3:     0.891813   -0.242090           1       DLOH           1       0       1
4:     0.565217   -0.020486           3        HET           2       1       1
5:     0.650000    0.013133           1       DLOH           1       0       1
6:     0.791667    0.543601           4       ALOH           3       0       3
   Clonal_Cluster Cellular_Prevalence
1:              2           0.5510502
2:             NA                  NA
3:              1           1.0000000
4:             NA                  NA
5:              2           0.5510502
6:              2           0.5510502
```

The definitions of the columns of the segment object `segs` are the following:

1. `Sample`: Name of sample
2. `Chromosome`, `Start_Position.bp.`, `End_Position.bp.`: Coordinates of segment
3. `Length.snps.`: Number of SNPs in the segment
4. `Median_Ratio`: Median allelic ratio across SNPs in the segment
5. `Median_logR`: Median log ratio across SNPs in the segment
6. `TITAN_state`: Same as defined above
7. `TITAN_call`: Same as defined above
8. `Copy_Number`: Same as defined above
9. `MinorCN`: Copy number of minor allele
10. `MajorCN`: Copy number of major allele
11. `Clonal_Cluster`: Same as defined above
12. `Cellular_Frequency`: Same as defined above

5.7.3 Adjusted Copy Number Results

New in version 1.17.1, copy number results can be further adjusted to reflect total integer and allelic copy number by correcting the HMM copy number states. The HMM normally supports a maximum copy number value of 8, but this correction will output an adjusted copy number based on the log ratio values.

Users can specify the initial copy number state to correct using `maxCNtoCorrect.autosomes`. For example, if `maxCNtoCorrect.autosomes=8`, then all positions and segments with copy number `>=8` will be corrected. If `correctHOMD=TRUE`, then homozygous deletion (copy number of 0) calls will be adjusted to account for potential label-switching (i.e. missed HOMD call) and false positives (i.e. incorrect HOMD call). Finally, users can specify the minimum tumor content required to performed the tumor purity and ploidy correction. Samples with lower tumor conent will likely have noisier adjusted values since tumor purity is used in the correction.

The correction is cmoputed based on the following formula from Ref[2] using observed log ratio l_t at position or segment t and tumor purity $(1-n)$ and tumor ploidy ϕ:

$$l_t = \log\left(\frac{nc_N + (1-n)\, s_z c_N + (1-n)\,(1-s_z)\, c_t}{nc_N + (1-n)\,\phi}\right) \tag{1}$$

$$\hat{c_T} = \frac{2^{l_T}\left[nc_N + (1-n)\,\phi\right] - nc_N + (1-n)\, s_z c_N}{(1-n)\,(1-s_z)} \tag{2}$$

where $\hat{c_t}$ is the adjusted tumor copy number at position or segment t.

```
> # get the estimated tumor ploidy
> ploidy <- tail(convergeParams$phi, 1)
> # get the estimated normal
> normal <- tail(convergeParams$n, 1)
> # apply tumor purity and ploidy correction
> corrIntCN.results <- correctIntegerCN(results, segs, 1 - normal, ploidy,
+                                       maxCNtoCorrect.autosomes = 8,
+                                       maxCNtoCorrect.X = NULL,
+                                       correctHOMD = FALSE, minPurityToCorrect = 0.2,
+                                       gender = "female", chrs = 2)
> head(corrIntCN.results$segs)
```

```
   Sample Chromosome Start_Position.bp. End_Position.bp. Length.snp.
1:   test          2              55802         37796810        2617
2:   test          2           37801146         37801146           1
3:   test          2           37811950         37875052          74
4:   test          2           37887950         37887950           1
5:   test          2           37893503         59989919        1789
6:   test          2           60088960         72371178         903
   Median_Ratio Median_logR TITAN_state TITAN_call Copy_Number MinorCN MajorCN
1:     0.666667   -0.019281           1       DLOH           1       0       1
2:     0.684211    0.075778           3        HET           2       1       1
3:     0.891813   -0.242090           1       DLOH           1       0       1
4:     0.565217   -0.020486           3        HET           2       1       1
5:     0.650000    0.013133           1       DLOH           1       0       1
6:     0.791667    0.543601           4       ALOH           3       0       3
   Clonal_Cluster Cellular_Prevalence logR_Copy_Number Corrected_Ratio
1:              2           0.5510502         1.013940       1.0000000
2:             NA                  NA         1.593573       0.7482512
3:              1           1.0000000         1.168901       1.0000000
4:             NA                  NA         1.454953       0.5900495
5:              2           0.5510502         1.096846       1.0000000
6:              2           0.5510502         2.755174       1.0000000
   Corrected_Copy_Number Corrected_Call Corrected_MajorCN Corrected_MinorCN
1:                     1           HETD                 1                 0
2:                     2           NEUT                 1                 1
3:                     1           HETD                 1                 0
4:                     2           NEUT                 1                 1
5:                     1           HETD                 1                 0
6:                     3           GAIN                 3                 0
```

```
> # re-assign to results and segs objects
> results <- corrIntCN.results$cn
> segs <- corrIntCN.results$segs
```

Both position-level (`results`) and segment-level (`segs`) results will have additional columns appended for the adjusted copy number results. The additional columns are the following:

1. `logR_Copy_Number`: Purity and ploidy corrected log ratios that have been converted to a decimal-based copy number value.
2. `Corrected_Copy_Number`: Purity and ploidy corrected total copy number rounded to the nearest integer.
3. `Corrected_Call`: Copy number status of the total corrected copy number.
4. `Corrected_MajorCN`: Purity and ploidy corrected integer (rounded) major copy number value.
5. `Corrected_MinorCN`: Purity and ploidy corrected integer (rounded) minor copy number value.

5.7.4 Estimated Model Parameters

Model parameters and summary values such as the number of clonal clusters and their corresponding cellular prevalence, normal contamination, ploidy, and the S_Dbw validity index for model selection later.

```
> outparam <- paste("test_cluster02_params.txt", sep = "")
> outputModelParameters(convergeParams, results, outparam, S_Dbw.scale = 1)
```

The format of the parameter output file is the following:

1. Normal contamination estimate: proportion of normal content in the sample; tumour content is 1 minus this number
2. Average tumour ploidy estimate: average number of estimated copies in the genome; 2 represents diploid
3. Clonal cluster cellular prevalence: Z denotes the number of clonal clusters; each value (space-delimited) following are the cellular prevalence estimates for each cluster
4. Genotype binomial means for clonal cluster Z: set of 21 binomial estimated parameters for each specified cluster
5. Genotype Gaussian means for clonal cluster Z: set of 21 Gaussian estimated means for each specified cluster
6. Genotype Gaussian variance: set of 21 Gaussian estimated variances; variances are shared for across all clusters
7. Number of iterations: number of EM iterations needed for convergence
8. Log likelihood: complete data log-likelihood for current cluster run
9. S_Dbw dens.bw: density component of S_Dbw index
10. S_Dbw scat: scatter component of S_Dbw index
11. S_Dbw validity index: used for model selection; choose run with optimal number of clusters based on lowest S_Dbw index

Users may alter the `S_Dbw.scale` argument to penalize higher number of clonal clusters.

```
> outputModelParameters(convergeParams, results, outparam, S_Dbw.scale = 10)
```

5.8 Plotting the results

Plot the results using 3 built in functions. For each chromosome, there is a separate plot function for the log ratios (CNA), allelic ratios (LOH), and cellular prevalence. Each function adds to an existing plot.

5.8.1 Copy number alterations (log ratio)

Generate the figure of copy number alteration results by plotting the log ratios. Optionally, the data can be corrected by using the estimated `ploidy`. Otherwise, use `ploidy=NULL`.

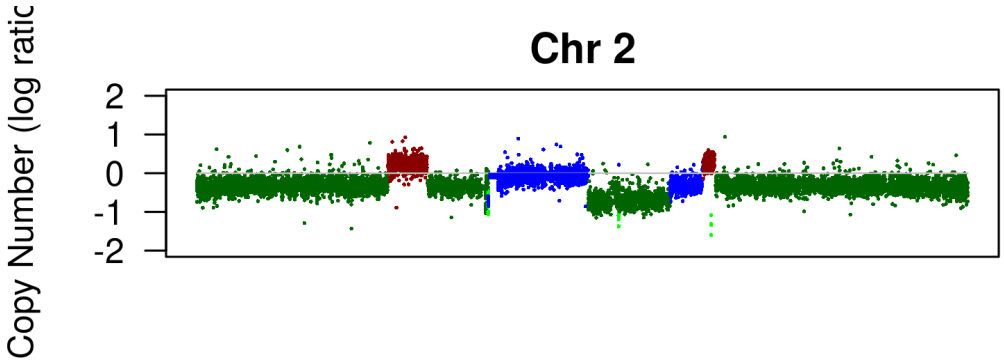
```
> ploidy <- tail(convergeParams$phi, 1)
> ploidy

[1] 1.483683

> normal <- tail(convergeParams$n, 1)
> normal

[1] 0.2169144

> plotCNlogRByChr(results, segs = segs, chr = 2, ploidy = ploidy, normal = normal,
+                 ylim = c(-2, 2), cex = 0.25, xlab = "", main = "Chr 2")
```

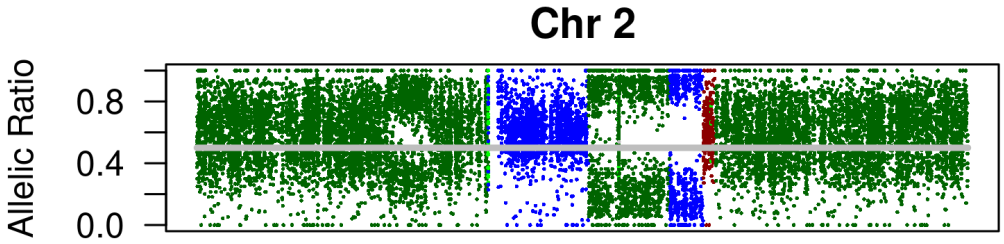


The Y-axis is based on log ratios. Log ratios are computed ratios between normalized tumour and normal read depths. Data points close to 0 represent diploid, above 0 are copy gains, below 0 are deletions. Bright Green - HOMD Green - DLOH Blue - HET, NLOH Dark Red - GAIN Red - ASCNA, UBCNA, BCNA

5.8.2 Loss of heterozygosity (allelic ratio)

The loss of heterozygosity (LOH) results is shown by plotting the allelic ratio.

```
> plotAllelicRatio(results, chr = 2, ylim = c(0, 1), cex = 0.25,
+                  xlab = "", main = "Chr 2")
```



The Y-axis is based on allelic ratios. Allelic ratios are computed as RefCount/Depth. Data points close to 1 represent homozygous reference base, close to 0 represent homozygous non-reference base, and close to 0.5 represent heterozygous. Normal contamination influences the divergence away from 0.5 for LOH events. Grey - HET, BCNA Bright Green - HOMD Green - DLOH, ALOH Blue - NLOH Dark Red - GAIN Red - ASCNA, UBCNA

5.8.3 Cellular prevalence and clonal clusters

One of the key features of *TITAN* is the estimation of cellular prevalence and the inference of clonal cluster membership. The estimated normal proportion is required for this plot. This can be obtained from `convergeParams`.

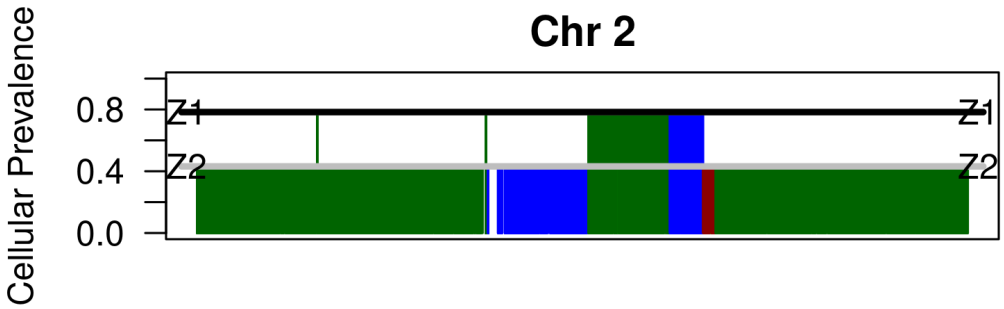
```
> norm <- tail(convergeParams$n, 1)
> norm # estimated normal contamination
```

```
[1] 0.2169144
```

```
> 1 - convergeParams$s[, ncol(convergeParams$s)] # estimated cellular prevalence

[1] 1.0000000 0.5510502

> plotClonalFrequency(results, chr = 2, normal = norm, ylim = c(0, 1),
+                     cex = 0.25, xlab = "", main = "Chr 2")
```

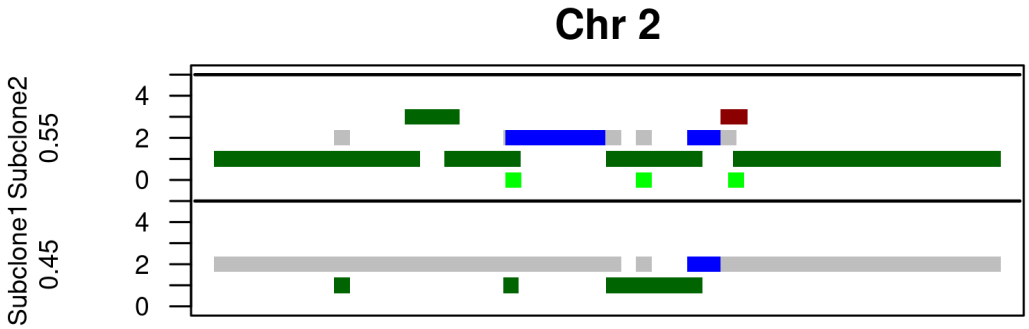


The Y-axis is the cellular prevalence that includes the normal proportion. Therefore, the cellular prevalence here refers to the proportion in the sample (including normal). Lines are drawn for each data point indicating the cellular prevalence. Heterozygous diploid are not shown because it is a normal genotype and is not categorized as being subclonal (this means 100% of cells are normal). The black horizontal line represents the tumour content labeled as "T". Each horizontal grey line represents the cellular prevalence of the clonal clusters labeled as Z1, Z2, etc. Colours are the sames for allelic ratio plots.

5.8.4 Subclone profiles

New since TitanCNA v1.2.0, users can plot the copy number profiles for the predicted subclones. This function only works for solutions containing 1 or 2 clonal clusters.

```
> plotSubcloneProfiles(results, chr = 2, cex = 1, spacing = 2, main = "Chr 2")
```

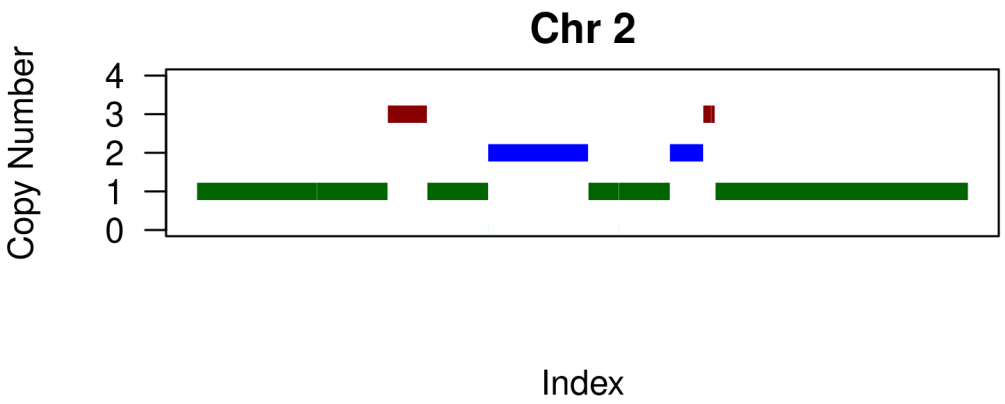


Colours have the same definition as for the allelic ratio plots.

5.8.5 Segment medians

For a cleaner visualization of copy number results, users can plot segment means of total copy number and allelic copy number results

```
> plotSegmentMedians(segs, chr=2, resultType = "LogRatio", plotType = "CopyNumber",
+                    plot.new = TRUE, ylim = c(0, 4), main="Chr 2")
```



6 Session Information

The version number of R and packages loaded for generating the vignette

- R version 4.2.0 RC (2022-04-19 r82224), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_GB`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Running under: `Ubuntu 20.04.4 LTS`
- Matrix products: default
- BLAS: `/home/biocbuild/bbs-3.15-bioc/R/lib/libRblas.so`
- LAPACK: `/home/biocbuild/bbs-3.15-bioc/R/lib/libRlapack.so`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: TitanCNA 1.34.0
- Loaded via a namespace (and not attached): AnnotationDbi 1.58.0, BSgenome 1.64.0, Biobase 2.56.0, BiocFileCache 2.4.0, BiocGenerics 0.42.0, BiocIO 1.6.0, BiocParallel 1.30.0, Biostrings 2.64.0, DBI 1.1.2, DelayedArray 0.22.0, GenomeInfoDb 1.32.0, GenomeInfoDbData 1.2.8, GenomicAlignments 1.32.0, GenomicFeatures 1.48.0, GenomicRanges 1.48.0, IRanges 2.30.0, KEGGREST 1.36.0, Matrix 1.4-1, MatrixGenerics 1.8.0, R6 2.5.1, RCurl 1.98-1.6, RSQLite 2.2.12, Rcpp 1.0.8.3, Rsamtools 2.12.0, S4Vectors 0.34.0, SummarizedExperiment 1.26.0, VariantAnnotation 1.42.0, XML 3.99-0.9, XVector 0.36.0, assertthat 0.2.1, biomaRt 2.52.0, bit 4.0.4, bit64 4.0.5, bitops 1.0-7, blob 1.2.3, cachem 1.0.6, cli 3.3.0, codetools 0.2-18, compiler 4.2.0, crayon 1.5.1, curl 4.3.2, data.table 1.14.2, dbplyr 2.1.1, digest 0.6.29, dplyr 1.0.8, ellipsis 0.3.2, fansi 1.0.3, fastmap 1.1.0, filelock 1.0.2, foreach 1.5.2, generics 0.1.2, glue 1.6.2, grid 4.2.0, hms 1.1.1, httr 1.4.2, iterators 1.0.14, lattice 0.20-45, lifecycle 1.0.1, magrittr 2.0.3, matrixStats 0.62.0, memoise 2.0.1, parallel 4.2.0, pillar 1.7.0, pkgconfig 2.0.3, png 0.1-7, prettyunits 1.1.1, progress 1.2.2, purrr 0.3.4, rappdirs 0.3.3, restfulr 0.0.13, rjson 0.2.21, rlang 1.0.2, rtracklayer 1.56.0, stats4 4.2.0, stringi 1.7.6, stringr 1.4.0, tibble 3.1.6, tidyselect 1.1.2, tools 4.2.0, utf8 1.2.2, vctrs 0.4.1, xml2 1.3.3, yaml 2.3.5, zlibbioc 1.42.0

References

[1] Mel Greaves and Carlo C Maley. Clonal evolution in cancer. *Nature*, 481(7381):306–313, Jan 2012.

[2] Gavin Ha, Andrew Roth, Jaswinder Khattra, Julie Ho, Damian Yap, Leah M Prentice, Nataliya Melnyk, Andrew McPherson, Ali Bashashati, Emma Laks, Justina Biele, Jiarui Ding, Alan Le, Jamie Rosner, Karey Shumansky, Marco a Marra, C Blake Gilks, David G Huntsman, Jessica N McAlpine, Samuel Aparicio, and Sohrab P Shah. TITAN: inference of copy number architectures in clonal cell populations from tumor whole-genome sequence data. *Genome research*, pages 1881–1893, July 2014.

[3] Gavin Ha, Andrew Roth, Daniel Lai, Ali Bashashati, Jiarui Ding, Rodrigo Goya, Ryan Giuliany, Jamie Rosner, Arusha Oloumi, Karey Shumansky, Suet-Feung Chin, Gulisa Turashvili, Martin Hirst, Carlos Caldas, Marco A. Marra, Samuel Aparicio, and Sohrab P. Shah. Integrative analysis of genome-wide loss of heterozygosity and monoallelic expression at nucleotide resolution reveals disrupted pathways in triple-negative breast cancer. *Genome research*, 22(10):1995–2007, Oct 2012.

[4] C Yau, D Mouradov, R N Jorissen, S Colella, G Mirza, G Steers, A Harris, J Ragoussis, O Sieber, and C C Holmes. A statistical approach for detecting genomic aberrations in heterogeneous tumor samples from single nucleotide polymorphism genotyping data. *Genome Biol*, 11(9), 2010.