# Package: calmate (via r-universe)

August 24, 2024

**Version** 0.13.0

**Depends** R (>= 3.2.1), R.utils (>= 2.11.0), aroma.core (>= 3.2.2)

**Imports** utils, MASS, R.methodsS3 (>= 1.8.1), R.oo (>= 1.24.0), matrixStats (>= 0.61.0), R.filesets (>= 2.14.0)

**Suggests** DNAcopy

**Title** Improved Allele-Specific Copy Number of SNP Microarrays for Downstream Segmentation

**Description** The CalMaTe method calibrates preprocessed allele-specific copy number estimates (ASCNs) from DNA microarrays by controlling for single-nucleotide polymorphism-specific allelic crosstalk. The resulting ASCNs are on average more accurate, which increases the power of segmentation methods for detecting changes between copy number states in tumor studies including copy neutral loss of heterozygosity. CalMaTe applies to any ASCNs regardless of preprocessing method and microarray technology, e.g. Affymetrix and Illumina.

**License** LGPL (>= 2.1)

**URL** https://github.com/HenrikBengtsson/calmate/

**BugReports** https://github.com/HenrikBengtsson/calmate/issues

**LazyLoad** TRUE

**biocViews** aCGH, CopyNumberVariants, SNP, Microarray, OneChannel, TwoChannel, Genetics

**Repository** https://henrikbengtsson.r-universe.dev

**RemoteUrl** https://github.com/HenrikBengtsson/calmate

**RemoteRef** master

**RemoteSha** e1ce582ed1a5c09a177ad53170c957b625bec546

## Contents

---

`calmate-package` *Package calmate*

---

## Description

The CalMaTe method calibrates preprocessed allele-specific copy number estimates (ASCNs) from DNA microarrays by controlling for single-nucleotide polymorphism-specific allelic crosstalk. The resulting ASCNs are on average more accurate, which increases the power of segmentation methods for detecting changes between copy number states in tumor studies including copy neutral loss of heterozygosity. CalMaTe applies to any ASCNs regardless of preprocessing method and microarray technology, e.g. Affymetrix and Illumina.

## Requirements

This package depends on a set of packages that are all available via CRAN. It has been tested and verified to run on all common operating systems on which R runs, including Linux, Windows and OSX.

## Installation and updates

To install this package, do `install.packages("calmate")`.

## To get started

1. To process SNP and non-polymorphic signals, see `calmateByTotalAndFracB`(). If you are working solely with SNP signals, `calmateByThetaAB`() is also available, but we recommend the former.
2. For processing data in the aroma framework, see `CalMaTeCalibration`.

## How to cite

Please cite [1] when using CalMaTe.

## License

LGPL (>= 2.1).

## Author(s)

Maria Ortiz [aut, ctb], Ander Aramburu [ctb], Henrik Bengtsson [aut, cre, cph], Pierre Neuvial [aut, ctb], Angel Rubio [aut, ctb].

## References

[1] M. Ortiz-Estevez, A. Aramburu, H. Bengtsson, P. Neuvial and A. Rubio, *CalMaTe: A method and software to improve allele-specific copy number of SNP arrays for downstream segmentation*, Bioinformatics, 2012 [PMC3381965].

---

`calmateByThetaAB.array` *Normalize allele-specific copy numbers (CA,CB)*

---

## Description

Normalize allele-specific copy numbers (CA,CB).

## Usage

```
## S3 method for class 'array'
calmateByThetaAB(data, references=NULL, ..., truncate=FALSE, refAvgFcn=NULL,
  flavor=c("v2", "v1"), verbose=FALSE)
```

## Arguments

| | |
|---|---|
| `data` | An Jx2xI `numeric array`, where J is the number of SNPs, 2 is the number of alleles, and I is the number of samples. |
| `references` | An index `vector` in [1,I] or a `logical vector` of length I specifying which samples are used when calculating the reference signals. If `NULL`, all samples are used. At least 3 samples. |
| `...` | Additional arguments passed to the internal fit function `fitCalMaTeInternal`. |
| `truncate` | If `TRUE`, final ASCNs are forced to be non-negative while preserving the total CNs. |
| `refAvgFcn` | (optional) A `function` that takes a JxI `numeric matrix` an argument `na.rm` and returns a `numeric vector` of length J. It should calculate some type of average for each of the J rows, e.g. `rowMedians`. If specified, then the total copy numbers of the calibrated ASCNs are standardized toward (twice) the average of the total copy numbers of the calibrated reference ASCNs. |
| `flavor` | A `character` string specifying which flavor of the CalMaTe algorithm to use for fitting the model. |
| `verbose` | See `Verbose`. |

## Value

Returns an Jx2xI `numeric array` with the same dimension names as argument `data`.

## Flavors

For backward compatibility, we try to keep all major versions of the CalMaTe algorithm available. Older versions can be used by specifying argument `flavor`. The default flavor is `v2`. For more information about the different flavors, see `fitCalMaTeInternal`.

## References

[1] M. Ortiz-Estevez, A. Aramburu, H. Bengtsson, P. Neuvial and A. Rubio, *CalMaTe: A method and software to improve allele-specific copy number of SNP arrays for downstream segmentation*, Bioinformatics, 2012 [PMC3381965].

## See Also

To calibrate (total,fracB) data, see `*calmateByTotalAndFracB()`. We strongly recommend to always work with (total,fracB) data instead of (CA,CB) data, because it is much more general.

For further information on the internal fit functions, see `fitCalMaTeInternal`.

## Examples

```
# Load example (thetaA,thetaB) signals
path <- system.file("exData", package="calmate");
theta <- loadObject("thetaAB,100x2x40.Rbin", path=path);

# Calculate (CA,CB)
thetaR <- matrixStats::rowMedians(theta[,"A",] + theta[,"B",], na.rm=TRUE);
C <- 2*theta/thetaR;

# Calibrate (CA,CB) by CalMaTe
CC <- calmateByThetaAB(theta);

# Plot two "random" arrays
Clim <- c(0,4);
subplots(4, ncol=2, byrow=FALSE);
for (ii in c(1,5)) {
  sampleName <- dimnames(C)[[3]][ii];
  sampleLabel <- sprintf("Sample #%d ('%s')", ii, sampleName);
  plot(C[,,ii], xlim=Clim, ylim=Clim);
  title(main=sampleLabel);
  plot(CC[,,ii], xlim=Clim, ylim=Clim);
  title(main=sprintf("%s\ncalibrated", sampleLabel));
}
```

---

`calmateByTotalAndFracB.array` *Normalize allele-specific copy numbers (total,fracB)*

---

## Description

Normalize allele-specific copy numbers (total,fracB), where total is the total (non-polymorphic) signal and fracB is the allele B fraction. It is only loci with a non-missing (NA) fracB value that are considered to be SNPs and normalized by CalMaTe. The other loci are left untouched.

## Usage

```
## S3 method for class 'array'
calmateByTotalAndFracB(data, references=NULL, ..., refAvgFcn=NULL, verbose=FALSE)
```

## Arguments

| | |
|---|---|
| data | An Jx2xI numeric array, where J is the number of loci, 2 is total and fracB (in that order, if unnamed), and I is the number of samples. |
| references | A logical or numeric vector specifying which samples should be used as the reference set. By default, all samples are considered. If not NULL at least 3 samples. |
| ... | Additional arguments passed to *calmateByThetaAB(). |
| refAvgFcn | (optional) A function that takes a JxI numeric matrix an argument na.rm and returns a numeric vector of length J. It should calculate some type of average for each of the J rows, e.g. rowMedians. If specified, then the total copy numbers of the calibrated ASCNs are standardized toward (twice) the average of the total copy numbers of the calibrated reference ASCNs. |
| verbose | See Verbose. |

## Value

Returns an Jx2xI numeric array with the same dimension names as argument data.

## References

[1] M. Ortiz-Estevez, A. Aramburu, H. Bengtsson, P. Neuvial and A. Rubio, *CalMaTe: A method and software to improve allele-specific copy number of SNP arrays for downstream segmentation*, Bioinformatics, 2012 [PMC3381965].

## See Also

To calibrate (thetaA,thetaB) or (CA,CB) signals, see *calmateByThetaAB().

## Examples

```
# Load example (thetaA,thetaB) signals
path <- system.file("exData", package="calmate");
theta <- loadObject("thetaAB,100x2x40.Rbin", path=path);

# Transform to (total,fracB) signals
data <- thetaAB2TotalAndFracB(theta);
```

```
# Calibrate (total,fracB) by CalMaTe
dataC <- calmateByTotalAndFracB(data);

# Calculate copy-number ratios
theta <- data[,"total",];
thetaR <- matrixStats::rowMedians(theta, na.rm=TRUE);
data[,"total",] <- 2*theta/thetaR;

# Plot two "random" arrays
Clim <- c(0,4);
Blim <- c(0,1);
subplots(4, ncol=2, byrow=FALSE);
for (ii in c(1,5)) {
  sampleName <- dimnames(data)[[3]][ii];
  sampleLabel <- sprintf("Sample #%d ('%s')", ii, sampleName);
  plot(data[,,ii], xlim=Clim, ylim=Blim);
  title(main=sampleLabel);
  plot(dataC[,,ii], xlim=Clim, ylim=Blim);
  title(main=sprintf("%s\ncalibrated", sampleLabel));
}


# Assert that it also works with a single unit
dummy <- calmateByTotalAndFracB(data[1,,,drop=FALSE]);
stopifnot(length(dim(dummy)) == 3);
```

---

`CalMaTeCalibration` *The CalMaTeCalibration class*

---

## Description

Package: calmate
**Class CalMaTeCalibration**

```
Object
~~|
~~+--ParametersInterface
~~~~~~~|
~~~~~~~+--CalMaTeCalibration
```

**Directly known subclasses:**

public static class **CalMaTeCalibration**
extends ParametersInterface

This class represents the CalMaTe method [1], which corrects for SNP effects in allele-specific copy-number estimates (ASCNs).

## Usage

```
CalMaTeCalibration(data=NULL, tags="*", references=NULL, flavor=c("v2", "v1"), ...)
```

## Arguments

| | |
|---|---|
| data | A named list with data set named "total" and "fracB" where the former should be of class AromaUnitTotalCnBinarySet and the latter of class AromaUnitFracBCnBinarySet. The two data sets must be for the same chip type, have the same number of samples and the same sample names. |
| tags | Tags added to the output data sets. |
| references | An optional numeric vector specifying which samples should be as reference samples for estimating the model parameters. If NULL, all samples are used. |
| flavor | A character string specifying which flavor of the CalMaTe algorithm to use for fitting the model. See fitCalMaTeInternal for details. |
| ... | Additional arguments passed to calmateByTotalAndFracB(). |

## Fields and Methods

### Methods:

| | |
|---|---|
| findUnitsTodo | - |
| getDataSets | - |
| getFullName | - |
| getName | - |
| getOutputDataSets | - |
| getPath | - |
| getReferences | - |
| getRootPath | - |
| getTags | - |
| nbrOfFiles | - |
| process | - |
| setTags | - |

**Methods inherited from ParametersInterface**:
getParameterSets, getParameters, getParametersAsString

**Methods inherited from Object**:
$, $<-, [[, [[<-, as.character, attach, attachLocally, clearCache, clearLookupCache, clone, detach, equals, extend, finalize, getEnvironment, getFieldModifier, getFieldModifiers, getFields, getInstantiationTime, getStaticInstance, hasField, hashCode, ll, load, names, objectSize, print, save, asThis

## Reference samples

In order to estimate the calibration parameters, the model assumes that, for any given SNP, there are a majority of samples that are diploid at that SNP. Note that it does not have to be the same set of samples for all SNPs.

By using argument `references`, it is possible so specify which samples should be used when estimating the calibration parameters. This is useful when for instance there are several tumor samples with unknown properties as well as a set of normal samples that can be assumed to be diploid.

Theoretical, a minimum of three reference samples are needed in order for the model to be identifiable. If less, an error is thrown. However, in practice more reference samples should be used, that is, in the order of at least 6-10 reference samples with a diverse set of genotypes.

## Flavors

For backward compatibility, we try to keep all major versions of the CalMaTe algorithm available. Older versions can be used by specifying argument `flavor`. For more information about the different flavors, see `fitCalMaTeInternal`.

## References

[1] M. Ortiz-Estevez, A. Aramburu, H. Bengtsson, P. Neuvial and A. Rubio, *CalMaTe: A method and software to improve allele-specific copy number of SNP arrays for downstream segmentation*, Bioinformatics, 2012 [PMC3381965].

## See Also

Low-level versions of the CalMaTe method is available via the `calmateByThetaAB`() and `calmateByTotalAndFracB`() methods.

For further information on the internal fit functions, see `fitCalMaTeInternal`.

## Examples

```
## Not run:

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# CRMAv2 - Preprocess raw Affymetrix data
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
library("aroma.affymetrix");  # Needed for CRMAv2
dataSet <- "Affymetrix_2006-TumorNormal";
chipType <- "Mapping250K_Nsp";
dsList <- doCRMAv2(dataSet, chipType=chipType, combineAlleles=FALSE,
                                            plm="RmaCnPlm", verbose=-10);
print(dsList);


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# CalMaTe - Post-calibration of ASCNs estimates
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
asn <- CalMaTeCalibration(dsList);
print(asn);

# For speed issues, we will here only process loci on Chromosome 17.
chr <- 17;
ugp <- getAromaUgpFile(dsList$total);
```

```
units <- getUnitsOnChromosome(ugp, chr);

dsNList <- process(asn, units=units, verbose=verbose);
print(dsNList);


# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Plot allele B fractions (before and after)
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Sample #1 and Chromosome 17
ii <- 1;

# Extract raw (TCN,BAF)
df <- getFile(dsList$total, ii);
dfR <- getAverageFile(dsList$total, verbose=verbose);
gamma <- extractRawCopyNumbers(df, logBase=NULL, chromosome=chr);
gammaR <- extractRawCopyNumbers(dfR, logBase=NULL, chromosome=chr);
gamma <- 2*divideBy(gamma, gammaR);
df <- getFile(dsList$fracB, ii);
beta <- extractRawAlleleBFractions(df, chromosome=chr);

# Extract calibrated (TCN,BAF)
dfN <- getFile(dsNList$fracB, ii);
betaN <- extractRawAlleleBFractions(dfN, chromosome=chr);
dfN <- getFile(dsNList$total, ii);
gammaN <- extractRawCopyNumbers(dfN, logBase=NULL, chromosome=chr);

# Plot
subplots(4, ncol=2, byrow=FALSE);
plot(beta);
title(sprintf("%s", getName(beta)));
plot(gamma);
plot(betaN);
title(sprintf("%s (CalMaTe)", getName(betaN)));
plot(gammaN);


## End(Not run)
```

# Index