

# Package ‘RChASM’

May 7, 2026

**Title** Detection of Chromosomal Aneuploidies in Ancient DNA Studies

**Version** 1.0.1

**Description** An R implementation of ChASM (Chromosomal Aneuploidy Screening Methodology): a statistically rigorous Bayesian approach for screening data sets for autosomal and sex chromosomal aneuploidies. This package takes as input the number of (deduplicated) reads mapping to chromosomes 1-22 and the X and Y chromosomes, and models these using a Dirichlet-multinomial distribution. From this, This package returns posterior probabilities of sex chromosomal karyotypes (XX, XY, XXY, XYY, XXX and X) and full autosomal aneuploidies (trisomy 13, trisomy 18 and trisomy 21). This package also returns two diagnostic statistics: (i) a posterior probability addressing whether contamination between XX and XY may explain the observed sex chromosomal aneuploidy, and (ii) a chi-squared statistic measuring whether the observed read counts are too divergent from the underlying distribution (and may represent abnormal sequencing/quality issues).

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**Imports** checkmate, cowplot, dplyr, EnvStats, ggplot2, ggrepel, ggsci, ggstar, magrittr, matrixStats, mclust, rstatix, sirt, stringr, tibble, tidyr, readr

**LazyData** true

**Suggests** knitr, rmarkdown, extraDistr

**VignetteBuilder** knitr

**Depends** R (>= 3.5.0)

**URL** <https://jonotuke.github.io/RChASM/>,  
<https://github.com/jonotuke/RChASM>

**BugReports** <https://github.com/jonotuke/RChASM/issues>

**NeedsCompilation** no

**Author** Adam B. Rohrlach [aut, cph] (ORCID:

<<https://orcid.org/0000-0002-4204-5018>>),

Jono Tuke [aut, cre] (ORCID: <<https://orcid.org/0000-0002-1688-8951>>),

Wolfgang Haak [aut] (ORCID: <<https://orcid.org/0000-0003-2475-2007>>)

**Maintainer** Jono Tuke <simon.tuke@adelaide.edu.au>

**Repository** CRAN

**Date/Publication** 2026-04-22 08:10:10 UTC

## Contents

adjustA	2
bound	3
callKaryotypes	3
combineData	4
ddirichletultinomial	4
dirichlet.mle_filter	5
estimateGamma	5
example_data	6
filterOutliers	6
makeC	7
makeDirichlet	7
makeZscores	8
plot_diagnostic	8
printChASM	9
processReadCounts	9
runChASM	10
saveChASM	11
summary_calls	12

<b>Index</b>	<b>13</b>
--------------	-----------

---

adjustA	<i>Adjusts alpha vector for contamination karyotypes</i>
---------	--

---

### Description

Adjusts alpha vector for contamination karyotypes

### Usage

```
adjustA(a, c1, c2, g, correction = 1e-06)
```

### Arguments

a	vector to change
c1	X multiplier for karyotype 1
c2	X multiplier for karyotype 2
g	mixing parameter
correction	error rate for Y mapping for non-Y karyotypes

**Value**

adjusted alpha vector

---

bound	<i>bounds number to be in [0,1] (contamination estimation)</i>
-------	--

---

**Description**

bounds number to be in [0,1] (contamination estimation)

**Usage**

bound(x)

**Arguments**

x                    number

**Value**

bounded number

---

callKaryotypes	<i>Take prior and observed counts and calls karyotypes and Z-scores</i>
----------------	---

---

**Description**

Take prior and observed counts and calls karyotypes and Z-scores

**Usage**

callKaryotypes(indat, inDirichlet, p\_contamination = 0.1)

**Arguments**

indat	full input tibble
inDirichlet	associated Dirichlet parameters
p_contamination	the assumed probability of contamination

**Value**

karyotype calls

---

combineData            *combine data*

---

### Description

Combine both autosomal and sex chromosomal analyses into one output

### Usage

```
combineData(calls.auto, calls.sca, z.scores, printMissingIDs = FALSE)
```

### Arguments

calls.auto	a list or predefined piped string of sample IDs to look for
calls.sca	the autosomal karyotype calls for the samples
z.scores	the sex chromosomal karyotype calls for the samples
printMissingIDs	print the IDs of individuals not in all 3 input files (or just numbers)?

### Value

combined data

---

ddirichletultinomial    *calculate Dirichlet probabilities with filtering*

---

### Description

calculate Dirichlet probabilities with filtering

### Usage

```
ddirichletultinomial(nvector, avector, cvector, log = TRUE, correction = 1e-06)
```

### Arguments

nvector	observed read counts
avector	alpha vector to change
cvector	change vector for new karyotype
log	return value in log space
correction	error rate for Y mapping for non-Y karyotypes

### Value

Dirichlet probabilities

---

dirichlet.mle\_filter    *calculate Dirichlet parameters with filtering*

---

**Description**

calculate Dirichlet parameters with filtering

**Usage**

```
dirichlet.mle_filter(y, whichK)
```

**Arguments**

y	input values
whichK	which parameter to return

**Value**

parameters

---

estimateGamma    *Estimate contamination from XX and XY based on observed counts*

---

**Description**

Estimate contamination from XX and XY based on observed counts

**Usage**

```
estimateGamma(nvector, avector, c1, c2, correction)
```

**Arguments**

nvector	observed read counts
avector	vector to change
c1	X multiplier for karyotype 1
c2	X multiplier for karyotype 2
correction	error rate for Y mapping for non-Y karyotypes

**Value**

estimated contamination

---

example\_data

*example\_data*

---

**Description**

An example data set for RChASM

**Usage**

example\_data

**Format**

A data frame with 266 rows and 26 variables:

**Source**

"Ben Rohrlach"

---

filterOutliers

*filter for outliers*

---

**Description**

filter for outliers

**Usage**

filterOutliers(v)

**Arguments**

v                      vector to filter

**Value**

vector for filtering

---

makeC	<i>make c vector for new karyotypes</i>
-------	---

---

**Description**

make c vector for new karyotypes

**Usage**

```
makeC(x, y, n)
```

**Arguments**

x	value to replace 1
y	location to make replacement
n	length of vector

**Value**

c vector

---

makeDirichlet	<i>function to make a Dirichlet prior</i>
---------------	---

---

**Description**

function to make a Dirichlet prior

**Usage**

```
makeDirichlet(  
  indat,  
  refType,  
  min_reads = 30000,  
  max_reads = 1e+09,  
  show_plot = TRUE  
)
```

**Arguments**

indat	full input tibble
refType	"auto"="autosomal"; "sca"="sex chromosomal"; "diagnostic"
min_reads	min number of reads per sample
max_reads	max number of reads per sample
show_plot	boolean to show plot

**Value**

Dirichlet prior

---

makeZscores	<i>Calculates Z-scores per chromosome</i>
-------------	---

---

**Description**

Calculates Z-scores per chromosome

**Usage**

```
makeZscores(indat, refType = "auto", min_reads = 30000, max_reads = 1e+09)
```

**Arguments**

indat	full input tibble
refType	"auto"="autosomal"; "sca"="sex chromosomal"; "diagnostic"
min_reads	min number of reads per sample
max_reads	max number of reads per sample

**Value**

Z-scores

---

plot_diagnostic	<i>Plots diagnostic plots</i>
-----------------	-------------------------------

---

**Description**

Plots diagnostic plots

**Usage**

```
plot_diagnostic(IDs, inChASM, addLabels = FALSE)
```

**Arguments**

IDs	a list or predefined piped string of sample IDs to look for
inChASM	parameter object
addLabels	add "A", "B" and "C" to the plot panels?

**Value**

plots

### Examples

```
example_calls <- runChASM(rawReadCountsIn = example_data)
plot_diagnostic(IDs = 'Ind_255_1', inChASM = example_calls, addLabels = TRUE)
```

---

printChASM	<i>A function to print the result of the combined analysis to screen</i>
------------	--

---

### Description

A function to print the result of the combined analysis to screen

### Usage

```
printChASM(inChASM, lines = 20)
```

### Arguments

inChASM	the result of a full ChASM analysis (output from runChASM)
lines	the number of lines to print to screen

### Value

print output

### Examples

```
example_calls <- runChASM(rawReadCountsIn = example_data)
printChASM(inChASM = example_calls, lines = 10)
```

---

processReadCounts	<i>processReadCounts</i>
-------------------	--------------------------

---

### Description

Task: takes read counts and makes useful file

### Usage

```
processReadCounts(
  rawReadCountsIn,
  refType,
  minTotal = 1000,
  minSamplesPerProtocol = 30
)
```

**Arguments**

rawReadCountsIn            the reads counts via KP  
 refType                    for capture, use only on-target (F)?  
 minTotal                  minimum number of protocol counts to be included  
 minSamplesPerProtocol    "auto"="autosomal"; "sca"="sex chromosomal"

**Value**

outreadcounts

---

runChASM

*Title*

---

**Description**

Title

**Usage**

```
runChASM(
  rawReadCountsIn,
  minSamplesPerProtocol = 30,
  min_reads = 60000,
  max_reads = 1e+09,
  p_contamination = 0.01,
  show_plot = TRUE,
  printMissingIDs = FALSE
)
```

**Arguments**

rawReadCountsIn            the reads counts for each chromosome  
 minSamplesPerProtocol    minimum number of reads per protocol for parameter estimation  
 min\_reads                 the minimum number of reads for Dirichlet parameter estimation  
 max\_reads                 the maximum number of reads for Dirichlet parameter estimation  
 p\_contamination           the probability of a sample yielding significant contamination  
 show\_plot                 show the clustering plot for sex chromosomal aneuploidy Dirichlet estimation?  
 printMissingIDs           when combining karyotype calls, return names that are missing (or just the number of missing IDs)?

**Value**

summary

**Examples**

```
runChASM(rawReadCountsIn = example_data)
```

---

saveChASM	<i>A function for saving the results of a ChASM analysis as a tsv</i>
-----------	---

---

**Description**

A function for saving the results of a ChASM analysis as a tsv

**Usage**

```
saveChASM(inChASM, file, sort_by_samplename = FALSE)
```

**Arguments**

inChASM	the result of a full ChASM analysis (output from runChASM)
file	the full path and file name to place output
sort_by_samplename	reorder alphabetically by sample names?

**Value**

saves results

**Examples**

```
example_calls <- runChASM(rawReadCountsIn = example_data)
## Not run:
saveChASM(inChASM = example_calls)

## End(Not run)
```

---

summary_calls	<i>summary_calls</i>
---------------	----------------------

---

**Description**

Combine both autosomal and sex chromosomal analyses into one output

**Usage**

```
summary_calls(  
  inChASM,  
  minTotal = 60000,  
  minPosterior = 0.95,  
  ignoreUnusual = FALSE,  
  printProtocol = FALSE  
)
```

**Arguments**

<code>inChASM</code>	the result of combining both autosomal and sca analyses
<code>minTotal</code>	minimum total for autosomal or sca analyses
<code>minPosterior</code>	minimum value maxP can take (i.e. minimum posterior probability accepted)
<code>ignoreUnusual</code>	filter our unusual observations?
<code>printProtocol</code>	print protocol column?

**Value**

combined calls

**Examples**

```
example_calls <- runChASM(rawReadCountsIn = example_data)  
summary_calls(inChASM = example_calls, minTotal = 6e4, minPosterior = 0.95)
```

# Index

- \* **datasets**
  - example\_data, 6
- adjustA, 2
- bound, 3
- callKaryotypes, 3
- combineData, 4
- ddirichletultinomial, 4
- dirichlet.mle\_filter, 5
- estimateGamma, 5
- example\_data, 6
- filterOutliers, 6
- makeC, 7
- makeDirichlet, 7
- makeZscores, 8
- plot\_diagnostic, 8
- printChASM, 9
- processReadCounts, 9
- runChASM, 10
- saveChASM, 11
- summary\_calls, 12