

# Package: dnaEPICO (via r-universe)

May 30, 2026

**Title** dnaEPICO: Analysis Pipeline for Illumina DNA Methylation Array Data

**Version** 0.99.22

**Date** 2026-01-18

**Description** A modular and reproducible workflow for preprocessing and analysing Illumina DNA methylation array data from the EPICv2, EPIC, and 450K platforms. The package integrates quality control, probe filtering, cell-type deconvolution, phenotype preparation, generalised linear models, linear mixed-effects models, and automated report generation. It builds on established Bioconductor infrastructure and wraps commonly used tools including 'minfi', 'ENmix', and 'wateRmelon', with support for both local execution and high-performance computing workflows.

**License** AGPL-3 + file LICENSE

**URL** <https://github.com/paulYRP/dnaEPICO>

**BugReports** <https://github.com/paulYRP/dnaEPICO/issues>

**Depends** R (>= 4.4)

**Imports** Biobase, RColorBrewer, data.table, ggplot2, MASS, methods, minfi, openxlsx, limma, wateRmelon, ENmix, ggrepel, glm2, parallel, lme4, lmerTest, quadprog, SummarizedExperiment, utils, stats

**Suggests** IlluminaHumanMethylation450kmanifest, IlluminaHumanMethylation450kanno.ilmn12.hg19, IlluminaHumanMethylationEPICv2anno.20a1.hg38, IlluminaHumanMethylationEPICv2manifest, IlluminaHumanMethylationEPICmanifest, IlluminaHumanMethylationEPICanno.ilm10b4.hg19, minfiData, GenomicRanges, S4Vectors, remotes, BiocManager, BiocStyle, RefManagerR, sessioninfo, devtools, testthat, tiff, magick, knitr, rmarkdown, covr, withr, tinytex, FlowSorted.Blood.EPIC, FlowSorted.Blood.450k, BeadSorted.Saliva.EPIC

**VignetteBuilder** knitr

**biocViews** Software, Preprocessing, MethylationArray, QualityControl, Epigenetics, Microarray, StatisticalMethod, ChipOnChip

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Config/testthat/edition** 3

**Config/pak/sysreqs** cmake make libbz2-dev libicu-dev liblzma-dev libpng-dev libxml2-dev libssl-dev libx11-dev xz-utils zlib1g-dev

**Repository** <https://bioc.r-universe.dev>

**Date/Publication** 2026-05-28 19:50:14 UTC

**RemoteUrl** <https://github.com/bioc/dnaEPICO>

**RemoteRef** HEAD

**RemoteSha** 65bbf248d4e52d70151d3e6f04b0c3d91f512de4

## Contents

analyzeSvaEnmix . . . . .	3
annotateMethylationGLM_T1Summaries . . . . .	5
annotateMethylationGLMM_T1T2Summaries . . . . .	6
assessSamplesMinfiEwasWater . . . . .	7
buildClockFoundationInputsPreprocessingPheno . . . . .	8
buildRawMinfiEwasWater . . . . .	9
collectSignificantCpGsMethylationGLM_T1 . . . . .	10
collectSignificantInteractionsMethylationGLMM_T1T2 . . . . .	11
combineTimepointsPreprocessingPheno . . . . .	13
dnaEPICO . . . . .	14
dnaEPICO_dnamReport-class . . . . .	15
dnaEPICO_dnamReport_prepared-class . . . . .	15
dnaEPICO_dnamReport_render-class . . . . .	16
dnaEPICO_methylationGLM_T1-class . . . . .	16
dnaEPICO_methylationGLMM_T1T2-class . . . . .	17
dnaEPICO_preprocessingMinfiEwasWater-class . . . . .	17
dnaEPICO_preprocessingPheno-class . . . . .	18
dnaEPICO_svaEnmix-class . . . . .	19
dnamReport . . . . .	19
estimateLC . . . . .	22
estimateLCMinfiEwasWater . . . . .	23
estimateSvaEnmixControls . . . . .	24
extractMake . . . . .	25
extractMetricsMinfiEwasWater . . . . .	26
filterProbesMinfiEwasWater . . . . .	27
filterSamplesMinfiEwasWater . . . . .	28
fitMethylationGLM_T1Models . . . . .	30
fitMethylationGLMM_T1T2Models . . . . .	31

loadMetricsPreprocessingPheno . . . . .	32
mergeSvaTargetsEnmix . . . . .	33
methylationGLM_T1 . . . . .	34
methylationGLMM_T1T2 . . . . .	38
normalizeMinfiEwasWater . . . . .	42
plotAssessmentMinfiEwasWater . . . . .	43
plotCtrlMinfiEwasWater . . . . .	45
plotMethylationGLM_T1Diagnostics . . . . .	46
plotMethylationGLM_T1Distributions . . . . .	47
plotMethylationGLMM_T1T2Diagnostics . . . . .	48
plotMetricsMinfiEwasWater . . . . .	50
plotNormalizationMinfiEwasWater . . . . .	51
plotRawDensityMinfiEwasWater . . . . .	52
plotSexMinfiEwasWater . . . . .	54
plotSvaEnmix . . . . .	55
predictSexMinfiEwasWater . . . . .	56
prepareDnamReportInputs . . . . .	57
prepareMethylationGLM_T1Data . . . . .	58
prepareMethylationGLMM_T1T2Data . . . . .	60
preprocessingMinfiEwasWater . . . . .	62
preprocessingPheno . . . . .	65
print.dnaEPICO_dnamReport . . . . .	68
readPhenotypeTargets . . . . .	69
readRGSetMinfiEwasWater . . . . .	70
renderDnamReport . . . . .	71
splitTimepointsPreprocessingPheno . . . . .	72
summarizeMethylationGLM_T1Models . . . . .	74
summarizeMethylationGLMM_T1T2Models . . . . .	75
summarizeTimepointsMethylationGLMM_T1T2 . . . . .	76
svaEnmix . . . . .	77
writeMethylationGLM_T1Outputs . . . . .	80
writeMethylationGLMM_T1T2Outputs . . . . .	82
writePhenoLCMinfiEwasWater . . . . .	84
writePreprocessingPhenoOutputs . . . . .	85
writeSvaEnmixOutputs . . . . .	86
<b>Index</b>	<b>89</b>

---

analyzeSvaEnmix	<i>Analyze surrogate variables against Sentrix chip and position factors</i>
-----------------	--

---

## Description

Fit linear models for each surrogate variable against Sentrix chip and Sentrix position, perform backward elimination with `MASS::dropterm()`, and return the in-memory analysis objects.

**Usage**

```
analyzeSvaEnmix(
  sva,
  RGSet,
  SentrixIDColumn = "Sentrix_ID",
  SentrixPositionColumn = "Sentrix_Position",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_analyzeSvaEnmix.txt"
)
```

**Arguments**

sva	Numeric matrix of surrogate variables with samples in rows.
RGSet	An RGChannelSet aligned with sva.
SentrixIDColumn	Character. Name of the chip identifier column in SummarizedExperiment::colData(RGSet).
SentrixPositionColumn	Character. Name of the chip position column in SummarizedExperiment::colData(RGSet).
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_svaEnmix\_analysis" containing the aligned Sentrix factors, full and reduced linear models, and ANOVA tables.

**Examples**

```
ex <- dnaEPICO::exampleSvaAnalysisStateDnaEpico()
analysis_data <- analyzeSvaEnmix(
  sva = ex$sva,
  RGSet = ex$RGSet,
  SentrixIDColumn = "Sentrix_ID",
  SentrixPositionColumn = "Sentrix_Position",
  verbose = FALSE,
  logs = FALSE
)
analysis_data$K
```

---

annotateMethylationGLM\_T1Summaries

*Annotate one-timepoint GLM summary tables with array annotation metadata*

---

## Description

Merge phenotype-specific CpG summary tables with probe annotation metadata and return a single annotated result table.

## Usage

```
annotateMethylationGLM_T1Summaries(
  modelSummaries,
  annotationObject,
  annotationCols = c("Name", "chr", "pos", "UCSC_RefGene_Group", "UCSC_RefGene_Name",
    "Relation_to_Island", "GencodeV41_Group"),
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLM_T1.txt"
)
```

## Arguments

**modelSummaries** Object returned by `summarizeMethylationGLM_T1Models()` or a named list of CpG summary data frames.

**annotationObject** Character package/object name, annotation data frame, or annotation object understood by `minfi::getAnnotation()`.

**annotationCols** Character vector or comma-separated string of annotation columns to append.

**verbose** Logical. If TRUE, emit progress messages with `message()`.

**logs** Logical. If TRUE, write the same messages to a log file.

**log\_dir** Character or NULL. Directory used for the log file when `logs = TRUE`.

**log\_file** Character. File name used when `logs = TRUE`.

## Value

A list with class `"dnaEPIC0_methylationGLM_T1_annotation"` containing the annotated summary table and any requested annotation columns that were unavailable in the chosen annotation object.

**Examples**

```

ex <- dnaEPICO::exampleMethylationGLMStateDnaEpico()
annotation_data <- annotateMethylationGLM_T1Summaries(
  modelSummaries = ex$modelSummaries,
  annotationObject = ex$annotationData,
  annotationCols = "Name,chr,pos",
  verbose = FALSE,
  logs = FALSE
)
names(annotation_data)

```

---

```

annotateMethylationGLMM_T1T2Summaries

```

*Annotate longitudinal mixed-effects summary tables with array annotation metadata*

---

**Description**

Merge phenotype-specific longitudinal summary tables with probe annotation metadata and return a single annotated result table.

**Usage**

```

annotateMethylationGLMM_T1T2Summaries(
  modelSummaries,
  annotationObject,
  annotationCols = c("Name", "chr", "pos", "UCSC_RefGene_Group", "UCSC_RefGene_Name",
    "Relation_to_Island", "GencodeV41_Group"),
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLMM_T1T2.txt"
)

```

**Arguments**

<code>modelSummaries</code>	Object returned by <code>summarizeMethylationGLMM_T1T2Models()</code> or a named list of summary data frames.
<code>annotationObject</code>	Character package/object name, annotation data frame, or annotation object understood by <code>minfi::getAnnotation()</code> .
<code>annotationCols</code>	Character vector or comma-separated string of annotation columns to append.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

**Value**

A list with class "dnaEPICO\_methylationGLMM\_T1T2\_annotation" containing the annotated summary table and any requested annotation columns that were unavailable in the chosen annotation object.

**Examples**

```
ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
annotation_data <- annotateMethylationGLMM_T1T2Summaries(
  modelSummaries = ex$modelSummaries,
  annotationObject = ex$annotationData,
  annotationCols = "Name,chr,pos",
  verbose = FALSE,
  logs = FALSE
)
names(annotation_data)
```

---

assessSamplesMinfiEwasWater

*Assess sample quality before sample filtering*

---

**Description**

Compute minfi QC metrics and detection P values, identify failed samples using the requested threshold, and return the assessment as a single object.

**Usage**

```
assessSamplesMinfiEwasWater(
  rawData,
  RGSet,
  qcCutoff = 10.5,
  detPtype = "m+u",
  detPThreshold = 0.05,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_assessSamplesMinfiEwasWater.txt"
)
```

**Arguments**

rawData	Object returned by buildRawMinfiEwasWater().
RGSet	An RGChannelSet aligned with rawData.
qcCutoff	Numeric. Cutoff passed to minfi::plotQC() when the QC plot is drawn.

<code>detPtype</code>	Character. Detection P-value mode passed to <code>minfi::detectionP()</code> . Common values in minfi workflows are "m+u" and "negative". The default used here is "m+u".
<code>detPThreshold</code>	Numeric. Samples with mean detection P value above this threshold are marked as failed.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

**Value**

A list with class "dnaEPICO\_minfiEwasWater\_assessment" containing the QC object, detection P matrix, mean detection P values, and failed sample identifiers.

**Examples**

```
ex <- dnaEPICO::exampleMinfiBaseDataDnaEpico()
raw_data <- buildRawMinfiEwasWater(ex$RGSet, verbose = FALSE, logs = FALSE)
assessment <- assessSamplesMinfiEwasWater(
  rawData = raw_data,
  RGSet = ex$RGSet,
  detPThreshold = 1,
  verbose = FALSE,
  logs = FALSE
)
names(assessment)
```

---

`buildClockFoundationInputsPreprocessingPheno`

*Build Clock Foundation input tables from preprocessingPheno data*

---

**Description**

Prepare the beta and phenotype tables commonly exported for Clock Foundation style downstream workflows, without writing them to disk.

**Usage**

```
buildClockFoundationInputsPreprocessingPheno(
  beta,
  pheno,
  SampleID = "Sample_Name",
  sexColumn = "Sex",
  verbose = FALSE,
  logs = FALSE,
```

```

    log_dir = NULL,
    log_file = "log_buildClockFoundationInputsPreprocessingPheno.txt"
  )

```

### Arguments

beta	Numeric matrix of beta values with probes in rows and samples in columns.
pheno	Phenotype data frame aligned with the beta matrix columns.
SampleID	Character. Name of the phenotype sample identifier column.
sexColumn	Character. Name of the phenotype sex column.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

### Value

A list with class "dnaEPICO\_preprocessingPheno\_clock" containing betaCSV and phenoCF.

### Examples

```

ex <- dnaEPICO::examplePreprocessingPhenoStateDnaEpico()
clock_inputs <- buildClockFoundationInputsPreprocessingPheno(
  beta = ex$timepointData$data[["1"]]$beta,
  pheno = ex$timepointData$data[["1"]]$pheno,
  SampleID = "Sample_Name",
  sexColumn = "Sex",
  verbose = FALSE,
  logs = FALSE
)
names(clock_inputs)

```

---

buildRawMinfiEwasWater

*Build raw minfi preprocessing objects*

---

### Description

Create a raw MethylSet, RatioSet, and genome-mapped object from an RGChannelSet, and return them together in a single structured object.

**Usage**

```
buildRawMinfiEwasWater(  
  RGSet,  
  verbose = FALSE,  
  logs = FALSE,  
  log_dir = NULL,  
  log_file = "log_buildRawMinfiEwasWater.txt"  
)
```

**Arguments**

RGSet	An RGChannelSet.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_minfiEwasWater\_raw" containing MSet, RatioSet, and GSet.

**Examples**

```
ex <- dnaEPICO::exampleMinfiBaseDataDnaEpico()  
raw_data <- buildRawMinfiEwasWater(  
  RGSet = ex$RGSet,  
  verbose = FALSE,  
  logs = FALSE  
)  
names(raw_data)
```

---

collectSignificantCpGsMethylationGLM\_T1

*Collect significant CpG coefficient tables from fitted one-timepoint GLMs*

---

**Description**

Collect the raw coefficient tables for CpGs whose phenotype main effect or interaction p-value passes the requested threshold.

**Usage**

```
collectSignificantCpGsMethylationGLM_T1(
  modelResults,
  pvalThreshold = 0.05,
  interactionTerm = NULL,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLM_T1.txt"
)
```

**Arguments**

modelResults	Object returned by fitMethylationGLM_T1Models().
pvalThreshold	Numeric. Threshold applied to phenotype main-effect or interaction p-values.
interactionTerm	Character or NULL. Optional interaction term.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_methylationGLM\_T1\_significant\_cpgs".

**Examples**

```
ex <- dnaEPICO::exampleMethylationGLMStateDnaEpico()
significant_cpgs <- collectSignificantCpGsMethylationGLM_T1(
  modelResults = ex$modelResults,
  pvalThreshold = 1,
  verbose = FALSE,
  logs = FALSE
)
names(significant_cpgs)
```

---

collectSignificantInteractionsMethylationGLMM\_T1T2

*Collect significant longitudinal model terms from fitted mixed-effects models*

---

**Description**

Collect raw coefficient tables for CpGs whose phenotype main effect or requested interaction p-value passes the chosen threshold.

**Usage**

```
collectSignificantInteractionsMethylationGLMM_T1T2(
  modelResults,
  pvalThreshold = 0.05,
  interactionTerm = NULL,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLMM_T1T2.txt"
)
```

**Arguments**

<code>modelResults</code>	Object returned by <code>fitMethylationGLMM_T1T2Models()</code> .
<code>pvalThreshold</code>	Numeric. Threshold applied to the extracted phenotype or interaction p-values.
<code>interactionTerm</code>	Character or NULL. Optional interaction term. When NULL, phenotype main effects are used.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

**Value**

A list with class "dnaEPICO\_methylationGLMM\_T1T2\_significant" containing the retained coefficient tables for each phenotype.

**Examples**

```
ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
significant_hits <- collectSignificantInteractionsMethylationGLMM_T1T2(
  modelResults = ex$modelResults,
  pvalThreshold = 1,
  verbose = FALSE,
  logs = FALSE
)
names(significant_hits)
```

---

combineTimepointsPreprocessingPheno  
*Combine selected timepoints for downstream longitudinal modeling*

---

## Description

Combine selected timepoints that were already aligned by `splitTimepointsPreprocessingPheno()` into the wide phenotype-plus-beta objects used by downstream longitudinal models.

## Usage

```
combineTimepointsPreprocessingPheno(  
  timepointData,  
  combineTimepoints = "1,2",  
  verbose = FALSE,  
  logs = FALSE,  
  log_dir = NULL,  
  log_file = "log_combineTimepointsPreprocessingPheno.txt"  
)
```

## Arguments

<code>timepointData</code>	Object returned by <code>splitTimepointsPreprocessingPheno()</code> .
<code>combineTimepoints</code>	Character vector or comma-separated string of timepoints to combine.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

## Value

A list with class `"dnaEPICO_preprocessingPheno_combined"` containing the combined phenotype table, merged phenotype-plus-beta table, selected timepoints, and output suffix.

## Examples

```
ex <- dnaEPICO::examplePreprocessingPhenoStateDnaEpico()  
combined_data <- combineTimepointsPreprocessingPheno(  
  timepointData = ex$timepointData,  
  combineTimepoints = "1,2",  
  verbose = FALSE,  
  logs = FALSE  
)  
combined_data$suffix
```

---

dnaEPICO

*dnaEPICO: DNA methylation preprocessing and modeling workflows*

---

## Description

The dnaEPICO package provides a structured workflow for preprocessing and analyzing Illumina DNA methylation array data, including quality control, normalization, cell-type estimation, surrogate-variable analysis, phenotype preparation, cross-sectional modeling, longitudinal mixed-effects modeling, and PDF reporting.

## Details

The package supports two complementary usage styles:

- interactive use, where functions return structured in-memory result objects for inspection and composition; and
- file-based pipeline use, where the same functions can write logs, plots, tables, and serialized objects when `saveOutputs = TRUE`.

The main high-level entry points are:

- `preprocessingMinfiEwasWater()`
- `svaEnmix()`
- `preprocessingPheno()`
- `methylationGLM_T1()`
- `methylationGLMM_T1T2()`
- `dnamReport()`

## Author(s)

**Maintainer:** Paul Ruiz <ruizpint@qut.edu.au> ([ORCID](#))

Authors:

- Divya Mehta ([ORCID](#))

## See Also

Useful links:

- <https://github.com/paulYRP/dnaEPICO>
- Report bugs at <https://github.com/paulYRP/dnaEPICO/issues>

---

 dnaEPICO\_dnamReport-class

*Result class returned by dnamReport*


---

### Description

Objects of class "dnaEPICO\_dnamReport" are list-based results returned by [dnamReport\(\)](#). They combine the prepared report inputs, render result, and final status metadata into one convenience object.

### Structure

**preparedReport** Object returned by [prepareDnamReportInputs\(\)](#).

**renderResult** Structured render metadata created by [dnamReport\(\)](#).

**status** Final status string such as "rendered", "skipped", or "failed".

**outputFile** Path to docs/index.html.

**errorMessage** Final render error or skip message when available, otherwise NULL.

**logFile** Resolved path to the optional log file, or NULL when logging was disabled.

### See Also

[dnamReport\(\)](#)

---

 dnaEPICO\_dnamReport\_prepared-class

*Result class returned by prepareDnamReportInputs*


---

### Description

Objects of class "dnaEPICO\_dnamReport\_prepared" are list-based results returned by [prepareDnamReportInputs\(\)](#). They capture normalized report paths, available figures, and logging metadata before rendering.

### Structure

**output** Requested output file name.

**outputDir** Normalized report output directory.

**outputFile** Normalized full path to the intended report output file.

**figDir** Normalized directory used by the report template for copied figures.

**figureInventory** Named list describing the available figures for each report section.

**missingFigureDirectories** Character vector of expected figure directories that were not present at preparation time.

**logFile** Resolved path to the optional log file, or NULL when logging was disabled.

**See Also**

[prepareDnamReportInputs\(\)](#)

---

dnaEPICO\_dnamReport\_render-class

*Result class returned by renderDnamReport*

---

**Description**

Objects of class "dnaEPICO\_dnamReport\_render" are list-based results returned by [renderDnamReport\(\)](#). They describe whether a prepared report was rendered, skipped, or failed.

**Structure**

**preparedReport** The input object supplied to [renderDnamReport\(\)](#).  
**status** Render status string such as "rendered", "skipped", or "failed".  
**renderedFile** Normalized path to the rendered PDF file when rendering succeeded, otherwise NULL.  
**errorMessage** Render error or skip message when available, otherwise NULL.  
**logFile** Resolved path to the optional log file, or NULL when logging was disabled.

**See Also**

[renderDnamReport\(\)](#)

---

dnaEPICO\_methylationGLM\_T1-class

*Result class returned by methylationGLM\_T1*

---

**Description**

Objects of class "dnaEPICO\_methylationGLM\_T1" are list-based results returned by [methylationGLM\\_T1\(\)](#). They collect the prepared analysis table, fitted models, summaries, diagnostics, annotations, and optional saved files.

**Structure**

**preparedData** Object returned by [prepareMethylationGLM\\_T1Data\(\)](#).  
**distributionPlots** Object returned by [plotMethylationGLM\\_T1Distributions\(\)](#).  
**modelFits** Object returned by [fitMethylationGLM\\_T1Models\(\)](#).  
**modelSummaries** Object returned by [summarizeMethylationGLM\\_T1Models\(\)](#).  
**significantCpGs** Object returned by [collectSignificantCpGsMethylationGLM\\_T1\(\)](#).  
**diagnosticPlots** Object returned by [plotMethylationGLM\\_T1Diagnostics\(\)](#).  
**annotation** Object returned by [annotateMethylationGLM\\_T1Summaries\(\)](#).  
**savedFiles** Object returned by [writeMethylationGLM\\_T1Outputs\(\)](#) when `saveOutputs = TRUE`, otherwise NULL.

**See Also**

[methylationGLM\\_T1\(\)](#)

---

dnaEPICO\_methylationGLMM\_T1T2-class

*Result class returned by methylationGLMM\_T1T2*

---

**Description**

Objects of class "dnaEPICO\_methylationGLMM\_T1T2" are list-based results returned by [methylationGLMM\\_T1T2\(\)](#). They collect the prepared longitudinal analysis table, fitted mixed models, summaries, diagnostics, annotations, and optional saved files.

**Structure**

**preparedData** Object returned by [prepareMethylationGLMM\\_T1T2Data\(\)](#).

**modelFits** Object returned by [fitMethylationGLMM\\_T1T2Models\(\)](#).

**modelSummaries** Object returned by [summarizeMethylationGLMM\\_T1T2Models\(\)](#).

**significantInteractions** Object returned by [collectSignificantInteractionsMethylationGLMM\\_T1T2\(\)](#).

**diagnosticPlots** Object returned by [plotMethylationGLMM\\_T1T2Diagnostics\(\)](#).

**annotation** Object returned by [annotateMethylationGLMM\\_T1T2Summaries\(\)](#).

**savedFiles** Object returned by [writeMethylationGLMM\\_T1T2Outputs\(\)](#) when `saveOutputs = TRUE`, otherwise NULL.

**See Also**

[methylationGLMM\\_T1T2\(\)](#)

---

dnaEPICO\_preprocessingMinfiEwasWater-class

*Result class returned by preprocessingMinfiEwasWater*

---

**Description**

Objects of class "dnaEPICO\_preprocessingMinfiEwasWater" are list-based results returned by [preprocessingMinfiEwasWater\(\)](#). They are lightweight S3-style containers rather than formal S4 classes.

**Structure**

- targets** Filtered phenotype table aligned to the retained samples.
- RGSet** Filtered RGChannelSet used in downstream preprocessing.
- rawData** Object returned by [buildRawMinfiEwasWater\(\)](#).
- assessment** Object returned by [assessSamplesMinfiEwasWater\(\)](#).
- sexData** Object returned by [predictSexMinfiEwasWater\(\)](#).
- normData** Object returned by [normalizeMinfiEwasWater\(\)](#).
- filterData** Object returned by [filterProbesMinfiEwasWater\(\)](#).
- metricsData** Object returned by [extractMetricsMinfiEwasWater\(\)](#).
- lcData** Object returned by [estimateLCMinfiEwasWater\(\)](#).
- logFile** Resolved path to the optional log file, or NULL when logging was disabled.

**See Also**

[preprocessingMinfiEwasWater\(\)](#)

---

dnaEPICO\_preprocessingPheno-class

*Result class returned by preprocessingPheno*

---

**Description**

Objects of class "dnaEPICO\_preprocessingPheno" are list-based results returned by [preprocessingPheno\(\)](#). They describe the phenotype data, methylation matrices, timepoint splits, longitudinal merges, and optional exported files.

**Structure**

- pheno** Phenotype table read from phenoFile.
- metricsData** Object returned by [loadMetricsPreprocessingPheno\(\)](#).
- timepointData** Object returned by [splitTimepointsPreprocessingPheno\(\)](#).
- combinedData** Object returned by [combineTimepointsPreprocessingPheno\(\)](#).
- clockFoundation** Object returned by [buildClockFoundationInputsPreprocessingPheno\(\)](#).
- savedFiles** Object returned by [writePreprocessingPhenoOutputs\(\)](#) when saveOutputs = TRUE, otherwise NULL.
- logFile** Resolved path to the optional log file, or NULL when logging was disabled.

**See Also**

[preprocessingPheno\(\)](#)

---

 dnaEPICO\_svaEnmix-class

*Result class returned by svaEnmix*


---

## Description

Objects of class "dnaEPICO\_svaEnmix" are list-based results returned by [svaEnmix\(\)](#). They collect the loaded inputs, surrogate-variable results, association-analysis summaries, and optional file outputs.

## Structure

**targets** Phenotype table read from phenoFile after any optional row subsetting.

**RGSet** Loaded RGChannelSet with sample names reset to match targets.

**svaData** Object returned by [estimateSvaEnmixControls\(\)](#).

**mergedPheno** Phenotype table returned by [mergeSvaTargetsEnmix\(\)](#) after surrogate variables were appended.

**analysisData** Object returned by [analyzeSvaEnmix\(\)](#).

**plotFiles** Named list of TIFF output paths for the SVA figures when saveOutputs = TRUE, otherwise NULL entries for plots not written.

**savedFiles** Object returned by [writeSvaEnmixOutputs\(\)](#) when saveOutputs = TRUE, otherwise NULL.

**logFile** Resolved path to the optional log file, or NULL when logging was disabled.

## See Also

[svaEnmix\(\)](#)

---

 dnamReport

*Generate a DNA methylation dashboard report*


---

## Description

Generate a DNA methylation dashboard report

## Usage

```
dnamReport(
  outputDir = "reports",
  phenoTab = NULL,
  enmixTab = file.path("figures", "preprocessingMinfiEwasWater", "enmix"),
  qcTab = file.path("figures", "preprocessingMinfiEwasWater", "qc"),
  svaTab = file.path("figures", "svaEnmix"),
```

```

metricTab = file.path("figures", "preprocessingMinfiEwasWater", "metrics"),
glmTab = NULL,
lmerTab = NULL,
logTab = outputDir,
verbose = FALSE,
logs = FALSE,
projectName = "dnaEPICO",
detPPath = NULL,
detPThreshold = 0.01,
cpgDetectionPath = NULL,
sampleDetectionPath = NULL,
logoPath = system.file("extdata", "dnaEPICO.svg", package = "dnaEPICO"),
imagePattern = "\\.(png|jpg|jpeg|gif|webp|svg|tif|tiff)$",
recursive = TRUE
)

```

### Arguments

outputDir	Character. Directory where the Quarto project is written.
phenoTab	Character or NULL. CSV file shown in the Data tab. When NULL, the path is inferred from the Makefile output layout.
enmixTab	Character. Directory containing ENmix quality-control figures.
qcTab	Character. Directory containing Quality Control figures.
svaTab	Character. Directory containing Batch Effect or SVA figures.
metricTab	Character. Directory containing Metrics figures.
glmTab	Character or NULL. XLSX workbook shown in the GLM Analysis tab. When NULL, the path is inferred from the Makefile output layout.
lmerTab	Character or NULL. XLSX workbook shown in the LMER Analysis tab. When NULL, the path is inferred from the Makefile output layout.
logTab	Character. Directory containing workflow logs shown in the Logs tab.
verbose	Logical. If TRUE, emit progress messages.
logs	Logical. If TRUE, write a report log.
projectName	Character. Name used for the generated Quarto project.
detPPath	Character or NULL. RData file containing the detection P-value matrix object detP, used to build the quality-control tables. When NULL, the path is inferred from the Makefile output layout.
detPThreshold	Numeric. Detection P-value threshold used when summarising the detP matrix.
cpgDetectionPath	Character or NULL. Optional fallback CpG detection summary CSV.
sampleDetectionPath	Character or NULL. Optional fallback sample detection summary CSV.
logoPath	Character. Path to the navbar logo. Defaults to the packaged inst/extdata/dnaEPICO.svg asset.
imagePattern	Character. Regular expression used to identify image files inside the section directories.
recursive	Logical. If TRUE, search section directories recursively.

**Value**

A list with class "dnaEPICO\_dnamReport".

**Examples**

```

report_root <- file.path(tempdir(), "dnaepico-dnam-report")
pheno_file <- file.path(
  report_root,
  "data",
  "model1",
  "preprocessingMinfiEwasWater",
  "phenolC.csv"
)
dir.create(dirname(pheno_file), recursive = TRUE, showWarnings = FALSE)
utils::write.csv(
  data.frame(
    UID = c("sample1", "sample2"),
    Timepoint = c(1, 2),
    Sex = c("F", "M")
  ),
  pheno_file,
  row.names = FALSE
)

result <- dnamReport(
  outputDir = file.path(report_root, "reports", "model1"),
  phenoTab = pheno_file,
  enmixTab = file.path(
    report_root,
    "figures",
    "model1",
    "preprocessingMinfiEwasWater",
    "enmix"
  ),
  qcTab = file.path(
    report_root,
    "figures",
    "model1",
    "preprocessingMinfiEwasWater",
    "qc"
  ),
  svaTab = file.path(report_root, "figures", "model1", "svaEnmix"),
  metricTab = file.path(
    report_root,
    "figures",
    "model1",
    "preprocessingMinfiEwasWater",
    "metrics"
  ),
  logTab = file.path(report_root, "logs", "model1")
)
result$status

```

estimateLC

*Estimate saliva cell proportions from DNA methylation beta values***Description**

Estimate cell-type proportions with the saliva reference panels bundled in dnaEPICO. This function keeps the original estimateLC() interface used by the package while using the internal reference files distributed in inst/extdata.

**Usage**

```
estimateLC(meth, ref, constrained = FALSE)
```

**Arguments**

meth	Numeric matrix of beta values with CpGs in rows and samples in columns. Row names must contain probe identifiers compatible with the selected reference.
ref	Character. Reference panel name. Supported values are "saliva" and "salivaEPIC".
constrained	Logical. If TRUE, estimated cell proportions are constrained to sum to one.

**Value**

A data.table with one row per sample and one column per estimated cell type.

**References**

Murat K, et al. Ewastools: Infinium Human Methylation BeadChip pipeline for population epigenetics integrated into Galaxy. *GigaScience*. 2020;9(5):giaa049. Houseman EA, Accomando WP, Koestler DC, et al. DNA methylation arrays as surrogate measures of cell mixture distribution. *BMC Bioinformatics*. 2012;13:86. Reinius LE, Acevedo N, Joerink M, et al. Differential DNA methylation in purified human blood cells: implications for cell lineage and studies on disease susceptibility. *PLoS One*. 2012;7(7):e41361. Bakulski KM, Feinberg JI, Andrews SV, et al. DNA methylation of cord blood cell types: applications for mixed cell birth studies. *Epigenetics*. 2016;11(5):354-362. de Goede OM, Razzaghian HR, Price EM, et al. Nucleated red blood cells impact DNA methylation and expression analyses of cord blood hematopoietic cells. *Clinical Epigenetics*. 2015;7:95. Gervin K, Salas LA, Bakulski KM, et al. Cell type specific DNA methylation in cord blood: a 450K reference data set and cell count-based validation of estimated cell type composition. *Epigenetics*. 2016;11(9):690-698. Gervin K, Salas LA, Bakulski KM, et al. Systematic evaluation and validation of reference and library selection methods for deconvolution of cord blood DNA methylation data. *bioRxiv*. 2019. doi:10.1101/570457. Salas LA, Koestler DC, Butler RA, et al. An optimized library for reference-based deconvolution of whole-blood biospecimens assayed using the Illumina HumanMethylationEPIC BeadArray. *Genome Biology*. 2018;19:64. Heiss JA, Just AC, Brenner H. Training a model for estimating leukocyte composition using whole-blood DNA methylation and cell counts as reference. *Epigenomics*. 2017;9(1):13-20. Middleton LYM, Dou J, Mill J, et al. Saliva cell type DNA methylation reference panel for epidemiology studies in children. 2020.

**Examples**

```

ref_file <- system.file("extdata", "saliva.txt", package = "dnaEPIC")
ref_panel <- as.matrix(utils::read.table(ref_file))
meth <- ref_panel[1:20, , drop = FALSE]
colnames(meth) <- c("sample1", "sample2")
estimateLC(
  meth = meth,
  ref = "saliva",
  constrained = FALSE
)

```

---

```
estimateLCMinfiEwasWater
```

*Estimate cell composition for preprocessing MinfiEwasWater*

---

**Description**

Estimate cell proportions from beta values using estimateLC() for saliva reference panels or ENmix::estimateCellProp() for other supported references, then merge the estimates into the phenotype table.

**Usage**

```

estimateLCMinfiEwasWater(
  beta,
  targets,
  lcRef = "salivaEPIC",
  phenoOrder = "Sample_Name;Timepoint;Sex;PredSex;Basename;Sentrix_ID;Sentrix_Position",
  constrained = FALSE,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_estimateLCMinfiEwasWater.txt"
)

```

**Arguments**

beta	Numeric matrix of beta values with probes in rows and samples in columns.
targets	Phenotype data frame aligned with the columns of beta.
lcRef	Character. Cell-composition reference. Internal saliva-based references supported through estimateLC() are "saliva" and "salivaEPIC". Other references are passed to ENmix::estimateCellProp().
phenoOrder	Character vector or semicolon-separated string describing the phenotype columns that should appear first in the merged phenoLC output.

constrained	Logical. Passed to estimateLC() when an internal saliva reference is used. If TRUE, estimated proportions are constrained to sum to one.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

### Value

A list with class "dnaEPICO\_minfiEwasWater\_lc" containing the cell proportion matrix, merged phenotype table, reference name, and method used.

### Examples

```
ref_file <- system.file("extdata", "saliva.txt", package = "dnaEPICO")
beta <- as.matrix(utils::read.table(ref_file))[1:20, , drop = FALSE]
colnames(beta) <- c("sample1", "sample2")
targets <- data.frame(
  Sample_Name = colnames(beta),
  Timepoint = c("T1", "T2"),
  stringsAsFactors = FALSE
)
lc_data <- estimateLCMinfiEwasWater(
  beta = beta,
  targets = targets,
  lcRef = "saliva",
  phenoOrder = "Sample_Name;Timepoint"
)
stopifnot(is.data.frame(lc_data$phenoLC))
```

---

estimateSvaEnmixControls

*Estimate surrogate variables from ENmix control probes*

---

### Description

Run `ENmix::ctrlsva()` on an `RGChannelSet` and return the surrogate variable matrix as an in-memory object.

### Usage

```
estimateSvaEnmixControls(
  RGSet,
  ctrlSvaPercVar = 0.9,
  ctrlSvaFlag = 1,
  verbose = FALSE,
```

```

    logs = FALSE,
    log_dir = NULL,
    log_file = "log_estimateSvaEnmixControls.txt"
  )

```

### Arguments

RGSet	An RGChannelSet.
ctrlSvaPercVar	Numeric. Proportion of variance explained by control probes, passed to ENmix::ctrlsva().
ctrlSvaFlag	Integer. Control-probe flag passed to ENmix::ctrlsva().
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

### Value

A list with class "dnaEPICO\_svaEnmix\_sva" containing the surrogate variable matrix and the parameters used to estimate it.

### Examples

```

ex <- dnaEPICO::exampleMinfiBaseDataDnaEpico()
sva_data <- estimateSvaEnmixControls(
  RGSet = ex$RGSet,
  ctrlSvaPercVar = 0.5,
  ctrlSvaFlag = 1,
  verbose = FALSE,
  logs = FALSE
)
sva_data$K

```

---

extractMake

*Copy dnaEPICO Makefile to a user directory*

---

### Description

Copies the example Makefile pipeline shipped with dnaEPICO to a user-specified directory for local execution or modification.

### Usage

```
extractMake(destDir, overwrite = FALSE)
```

**Arguments**

destDir            Character. Destination directory where the Makefile will be copied.  
 overwrite        Logical. Whether to overwrite an existing Makefile in destDir. The default is FALSE.

**Value**

Character scalar containing the path to the copied Makefile.

**Examples**

```
tmp <- file.path(tempdir(), "dnaEPICO-make-example")
dir.create(tmp, recursive = TRUE, showWarnings = FALSE)
makefile_path <- extractMake(
  destDir = tmp,
  overwrite = TRUE
)
stopifnot(file.exists(makefile_path))
```

---

extractMetricsMinfiEwasWater

*Extract beta, M, and copy-number matrices from a filtered object*

---

**Description**

Extract beta, M, and copy-number matrices from a filtered object

**Usage**

```
extractMetricsMinfiEwasWater(
  filteredData,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_extractMetricsMinfiEwasWater.txt"
)
```

**Arguments**

filteredData    Object returned by filterProbesMinfiEwasWater().  
 verbose        Logical. If TRUE, emit progress messages with message().  
 logs           Logical. If TRUE, write the same messages to a log file.  
 log\_dir        Character or NULL. Directory used for the log file when logs = TRUE.  
 log\_file       Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_minfiEwasWater\_metrics" containing beta, m, and cn.

**Examples**

```
ex <- dnaEPICO::exampleMinfiMetricsStateDnaEpico()
metrics_data <- extractMetricsMinfiEwasWater(
  filteredData = ex$filteredData,
  verbose = FALSE,
  logs = FALSE
)
names(metrics_data)
```

---

filterProbesMinfiEwasWater

*Filter probes from a normalized methylation object*

---

**Description**

Apply detection P-value, chromosome, SNP, and cross-reactive probe filters to the primary normalized object and return the filtered result.

**Usage**

```
filterProbesMinfiEwasWater(
  normData,
  RGSet,
  pvalThreshold = 0.01,
  chrToRemove = "chrX,chrY",
  snpsToRemove = "SBE,CpG",
  mafThreshold = 0.1,
  crossReactivePath,
  detPtype = "m+u",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_filterProbesMinfiEwasWater.txt"
)
```

**Arguments**

normData	Object returned by normalizeMinfiEwasWater().
RGSet	Filtered RGChannelSet aligned with normData.
pvalThreshold	Numeric. Probes must have detection P values below this threshold in all samples to be retained.

chrToRemove	Character vector or comma-separated string of chromosome names to remove, for example "chrX, chrY".
snpsToRemove	Character vector or comma-separated string of SNP probe types to remove, for example "SBE, CpG".
mafThreshold	Numeric. Minor allele frequency threshold passed to <code>minfi::dropLociWithSnps()</code> .
crossReactivePath	Character. Path to a CSV file containing a ProbeID column of cross-reactive probes to remove.
detPtype	Character. Detection P-value mode passed to <code>minfi::detectionP()</code> for the probe filter. Common values in minfi workflows are "m+u" and "negative".
verbose	Logical. If TRUE, emit progress messages with <code>message()</code> .
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
log_file	Character. File name used when <code>logs = TRUE</code> .

### Value

A list with class "dnaEPICO\_minfiEwasWater\_filter" containing the filtered object and counts for each filtering stage.

### Examples

```
ex <- dnaEPICO::exampleMinfiWorkflowStateDnaEpico()
filtered_data <- filterProbesMinfiEwasWater(
  normData = ex$normData,
  RGSet = ex$sampleData$RGSet,
  pvalThreshold = 1,
  chrToRemove = "chrY",
  snpsToRemove = "SBE",
  mafThreshold = 1,
  crossReactivePath = ex$crossReactivePath,
  detPtype = "m+u",
  verbose = FALSE,
  logs = FALSE
)
filtered_data$counts[["crossReactive"]]
```

---

```
filterSamplesMinfiEwasWater
```

*Filter failed samples from an RGSet and phenotype table*

---

### Description

Remove failed samples identified during sample assessment and return the filtered `RGChannelSet` together with the aligned phenotype table.

**Usage**

```

filterSamplesMinfiEwasWater(
  RGSet,
  targets,
  failedSamples = character(0),
  SampleID = "Sample_Name",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_filterSamplesMinfiEwasWater.txt"
)

```

**Arguments**

RGSet	An RGChannelSet.
targets	Data frame containing phenotype information.
failedSamples	Character vector of sample identifiers to remove.
SampleID	Character. Name of the sample identifier column in targets.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_minfiEwasWater\_samples" containing the filtered RGSet, aligned phenotype table, and failed sample identifiers.

**Examples**

```

ex <- dnaEPICO::exampleMinfiBaseDataDnaEpico()
filtered_samples <- filterSamplesMinfiEwasWater(
  RGSet = ex$RGSet,
  targets = ex$targets,
  failedSamples = ex$targets$Sample_Name[1],
  SampleID = "Sample_Name",
  verbose = FALSE,
  logs = FALSE
)
nrow(filtered_samples$targets)

```

---

fitMethylationGLM\_T1Models

*Fit CpG-wise Gaussian GLMs for one-timepoint methylation analyses*


---

## Description

Fit one Gaussian GLM per CpG for each phenotype requested in the object returned by prepareMethylationGLM\_T1Data().

## Usage

```
fitMethylationGLM_T1Models(
  preparedData,
  nCores = 1L,
  libPath = NULL,
  glmLibs = "glm2",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLM_T1.txt"
)
```

## Arguments

preparedData	Object returned by prepareMethylationGLM_T1Data().
nCores	Integer. Number of worker processes to use.
libPath	Character vector or NULL. Optional library paths forwarded to worker processes.
glmLibs	Character vector or comma-separated string of package names to check on worker processes. The default is "glm2".
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

## Value

A list with class "dnaEPICO\_methylationGLM\_T1\_models" containing fitted model lists, model formulas, and counts of failed CpG fits.

## Examples

```
ex <- dnaEPICO:::exampleMethylationGLMStateDnaEpico()
model_results <- fitMethylationGLM_T1Models(
  preparedData = ex$preparedData,
  nCores = 1,
  verbose = FALSE,
  logs = FALSE
```

```
)
names(model_results$fits)
```

---

```
fitMethylationGLMM_T1T2Models
```

*Fit CpG-wise mixed-effects models for longitudinal methylation analyses*

---

### Description

Fit one linear mixed-effects model per CpG for each phenotype requested in the object returned by `prepareMethylationGLMM_T1T2Data()`.

### Usage

```
fitMethylationGLMM_T1T2Models(
  preparedData,
  nCores = 1L,
  libPath = NULL,
  lmeLibs = "lme4,lmerTest",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLMM_T1T2.txt"
)
```

### Arguments

<code>preparedData</code>	Object returned by <code>prepareMethylationGLMM_T1T2Data()</code> .
<code>nCores</code>	Integer. Number of worker processes to use.
<code>libPath</code>	Character vector or NULL. Optional library paths forwarded to worker processes.
<code>lmeLibs</code>	Character vector or comma-separated string of package names to check on worker processes. The default is "lme4,lmerTest".
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

### Value

A list with class "dnaEPIC0\_methylationGLMM\_T1T2\_models" containing fitted model lists, model formulas, and counts of failed CpG fits.

**Examples**

```

ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
model_results <- fitMethylationGLMM_T1T2Models(
  preparedData = ex$preparedData,
  nCores = 1,
  verbose = FALSE,
  logs = FALSE
)
names(model_results$fits)

```

---

loadMetricsPreprocessingPheno

*Load methylation metric matrices for preprocessingPheno*

---

**Description**

Load the metric matrices generated by preprocessingMinfiEwasWater() and return them as a single in-memory object for downstream phenotype alignment.

**Usage**

```

loadMetricsPreprocessingPheno(
  betaPath,
  mPath,
  cnPath,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_loadMetricsPreprocessingPheno.txt"
)

```

**Arguments**

betaPath	Character. Path to the saved beta-value object.
mPath	Character. Path to the saved M-value object.
cnPath	Character. Path to the saved copy-number object.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_preprocessingPheno\_metrics" containing beta, m, and cn.

**Examples**

```

ex <- dnaEPICO::examplePreprocessingPhenoStateDnaEpico()
metrics_data <- loadMetricsPreprocessingPheno(
  betaPath = ex$betaPath,
  mPath = ex$mPath,
  cnPath = ex$cnPath,
  verbose = FALSE,
  logs = FALSE
)
names(metrics_data)

```

---

mergeSvaTargetsEnmix    *Merge surrogate variables into the phenotype table*

---

**Description**

Merge the surrogate variable matrix back into the phenotype table while preserving the original row order of targets.

**Usage**

```

mergeSvaTargetsEnmix(
  targets,
  sva,
  SampleID = "Sample_Name",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_mergeSvaTargetsEnmix.txt"
)

```

**Arguments**

targets	Phenotype data frame aligned with the samples in sva.
sva	Numeric matrix of surrogate variables with samples in rows.
SampleID	Character. Name of the phenotype sample identifier column.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A phenotype data frame with the surrogate variables appended.

**Examples**

```

ex <- dnaEPIC0::exampleSvaAnalysisStateDnaEpico()
merged_pheno <- mergeSvaTargetsEnmix(
  targets = ex$targets,
  sva = ex$sva,
  SampleID = "Sample_Name",
  verbose = FALSE,
  logs = FALSE
)
colnames(merged_pheno)[seq_len(4)]

```

---

methylationGLM\_T1

*Fit CpG-wise GLMs for one-timepoint methylation analyses*


---

**Description**

methylationGLM\_T1() is the high-level coordinator for the one-timepoint GLM stage of the dnaEPIC0 workflow. It prepares the merged phenotype-plus-beta input, optionally creates exploratory plots, fits one Gaussian GLM per CpG for each requested phenotype, extracts CpG-level summaries, optionally collects significant CpG coefficient tables, generates diagnostic plots, annotates the combined summary table, and optionally writes legacy-style outputs to disk. The default behavior is now in-memory and quiet, which makes the function easier to compose with other package functions and more aligned with typical Bioconductor usage.

**Usage**

```

methylationGLM_T1(
  inputPheno = "rData/preprocessingPheno/mergeData/phenoBetaT1.RData",
  outputLogs = "logs",
  outputRData = "rData/methylationGLM_T1/models",
  outputPlots = "figures/methylationGLM_T1",
  phenotypes = c("DASS_Depression", "DASS_Anxiety", "DASS_Stress", "PCL5_TotalScore",
    "MHCSF_TotalScore", "BRS_TotalScore"),
  covariates = "Sex, Age, Ethnicity, TraumaDefinition, Leukocytes, Epithelial.cells",
  factorVars = "Sex, Ethnicity, TraumaDefinition",
  cpGPrefix = "cg",
  cpGLimit = NA,
  nCores = 32,
  plotWidth = 2000,
  plotHeight = 1000,
  plotDPI = 150,
  interactionTerm = NULL,
  libPath = NULL,
  glmLibs = "glm2",
  prsMap = NULL,
  summaryPval = NA,

```

```

summaryResidualSD = TRUE,
saveSignificantCpGs = FALSE,
significantCpGDir = "preliminaryResults/cpgs/methylationGLM_T1",
significantCpGPval = 0.05,
saveTxtSummaries = TRUE,
chunkSize = NULL,
summaryTxtDir = "preliminaryResults/summary/methylationGLM_T1/glm",
fdrThreshold = 0.05,
padjmethod = "fdr",
annotationPackage = "IlluminaHumanMethylationEPICv2anno.20a1.hg38",
annotationCols = c("Name", "chr", "pos", "UCSC_RefGene_Group", "UCSC_RefGene_Name",
  "Relation_to_Island", "GencodeV41_Group"),
annotatedGLMOut = "data/methylationGLM_T1",
display = FALSE,
verbose = FALSE,
logs = FALSE,
saveOutputs = FALSE
)

```

## Arguments

inputPheno	Character. Path to the merged phenotype-plus-beta .RData or .rds object created by preprocessingPheno(). The default points to the timepoint-1 object produced by the package workflow.
outputLogs	Character. Directory used for optional log files.
outputRData	Character. Directory used for optional serialized model and summary outputs.
outputPlots	Character. Directory used for optional TIFF plots.
phenotypes	Character vector or comma-separated phenotype variables to model.
covariates	Character. Comma-separated covariate variables included in each GLM.
factorVars	Character. Comma-separated variables that should be treated as factors before modeling.
cpgPrefix	Character. Prefix used to identify methylation columns in the merged phenotype-plus-beta input object. The default is "cg".
cpgLimit	Integer or NA. Maximum number of CpGs to analyse. Use NA to keep all CpGs matching cpgPrefix.
nCores	Integer. Number of worker processes to use while fitting models and extracting summaries.
plotWidth	Integer. TIFF width in pixels when plots are written to disk.
plotHeight	Integer. TIFF height in pixels when plots are written to disk.
plotDPI	Integer. TIFF resolution in DPI when plots are written to disk.
interactionTerm	Character or NULL. Optional interaction term. When supplied and present in the input data, the phenotype is modeled together with its interaction against this variable.

libPath	Character vector or NULL. Optional library paths forwarded to worker processes. By default, the current <code>.libPaths()</code> are used.
glmLibs	Character. Comma-separated package names to validate on worker processes. The default is "glm2".
prsMap	Character or NULL. Optional phenotype-to-PRS mapping in the form "Phenotype1:PRS_1,Phenotype2:PRS_2".
summaryPval	Numeric or NA. Optional p-value threshold applied to the returned CpG summary tables. Use NA to keep all summary rows.
summaryResidualSD	Logical. If TRUE, append residual standard deviations to the CpG summary tables and residual diagnostic plots.
saveSignificantCpGs	Logical. If TRUE, collect significant CpG coefficient tables in the returned object and optionally write them to disk when <code>saveOutputs = TRUE</code> .
significantCpGDir	Character. Directory used for optional significant CpG coefficient tables.
significantCpGPval	Numeric. P-value threshold used to collect or write significant CpG coefficient tables.
saveTxtSummaries	Logical. If TRUE and <code>saveOutputs = TRUE</code> , write tab-delimited summary tables to <code>summaryTxtDir</code> .
chunkSize	Integer or NULL. Number of CpGs processed per summary extraction chunk. NULL chooses a value automatically.
summaryTxtDir	Character. Directory used for optional tab-delimited GLM summary tables.
fdrThreshold	Numeric. False-discovery-rate threshold used to highlight CpGs in the residual-significance diagnostic plots.
padjmethod	Character. P-value adjustment method passed to <code>stats::p.adjust()</code> . The default is "fdr".
annotationPackage	Character. Annotation package or object name passed to <code>minfi::getAnnotation()</code> , for example "IlluminaHumanMethylationEPICv2anno.20a1.hg38".
annotationCols	Character vector or comma-separated annotation columns to append to the combined GLM summary table. Available columns depend on the selected annotation package.
annotatedGLMOut	Character. Directory used for the optional annotated GLM summary XLSX workbook.
display	Logical. If TRUE, draw exploratory and diagnostic plots on the active graphics device.
verbose	Logical. If TRUE, emit progress messages with <code>message()</code> . The default is FALSE, so the function is quiet unless requested.
logs	Logical. If TRUE, write the same progress messages to <code>file.path(outputLogs, "log_methylationGLM_T1.txt")</code> .

**saveOutputs** Logical. If TRUE, write optional serialized model files, summary tables, significant-CpG tables, annotated results, and TIFF plots to the requested output directories. The default is FALSE, so the function returns in-memory results without writing files.

## Value

A list with class "dnaEPICO\_methylationGLM\_T1".

**preparedData** Object returned by [prepareMethylationGLM\\_T1Data\(\)](#) containing the merged phenotype-plus-beta analysis table and modeling metadata.

**distributionPlots** Object returned by [plotMethylationGLM\\_T1Distributions\(\)](#) describing any exploratory plots that were generated or written.

**modelFits** Object returned by [fitMethylationGLM\\_T1Models\(\)](#) containing the per-phenotype CpG model fits.

**modelSummaries** Object returned by [summarizeMethylationGLM\\_T1Models\(\)](#) containing the combined CpG summary tables used for reporting and annotation.

**significantCpGs** Object returned by [collectSignificantCpGsMethylationGLM\\_T1\(\)](#) containing optional phenotype-specific significant-CpG tables.

**diagnosticPlots** Object returned by [plotMethylationGLM\\_T1Diagnostics\(\)](#) describing the diagnostic plot objects and any written TIFF files.

**annotation** Object returned by [annotateMethylationGLM\\_T1Summaries\(\)](#) containing the annotated combined summary table.

**savedFiles** Object returned by [writeMethylationGLM\\_T1Outputs\(\)](#) when `saveOutputs = TRUE`, otherwise NULL.

See [dnaEPICO\\_methylationGLM\\_T1](#) for a class-level overview.

## See Also

[dnaEPICO\\_methylationGLM\\_T1](#)

## Examples

```
if (requireNamespace("IlluminaHumanMethylation450kanno.ilmn12.hg19", quietly = TRUE)) {
  tmp <- tempdir()
  toy_path <- file.path(tmp, "phenoBetaT1.RData")
  phenoBT1 <- data.frame(
    Sample_Name = c("S1", "S2", "S3", "S4"),
    status = factor(c("Case", "Case", "Control", "Control")),
    sex = factor(c("F", "M", "F", "M")),
    cg00000029 = c(0.20, 0.25, 0.22, 0.27),
    cg00000108 = c(0.60, 0.55, 0.52, 0.58),
    check.names = FALSE
  )
  save(phenoBT1, file = toy_path)

  result <- methylationGLM_T1(
    inputPheno = toy_path,
```

```

phenotypes = "status",
covariates = "sex",
factorVars = "status,sex",
cpgLimit = 2,
nCores = 1,
summaryPval = 1,
annotationPackage = "IlluminaHumanMethylation450kanno.ilmn12.hg19",
annotationCols = "Name,chr,pos",
display = FALSE,
verbose = FALSE,
logs = FALSE,
saveOutputs = FALSE
)

class(result)
}

```

---

methylationGLMM\_T1T2 *Fit CpG-wise linear mixed-effects models for longitudinal methylation analyses*

---

## Description

methylationGLMM\_T1T2() is the high-level coordinator for the longitudinal linear mixed-effects stage of the dnaEPICo workflow. It prepares the merged phenotype-plus-beta input, fits one mixed-effects model per CpG for each requested phenotype, extracts phenotype-specific coefficient summaries, optionally collects significant interaction tables, generates diagnostic plots, annotates the combined summary table, and optionally writes legacy-style outputs to disk. The default behavior is now in-memory and quiet, which makes the function easier to compose with other package functions and more aligned with typical Bioconductor usage.

## Usage

```

methylationGLMM_T1T2(
  inputPheno = "rData/preprocessingPheno/mergeData/phenoBetaT1T2.RData",
  outputLogs = "logs",
  outputRData = "rData/methylationGLMM_T1T2/models",
  outputPlots = "figures/methylationGLMM_T1T2",
  personVar = "person",
  timeVar = "Timepoint",
  phenotypes = c("DASS_Depression", "DASS_Anxiety", "DASS_Stress", "PCL5_TotalScore",
    "MHCSF_TotalScore", "BRS_TotalScore"),
  covariates = "Sex, Age, Ethnicity, TraumaDefinition, Leukocytes, Epithelial.cells",
  factorVars = "Sex, Ethnicity, TraumaDefinition, Timepoint",
  lmeLibs = "lme4, lmerTest",
  prsMap = NULL,
  libPath = NULL,

```

```

cpgPrefix = "cg",
cpgLimit = NA,
nCores = 32,
summaryPval = NA,
plotWidth = 2000,
plotHeight = 1000,
plotDPI = 150,
interactionTerm = NULL,
saveSignificantInteractions = TRUE,
significantInteractionDir = "preliminaryResults/cpgs/methylationGLMM_T1T2",
significantInteractionPval = 0.05,
saveTxtSummaries = TRUE,
chunkSize = NULL,
summaryTxtDir = "preliminaryResults/summary/methylationGLMM_T1T2/lmer",
fdrThreshold = 0.05,
padjmethod = "fdr",
annotationPackage = "IlluminaHumanMethylationEPICv2anno.20a1.hg38",
annotationCols = c("Name", "chr", "pos", "UCSC_RefGene_Group", "UCSC_RefGene_Name",
  "Relation_to_Island", "GencodeV41_Group"),
annotatedLMEOut = "data/methylationGLMM_T1T2",
display = FALSE,
verbose = FALSE,
logs = FALSE,
saveOutputs = FALSE
)

```

### Arguments

inputPheno	Character. Path to the merged longitudinal phenotype-plus-beta .RData or .rds object created by preprocessingPheno(). The default points to the combined timepoint object produced by the package workflow.
outputLogs	Character. Directory used for optional log files.
outputRData	Character. Directory used for optional serialized mixed-model and summary outputs.
outputPlots	Character. Directory used for optional TIFF diagnostic plots.
personVar	Character. Subject identifier variable used for the random intercept. When this column is missing, it is derived from SID using the package's existing sample naming convention.
timeVar	Character. Name of the longitudinal time variable included as a fixed effect in every model.
phenotypes	Character vector or comma-separated phenotype variables to model.
covariates	Character. Comma-separated fixed-effect covariates included in every mixed model.
factorVars	Character. Comma-separated variables that should be coerced to factors before modeling. This usually includes categorical covariates and timeVar.
lmeLibs	Character. Comma-separated package names to validate on worker processes. The default is "lme4, lmerTest".

prsMap	Character or NULL. Optional phenotype-to-PRS mapping in the form "Phenotype1:PRS_1,Phenotype2:PRS_2".
libPath	Character vector or NULL. Optional library paths forwarded to worker processes. By default, the current <code>.libPaths()</code> are used.
cpgPrefix	Character. Prefix used to identify methylation columns in the merged phenotype-plus-beta input object. The default is "cg".
cpgLimit	Integer or NA. Maximum number of CpGs to analyse. Use NA to keep all CpGs matching <code>cpgPrefix</code> .
nCores	Integer. Number of worker processes to use while fitting models and extracting summaries.
summaryPval	Numeric or NA. Optional p-value threshold applied to the returned longitudinal CpG summary tables. Use NA to keep all summary rows.
plotWidth	Integer. TIFF width in pixels when plots are written to disk.
plotHeight	Integer. TIFF height in pixels when plots are written to disk.
plotDPI	Integer. TIFF resolution in DPI when plots are written to disk.
interactionTerm	Character or NULL. Optional interaction term. When supplied and present in the input data, the phenotype is modeled together with its interaction against this variable.
saveSignificantInteractions	Logical. If TRUE, collect coefficient tables for CpGs passing <code>significantInteractionPval</code> in the returned object and optionally write them to disk when <code>saveOutputs = TRUE</code> .
significantInteractionDir	Character. Directory used for optional significant-interaction coefficient tables.
significantInteractionPval	Numeric. P-value threshold used to collect or write significant interaction coefficient tables.
saveTxtSummaries	Logical. If TRUE and <code>saveOutputs = TRUE</code> , write tab-delimited summary tables to <code>summaryTxtDir</code> .
chunkSize	Integer or NULL. Number of CpGs processed per summary extraction chunk. NULL chooses a value automatically.
summaryTxtDir	Character. Directory used for optional tab-delimited LME summary tables.
fdrThreshold	Numeric. False-discovery-rate threshold used to highlight CpGs in the residual-significance diagnostic plots.
padjmethod	Character. P-value adjustment method passed to <code>stats::p.adjust()</code> . The default is "fdr".
annotationPackage	Character. Annotation package or object name passed to <code>minfi::getAnnotation()</code> , for example "IlluminaHumanMethylationEPICv2anno.20a1.hg38".
annotationCols	Character vector or comma-separated annotation columns to append to the combined LME summary table. Available columns depend on the selected annotation package.

annotatedLMEOut	Character. Directory used for the optional annotated LME summary XLSX workbook.
display	Logical. If TRUE, draw diagnostic plots on the active graphics device.
verbose	Logical. If TRUE, emit progress messages with <code>message()</code> . The default is FALSE, so the function is quiet unless requested.
logs	Logical. If TRUE, write the same progress messages to <code>file.path(outputLogs, "log_methylationGLMM_T1T2.txt")</code> .
saveOutputs	Logical. If TRUE, write optional serialized model files, summary tables, significant-interaction tables, annotated results, and TIFF plots to the requested output directories. The default is FALSE, so the function returns in-memory results without writing files.

### Value

A list with class `"dnaEPICO_methylationGLMM_T1T2"`.

**preparedData** Object returned by `prepareMethylationGLMM_T1T2Data()` containing the merged longitudinal phenotype-plus-beta analysis table and modeling metadata.

**modelFits** Object returned by `fitMethylationGLMM_T1T2Models()` containing the per-phenotype CpG mixed-effects model fits.

**modelSummaries** Object returned by `summarizeMethylationGLMM_T1T2Models()` containing the combined CpG summary tables used for reporting and annotation.

**significantInteractions** Object returned by `collectSignificantInteractionsMethylationGLMM_T1T2()` containing optional phenotype-specific significant-interaction tables.

**diagnosticPlots** Object returned by `plotMethylationGLMM_T1T2Diagnostics()` describing the diagnostic plot objects and any written TIFF files.

**annotation** Object returned by `annotateMethylationGLMM_T1T2Summaries()` containing the annotated combined summary table.

**savedFiles** Object returned by `writeMethylationGLMM_T1T2Outputs()` when `saveOutputs = TRUE`, otherwise NULL.

See [dnaEPICO\\_methylationGLMM\\_T1T2](#) for a class-level overview.

### See Also

[dnaEPICO\\_methylationGLMM\\_T1T2](#)

### Examples

```
if (
  requireNamespace("IlluminaHumanMethylation450kanno.ilmn12.hg19", quietly = TRUE) &&
  requireNamespace("lmerTest", quietly = TRUE)
) {
  tmp <- tempdir()
  toy_path <- file.path(tmp, "phenoBetaT1T2.RData")
  phenoBT1T2 <- data.frame(
    SID = c("P1A", "P1B", "P2A", "P2B", "P3A", "P3B", "P4A", "P4B"),
```

```

person = c(1, 1, 2, 2, 3, 3, 4, 4),
Timepoint = factor(c("1", "2", "1", "2", "1", "2", "1", "2")),
score = c(10, 12, 9, 11, 13, 14, 8, 9),
sex = factor(c("F", "F", "M", "M", "F", "F", "M", "M")),
cg00000029 = c(0.25, 0.27, 0.20, 0.22, 0.30, 0.31, 0.18, 0.20),
cg00000108 = c(0.50, 0.53, 0.55, 0.57, 0.48, 0.49, 0.60, 0.61),
check.names = FALSE
)
save(phenoBT1T2, file = toy_path)

result <- methylationGLMM_T1T2(
  inputPheno = toy_path,
  phenotypes = "score",
  covariates = "sex",
  factorVars = "sex,Timepoint",
  cpGLimit = 2,
  nCores = 1,
  summaryPval = 1,
  annotationPackage = "IlluminaHumanMethylation450kanno.ilmn12.hg19",
  annotationCols = "Name,chr,pos",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE,
  saveOutputs = FALSE
)

class(result)
}

```

---

```
normalizeMinfiEwasWater
```

*Normalize filtered samples with minfi and wateRmelon methods*

---

## Description

Apply one or more supported normalization methods to a filtered RGSet and return all normalized objects together in a single result object.

## Usage

```

normalizeMinfiEwasWater(
  sampleData,
  sexColumn = "Sex",
  normMethods = "adjustedfunnorm",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_normalizeMinfiEwasWater.txt"
)

```

**Arguments**

sampleData	Object returned by filterSamplesMinfiEwasWater().
sexColumn	Character. Name of the phenotype column used as the optional sex covariate for normalization methods that support it.
normMethods	Character vector or semicolon-separated string of normalization methods. Supported values are "adjustedfunnorm", "funnorm", "illumina", "quantile", and "swan".
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_minfiEwasWater\_norm" containing the requested normalized objects and the first method as primary.

**Examples**

```
ex <- dnaEPICO:::exampleMinfiBaseDataDnaEpico()
sample_data <- filterSamplesMinfiEwasWater(
  RGSet = ex$RGSet,
  targets = ex$targets,
  failedSamples = character(0),
  SampleID = "Sample_Name",
  verbose = FALSE,
  logs = FALSE
)
norm_data <- normalizeMinfiEwasWater(
  sampleData = sample_data,
  sexColumn = "Sex",
  normMethods = "quantile",
  verbose = FALSE,
  logs = FALSE
)
names(norm_data$normalized)
```

---

plotAssessmentMinfiEwasWater

*Plot quality-assessment outputs for preprocessingMinfiEwasWater*

---

**Description**

Draw either the minfi QC plot or the detection P-value plot from an assessment object returned by assessSamplesMinfiEwasWater().

**Usage**

```
plotAssessmentMinfiEwasWater(
  assessment,
  plot = c("qc", "detection"),
  display = FALSE,
  file = NULL,
  width = 2000L,
  height = 1000L,
  res = 150L,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_plotAssessmentMinfiEwasWater.txt"
)
```

**Arguments**

<code>assessment</code>	Object returned by <code>assessSamplesMinfiEwasWater()</code> .
<code>plot</code>	Character. Plot type: "qc" or "detection".
<code>display</code>	Logical. If TRUE, draw the plot on the active graphics device.
<code>file</code>	Character or NULL. TIFF file written when supplied.
<code>width</code>	Integer. TIFF width in pixels when file is supplied.
<code>height</code>	Integer. TIFF height in pixels when file is supplied.
<code>res</code>	Integer. TIFF resolution in DPI when file is supplied.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

**Value**

Invisibly returns the saved TIFF path when `file` is supplied, otherwise NULL.

**Examples**

```
assessment <- list(
  meanDetP = c(S1 = 0.01, S2 = 0.02, S3 = 0.04),
  detPThreshold = 0.05
)
plotAssessmentMinfiEwasWater(
  assessment = assessment,
  plot = "detection",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
```

---

plotCtrlMinfiEwasWater

*Plot ENmix control images from an RGSet*


---

### Description

Call `ENmix::plotCtrl()` for a supplied `RGSet`. This function only writes files when `output_dir` is provided because `ENmix::plotCtrl()` produces JPG files on disk rather than returning a plot object.

### Usage

```
plotCtrlMinfiEwasWater(
  RGSet,
  output_dir = NULL,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_plotCtrlMinfiEwasWater.txt"
)
```

### Arguments

<code>RGSet</code>	An <code>RGChannelSet</code> .
<code>output_dir</code>	Character or <code>NULL</code> . Directory where ENmix control JPG files should be written. If <code>NULL</code> , the function returns without writing files.
<code>verbose</code>	Logical. If <code>TRUE</code> , emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If <code>TRUE</code> , write the same messages to a log file.
<code>log_dir</code>	Character or <code>NULL</code> . Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

### Value

Invisibly returns `output_dir`.

### Examples

```
ex <- dnaEPICO::exampleMinfiBaseDataDnaEpico()
output_dir <- file.path(tempdir(), "enmix-control-plots")
plotCtrlMinfiEwasWater(
  RGSet = ex$RGSet,
  output_dir = output_dir,
  verbose = FALSE,
  logs = FALSE
)
dir.exists(output_dir)
```

---

```
plotMethylationGLM_T1Diagnostics
```

*Plot diagnostic summaries for one-timepoint methylation GLMs*

---

## Description

Create Q-Q and residual-diagnostic plots from the CpG summary tables returned by `summarizeMethylationGLM_T1Models()`

## Usage

```
plotMethylationGLM_T1Diagnostics(
  modelSummaries,
  preparedData,
  fdrThreshold = 0.05,
  padjmethod = "fdr",
  outputDir = NULL,
  plotWidth = 2000L,
  plotHeight = 1000L,
  plotDPI = 150L,
  display = FALSE,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLM_T1.txt"
)
```

## Arguments

<code>modelSummaries</code>	Object returned by <code>summarizeMethylationGLM_T1Models()</code> .
<code>preparedData</code>	Object returned by <code>prepareMethylationGLM_T1Data()</code> .
<code>fdrThreshold</code>	Numeric. False-discovery-rate threshold used to highlight CpGs in the diagnostic plots.
<code>padjmethod</code>	Character. P-value adjustment method passed to <code>stats::p.adjust()</code> .
<code>outputDir</code>	Character or NULL. Directory used for TIFF files. When NULL, plots are returned in memory only.
<code>plotWidth</code>	Integer. TIFF width in pixels when plots are written to disk.
<code>plotHeight</code>	Integer. TIFF height in pixels when plots are written to disk.
<code>plotDPI</code>	Integer. TIFF resolution in DPI when plots are written to disk.
<code>display</code>	Logical. If TRUE, draw plots on the active graphics device.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

**Value**

A list with class "dnaEPICO\_methylationGLM\_T1\_diagnostic\_plots" containing the generated ggplot2 objects, genomic inflation factors, and any saved TIFF file paths.

**Examples**

```
ex <- dnaEPICO::exampleMethylationGLMStateDnaEpico()
diagnostic_plots <- plotMethylationGLM_T1Diagnostics(
  modelSummaries = ex$modelSummaries,
  preparedData = ex$preparedData,
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
names(diagnostic_plots$plots)
```

---

plotMethylationGLM\_T1Distributions

*Plot phenotype and covariate distributions for one-timepoint GLM analyses*

---

**Description**

Create phenotype, factor-variable, and numeric-covariate distribution plots from the object returned by prepareMethylationGLM\_T1Data().

**Usage**

```
plotMethylationGLM_T1Distributions(
  preparedData,
  plotWidth = 2000L,
  plotHeight = 1000L,
  plotDPI = 150L,
  outputDir = NULL,
  display = FALSE,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLM_T1.txt"
)
```

**Arguments**

preparedData	Object returned by prepareMethylationGLM_T1Data().
plotWidth	Integer. TIFF width in pixels when plots are written to disk.
plotHeight	Integer. TIFF height in pixels when plots are written to disk.

plotDPI	Integer. TIFF resolution in DPI when plots are written to disk.
outputDir	Character or NULL. Directory used for TIFF files. When NULL, plots are returned in memory only.
display	Logical. If TRUE, draw plots on the active graphics device.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_methylationGLM\_T1\_distribution\_plots" containing the generated ggplot2 objects and any saved TIFF file paths.

**Examples**

```
ex <- dnaEPICO::exampleMethylationGLMStateDnaEpico()
distribution_plots <- plotMethylationGLM_T1Distributions(
  preparedData = ex$preparedData,
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
names(distribution_plots)
```

---

plotMethylationGLMM\_T1T2Diagnostics

*Plot longitudinal mixed-effects model diagnostics*

---

**Description**

Create Q-Q and standard-error diagnostic plots from the mixed-effects summary tables returned by summarizeMethylationGLMM\_T1T2Models().

**Usage**

```
plotMethylationGLMM_T1T2Diagnostics(
  modelSummaries,
  preparedData,
  fdrThreshold = 0.05,
  padjmethod = "fdr",
  outputDir = NULL,
  plotWidth = 2000L,
  plotHeight = 1000L,
  plotDPI = 150L,
```

```

display = FALSE,
verbose = FALSE,
logs = FALSE,
log_dir = NULL,
log_file = "log_methylationGLMM_T1T2.txt"
)

```

### Arguments

modelSummaries	Object returned by summarizeMethylationGLMM_T1T2Models().
preparedData	Object returned by prepareMethylationGLMM_T1T2Data().
fdrThreshold	Numeric. False-discovery-rate threshold used to highlight CpGs in the diagnostic plots.
padjmethod	Character. P-value adjustment method passed to stats::p.adjust().
outputDir	Character or NULL. Directory used for TIFF files. When NULL, plots are returned in memory only.
plotWidth	Integer. TIFF width in pixels when plots are written to disk.
plotHeight	Integer. TIFF height in pixels when plots are written to disk.
plotDPI	Integer. TIFF resolution in DPI when plots are written to disk.
display	Logical. If TRUE, draw plots on the active graphics device.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

### Value

A list with class "dnaEPICO\_methylationGLMM\_T1T2\_diagnostic\_plots" containing the generated ggplot2 objects, genomic inflation factors, and any saved TIFF file paths.

### Examples

```

ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
diagnostic_plots <- plotMethylationGLMM_T1T2Diagnostics(
  modelSummaries = ex$modelSummaries,
  preparedData = ex$preparedData,
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
names(diagnostic_plots$plots)

```

---

plotMetricsMinfiEwasWater

*Plot multidimensional scaling or density summaries from final metrics*


---

## Description

Plot multidimensional scaling or density summaries from final metrics

## Usage

```
plotMetricsMinfiEwasWater(
  metricsData,
  targets,
  plot = c("mds", "density"),
  plotGroupVar = "Sex",
  sexColumn = "Sex",
  display = FALSE,
  file = NULL,
  width = 2000L,
  height = 1000L,
  res = 150L,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_plotMetricsMinfiEwasWater.txt"
)
```

## Arguments

metricsData	Object returned by extractMetricsMinfiEwasWater().
targets	Filtered phenotype data aligned with metricsData.
plot	Character. Plot type: "mds" or "density".
plotGroupVar	Character. Phenotype column used for the main grouping.
sexColumn	Character. Phenotype column used for the sex grouping in the MDS plot.
display	Logical. If TRUE, draw the plot on the active graphics device.
file	Character or NULL. TIFF file written when supplied.
width	Integer. TIFF width in pixels when file is supplied.
height	Integer. TIFF height in pixels when file is supplied.
res	Integer. TIFF resolution in DPI when file is supplied.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

Invisibly returns the saved TIFF path when file is supplied, otherwise NULL.

**Examples**

```
ex <- dnaEPIC0:::exampleMinfiMetricsStateDnaEpico()
plotMetricsMinfiEwasWater(
  metricsData = ex$metricsData,
  targets = ex$targets,
  plot = "density",
  plotGroupVar = "Sex",
  sexColumn = "Sex",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
```

---

plotNormalizationMinfiEwasWater

*Plot raw and normalized methylation distributions*

---

**Description**

Draw the density comparison plot used to inspect raw versus normalized data.

**Usage**

```
plotNormalizationMinfiEwasWater(
  RGSet,
  normData,
  targets,
  sexColumn = "Sex",
  display = FALSE,
  file = NULL,
  width = 2000L,
  height = 1000L,
  res = 150L,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_plotNormalizationMinfiEwasWater.txt"
)
```

**Arguments**

RGSet	An RGChannelSet aligned with targets.
normData	Object returned by normalizeMinfiEwasWater().
targets	Filtered phenotype data aligned with RGSet.
sexColumn	Character. Name of the phenotype column used to colour the density curves.
display	Logical. If TRUE, draw the plot on the active graphics device.
file	Character or NULL. TIFF file written when supplied.
width	Integer. TIFF width in pixels when file is supplied.
height	Integer. TIFF height in pixels when file is supplied.
res	Integer. TIFF resolution in DPI when file is supplied.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

Invisibly returns the saved TIFF path when file is supplied, otherwise NULL.

**Examples**

```
ex <- dnaEPICO::exampleMinfiMetricsStateDnaEpico()
plotNormalizationMinfiEwasWater(
  RGSet = ex$beta,
  normData = ex$normData,
  targets = ex$targets,
  sexColumn = "Sex",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
```

---

plotRawDensityMinfiEwasWater

*Plot raw beta-value density from a raw preprocessing object*

---

**Description**

Draw the pre-normalization beta density plot from a raw minfi object and a grouping variable in the phenotype table.

**Usage**

```
plotRawDensityMinfiEwasWater(
  rawData,
  targets,
  plotGroupVar = "Sex",
  display = FALSE,
  file = NULL,
  width = 2000L,
  height = 1000L,
  res = 150L,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_plotRawDensityMinfiEwasWater.txt"
)
```

**Arguments**

rawData	Object returned by buildRawMinfiEwasWater().
targets	Filtered phenotype data aligned with rawData.
plotGroupVar	Character. Phenotype column used to group samples in the density plot.
display	Logical. If TRUE, draw the plot on the active graphics device.
file	Character or NULL. TIFF file written when supplied.
width	Integer. TIFF width in pixels when file is supplied.
height	Integer. TIFF height in pixels when file is supplied.
res	Integer. TIFF resolution in DPI when file is supplied.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

Invisibly returns the saved TIFF path when file is supplied, otherwise NULL.

**Examples**

```
ex <- dnaEPICO::exampleMinfiMetricsStateDnaEpico()
plotRawDensityMinfiEwasWater(
  rawData = ex$rawData,
  targets = ex$targets,
  plotGroupVar = "Sex",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
```

---

plotSexMinfiEwasWater *Plot predicted or clinical sex from predictSexMinfiEwasWater()*

---

### Description

Plot predicted or clinical sex from predictSexMinfiEwasWater()

### Usage

```
plotSexMinfiEwasWater(
  sexData,
  type = c("predicted", "clinical"),
  display = FALSE,
  file = NULL,
  width = 2000L,
  height = 1000L,
  res = 70L,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_plotSexMinfiEwasWater.txt"
)
```

### Arguments

sexData	Object returned by predictSexMinfiEwasWater().
type	Character. Plot type: "predicted" for methylation-predicted sex or "clinical" for reported sex.
display	Logical. If TRUE, draw the plot on the active graphics device.
file	Character or NULL. TIFF file written when supplied.
width	Integer. TIFF width in pixels when file is supplied.
height	Integer. TIFF height in pixels when file is supplied.
res	Integer. TIFF resolution in DPI when file is supplied.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

### Value

Invisibly returns the saved TIFF path when file is supplied, otherwise NULL.

**Examples**

```
ex <- dnaEPICO::exampleSexPlotStateDnaEpico()
plotSexMinfiEwasWater(
  sexData = ex,
  type = "predicted",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
```

---

plotSvaEnmix

*Plot surrogate variables for svaEnmix*


---

**Description**

Draw one of the standard surrogate-variable plots used by `svaEnmix()`.

**Usage**

```
plotSvaEnmix(
  analysisData,
  plot = c("sentrix_id", "sentrix_position", "matrix"),
  display = FALSE,
  file = NULL,
  width = 2000L,
  height = 1000L,
  res = 150L,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_plotSvaEnmix.txt"
)
```

**Arguments**

<code>analysisData</code>	Object returned by <code>analyzeSvaEnmix()</code> .
<code>plot</code>	Character. Plot type: "sentrix_id", "sentrix_position", or "matrix".
<code>display</code>	Logical. If TRUE, draw the plot on the active graphics device.
<code>file</code>	Character or NULL. TIFF file path used for saved output.
<code>width</code>	Integer. Plot width in pixels when file is supplied.
<code>height</code>	Integer. Plot height in pixels when file is supplied.
<code>res</code>	Integer. TIFF resolution in DPI when file is supplied.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

**Value**

Invisibly returns file when a TIFF is written, otherwise NULL.

**Examples**

```
ex <- dnaEPICO:::exampleSvaAnalysisStateDnaEpico()
plotSvaEnmix(
  analysisData = ex$analysisData,
  plot = "sentry_id",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE
)
```

---

predictSexMinfiEwasWater

*Predict biological sex from a filtered raw-data object*

---

**Description**

Predict sample sex from a genome-mapped methylation object, align the predictions with phenotype data, and return a structured object that can be plotted or merged into downstream phenotype tables.

**Usage**

```
predictSexMinfiEwasWater(
  rawData,
  targets,
  SampleID = "Sample_Name",
  sexColumn = "Sex",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_predictSexMinfiEwasWater.txt"
)
```

**Arguments**

rawData	Object returned by buildRawMinfiEwasWater().
targets	Filtered phenotype data frame aligned with rawData.
SampleID	Character. Name of the sample identifier column in targets.
sexColumn	Character. Name of the phenotype column containing reported sex.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_minfiEwasWater\_sex" containing the sex prediction result, aligned phenotype data, plotting data, and mismatch table.

**Examples**

```
ex <- dnaEPICO::exampleMinfiWorkflowStateDnaEpico()
sex_data <- predictSexMinfiEwasWater(
  rawData = ex$rawFiltered,
  targets = ex$sampleData$targets,
  SampleID = "Sample_Name",
  sexColumn = "Sex",
  verbose = FALSE,
  logs = FALSE
)
names(sex_data)
```

---

```
prepareDnamReportInputs
```

*Prepare inputs for a DNA methylation report*

---

**Description**

Prepare inputs for a DNA methylation report

**Usage**

```
prepareDnamReportInputs(
  outputDir = "reports",
  qcDir = file.path("figures", "preprocessingMinfiEwasWater", "enmix"),
  preprocessingDir = file.path("figures", "preprocessingMinfiEwasWater", "qc"),
  postprocessingDir = file.path("figures", "preprocessingMinfiEwasWater", "metrics"),
  svaDir = file.path("figures", "svaEnmix"),
  glmDir = file.path("figures", "methylationGLM_T1"),
  glmmDir = file.path("figures", "methylationGLMM_T1T2"),
  figDir = file.path(outputDir, "assets", "figures"),
  verbose = FALSE,
  logs = FALSE,
  logDir = outputDir
)
```

**Arguments**

outputDir	Character. Directory where the report project is written.
qcDir	Character. Directory containing ENmix quality-control figures.
preprocessingDir	Character. Directory containing preprocessing quality-control figures.

postprocessingDir	Character. Directory containing postprocessing metric figures.
svaDir	Character. Directory containing SVA or batch-effect figures.
glmDir	Character. Directory containing GLM figures.
glmmDir	Character. Directory containing GLMM figures.
figDir	Character. Directory used for generated report figure assets.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write progress messages to file.path(logDir, "log_dnamReport.txt").
logDir	Character. Directory for optional log files.

**Value**

A list with class "dnaEPICO\_dnamReport\_prepared".

**Examples**

```
report_root <- file.path(tempdir(), "dnaepico-report-inputs")
prepared <- prepareDnamReportInputs(
  outputDir = file.path(report_root, "reports"),
  qcDir = file.path(
    report_root,
    "figures",
    "preprocessingMinfiEwasWater",
    "enmix"
  ),
  preprocessingDir = file.path(
    report_root,
    "figures",
    "preprocessingMinfiEwasWater",
    "qc"
  ),
  postprocessingDir = file.path(
    report_root,
    "figures",
    "preprocessingMinfiEwasWater",
    "metrics"
  ),
  svaDir = file.path(report_root, "figures", "svaEnmix")
)
inherits(prepared, "dnaEPICO_dnamReport_prepared")
```

**Description**

Load the merged phenotype-plus-beta input object, validate the requested modeling variables, convert selected variables to factors, and return a single in-memory object for downstream helpers.

**Usage**

```
prepareMethylationGLM_T1Data(
  inputPheno,
  phenotypes,
  covariates,
  factorVars,
  cpGPrefix = "cg",
  cpGLimit = NA,
  interactionTerm = NULL,
  prsMap = NULL,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLM_T1.txt"
)
```

**Arguments**

inputPheno	Character. Path to the merged phenotype-plus-beta object created by preprocessingPheno().
phenotypes	Character vector or comma-separated string of phenotype variables to model.
covariates	Character vector or comma-separated string of covariate variables to adjust for.
factorVars	Character vector or comma-separated string of variables that should be converted to factors before modeling.
cpGPrefix	Character. Prefix used to identify methylation columns.
cpGLimit	Integer or NA. Maximum number of CpGs to retain. NA keeps all matching CpGs.
interactionTerm	Character or NULL. Optional interaction term.
prsMap	Character vector or comma-separated string of phenotype-to-PRS mappings in the form "Phenotype:PRS".
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPIC0\_methylationGLM\_T1\_data" containing the prepared analysis data, parsed variable selections, CpG columns, and exploratory summaries.

**Examples**

```
ex <- dnaEPIC0::exampleMethylationGLMStateDnaEpico()
prepared_data <- prepareMethylationGLM_T1Data(
  inputPheno = ex$inputPath,
  phenotypes = "status",
  covariates = "sex,age",
  factorVars = "status,sex",
  cpGLimit = 2,
  verbose = FALSE,
  logs = FALSE
)
names(prepared_data)
```

---

```
prepareMethylationGLMM_T1T2Data
```

*Prepare longitudinal phenotype-plus-beta data for mixed-effects analyses*

---

**Description**

Load the merged longitudinal phenotype-plus-beta object, ensure that a subject identifier column is available, validate the requested modeling variables, convert selected variables to factors, and return a single in-memory object for downstream mixed-effects modeling helpers.

**Usage**

```
prepareMethylationGLMM_T1T2Data(
  inputPheno,
  personVar = "person",
  timeVar = "Timepoint",
  phenotypes,
  covariates,
  factorVars,
  prsMap = NULL,
  cpGLimit = NA,
  interactionTerm = NULL,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLMM_T1T2.txt"
)
```

**Arguments**

**inputPheno**      Character. Path to the merged longitudinal phenotype-plus- beta object created by preprocessingPheno().

personVar	Character. Name of the subject identifier column.
timeVar	Character. Name of the time variable.
phenotypes	Character vector or comma-separated string of phenotype variables to model.
covariates	Character vector or comma-separated string of covariate variables to adjust for.
factorVars	Character vector or comma-separated string of variables that should be converted to factors before modeling.
prsMap	Character vector or comma-separated string of phenotype-to-PRS mappings in the form "Phenotype:PRS".
cpgPrefix	Character. Prefix used to identify methylation columns.
cpgLimit	Integer or NA. Maximum number of CpGs to retain. NA keeps all matching CpGs.
interactionTerm	Character or NULL. Optional interaction term.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

### Value

A list with class "dnaEPICO\_methylationGLMM\_T1T2\_data" containing the prepared analysis data, parsed variable selections, CpG columns, timepoint summaries, and subject-ID diagnostics.

### Examples

```
ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
prepared_data <- prepareMethylationGLMM_T1T2Data(
  inputPheno = ex$inputPath,
  personVar = "person",
  timeVar = "Timepoint",
  phenotypes = "score",
  covariates = "sex",
  factorVars = "sex,Timepoint",
  cpgLimit = 2,
  verbose = FALSE,
  logs = FALSE
)
names(prepared_data)
```

---

```
preprocessingMinfiEwasWater
```

*Convenience preprocessing pipeline for Illumina methylation arrays*

---

## Description

Run the dnaEPIC0 preprocessing workflow as a convenience wrapper around the smaller minfi/ENmix/wateRmelon helper functions in this package. The wrapper now returns a structured result object containing the in-memory outputs from each stage. Legacy files are written only when `saveOutputs = TRUE`.

## Usage

```
preprocessingMinfiEwasWater(
  phenoFile = "data/preprocessingMinfiEwasWater/pheno.csv",
  idatFolder = "data/preprocessingMinfiEwasWater/idats",
  outputLogs = "logs",
  nSamples = NA,
  SampleID = "Sample_Name",
  arrayType = "IlluminaHumanMethylationEPICv2",
  annotationVersion = "20a1.hg38",
  scriptLabel = "preprocessingMinfiEwasWater",
  baseDataFolder = "rData",
  figureBaseDir = "figures",
  sepType = "",
  tiffWidth = 2000,
  tiffHeight = 1000,
  tiffRes = 150,
  qcCutoff = 10.5,
  detPtype = "m+u",
  detPThreshold = 0.05,
  normMethods = "adjustedfunnorm",
  sexColumn = "Sex",
  pvalThreshold = 0.01,
  chrToRemove = "chrX,chrY",
  snpsToRemove = "SBE,CpG",
  mafThreshold = 0.1,
  crossReactivePath = "data/preprocessingMinfiEwasWater/12864_2024_10027_MOESM8_ESM.csv",
  plotGroupVar = "Sex",
  lcRef = "salivaEPIC",
  phenoOrder = "Sample_Name;Timepoint;Sex;PredSex;Basename;Sentry_ID;Sentry_Position",
  lcPhenoDir = "data/preprocessingMinfiEwasWater",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE,
  saveOutputs = FALSE
)
```

**Arguments**

phenoFile	Character. Path to the phenotype CSV file.
idatFolder	Character. Directory containing the IDAT files.
outputLogs	Character. Directory used for log files when logs = TRUE.
nSamples	Integer or NA. Number of rows to keep from the phenotype table. Use NA to keep all samples.
SampleID	Character. Name of the phenotype column containing sample identifiers.
arrayType	Character. Illumina array identifier passed to Biobase::annotation(), for example "IlluminaHumanMethylationEPICv2".
annotationVersion	Character. Annotation build passed to Biobase::annotation(), for example "20a1.hg38" or "ilmn12.hg19".
scriptLabel	Character. Label used to name output folders when saveOutputs = TRUE.
baseDataFolder	Character. Base directory used for saved .RData outputs when saveOutputs = TRUE.
figureBaseDir	Character. Base directory used for saved figure outputs when saveOutputs = TRUE.
sepType	Character. Field separator used in phenoFile. Use "," for a comma-separated file, "\\t" for a tab-delimited file, or another separator accepted by utils::read.csv().
tiffWidth	Integer. Width of saved TIFF plots in pixels.
tiffHeight	Integer. Height of saved TIFF plots in pixels.
tiffRes	Integer. Resolution in DPI for saved TIFF plots.
qcCutoff	Numeric. QC cutoff passed to minfi::plotQC().
detPtype	Character. Detection P-value mode passed to minfi::detectionP(). Common values in minfi workflows are "m+u" and "negative". The default here is "m+u".
detPThreshold	Numeric. Samples with mean detection P value above this threshold are removed.
normMethods	Character vector or semicolon-separated string of normalization methods. Supported values are "adjustedfunnorm", "funnorm", "illumina", "quantile", and "swan".
sexColumn	Character. Name of the phenotype column containing reported sex.
pvalThreshold	Numeric. Probe-level detection P-value threshold used in the probe filter.
chrToRemove	Character vector or comma-separated string of chromosome names to remove, for example "chrX, chrY".
snpsToRemove	Character vector or comma-separated string of SNP probe types to remove, for example "SBE, CpG".
mafThreshold	Numeric. Minor allele frequency threshold passed to minfi::dropLociWithSnps().
crossReactivePath	Character. Path to a CSV file containing a ProbeID column of cross-reactive probes to remove.

plotGroupVar	Character. Phenotype column used for density and MDS grouping plots.
lcRef	Character. Reference panel used for cell composition estimation. "saliva" and "salivaEPIC" use estimateLC(). Other values are passed to ENmix::estimateCellProp().
phenoOrder	Character vector or semicolon-separated string describing which phenotype columns should appear first in the merged phenoLC table.
lcPhenoDir	Character. Directory used for the saved phenoLC.csv file when saveOutputs = TRUE.
display	Logical. If TRUE, draw plots on the active graphics device.
verbose	Logical. If TRUE, emit progress messages with message(). The default is FALSE.
logs	Logical. If TRUE, write log messages to outputLogs. The default is FALSE.
saveOutputs	Logical. If TRUE, write the legacy .RData, figure, and phenoLC.csv outputs to disk. The default is FALSE, so the function can be used in the more traditional in-memory Bioconductor style.

### Value

A list with class "dnaEPICO\_preprocessingMinfiEwasWater".

**targets** Filtered phenotype table aligned to the retained samples.

**RGSet** Filtered RGChannelSet used in downstream preprocessing and available for direct interactive inspection.

**rawData** Object returned by [buildRawMinfiEwasWater\(\)](#) containing the raw MSet, RatioSet, and genome-mapped object derived from RGSet.

**assessment** Object returned by [assessSamplesMinfiEwasWater\(\)](#) containing detection P values, QC summaries, and failed-sample tracking.

**sexData** Object returned by [predictSexMinfiEwasWater\(\)](#) containing predicted sex labels, mismatch summaries, and plotting data.

**normData** Object returned by [normalizeMinfiEwasWater\(\)](#) containing the requested normalized objects and metadata on the methods that were run.

**filterData** Object returned by [filterProbesMinfiEwasWater\(\)](#) containing the probe-filtered methylation objects at each filtering stage.

**metricsData** Object returned by [extractMetricsMinfiEwasWater\(\)](#) containing the beta-value, M-value, and copy-number matrices used by later workflow steps.

**lcData** Object returned by [estimateLCMinfiEwasWater\(\)](#) containing the estimated cell-type proportions and the phenotype table augmented with those proportions.

**logFile** Resolved path to the optional log file, or NULL when logging was disabled.

See [dnaEPICO\\_preprocessingMinfiEwasWater](#) for a class-level overview.

### See Also

[dnaEPICO\\_preprocessingMinfiEwasWater](#)

**Examples**

```

if (requireNamespace("minfiData", quietly = TRUE) &&
    requireNamespace("IlluminaHumanMethylation450kmanifest", quietly = TRUE) &&
    requireNamespace("IlluminaHumanMethylation450kanno.ilmn12.hg19", quietly = TRUE)) {
ex <- dnaEPICO::exampleMinfiIdatInputsDnaEpico(n = 4)
result <- preprocessingMinfiEwasWater(
  phenoFile = ex$phenoFile,
  idatFolder = ex$idatFolder,
  outputLogs = file.path(ex$tempDir, "logs"),
  nSamples = 4,
  SampleID = "Sample_Name",
  arrayType = ex$arrayType,
  annotationVersion = ex$annotationVersion,
  scriptLabel = "preprocessingMinfiEwasWater",
  baseDataFolder = file.path(ex$tempDir, "rData"),
  figureBaseDir = file.path(ex$tempDir, "figures"),
  detPThreshold = 1,
  normMethods = "quantile",
  sexColumn = "Sex",
  pvalThreshold = 1,
  chrToRemove = "",
  snpsToRemove = "SBE",
  mafThreshold = 1,
  crossReactivePath = ex$crossReactivePath,
  plotGroupVar = "Sex",
  lcRef = "saliva",
  phenoOrder = "Sample_Name;Sex;Basename;Sentrix_ID;Sentrix_Position",
  lcPhenoDir = ex$tempDir,
  saveOutputs = FALSE,
  verbose = FALSE,
  logs = FALSE
)
inherits(result, "dnaEPICO_preprocessingMinfiEwasWater")
}

```

---

preprocessingPheno	<i>Prepare phenotype and methylation matrices for downstream modeling</i>
--------------------	---

---

**Description**

Read the phenotype table and the preprocessed beta, M-value, and copy-number matrices; align them by sample identifier; split them by timepoint; prepare combined longitudinal objects; and build Clock Foundation export tables. The function returns a structured in-memory result, while legacy files are written only when `saveOutputs = TRUE`.

**Usage**

```

preprocessingPheno(
  phenoFile = "data/preprocessingMinfiEwasWater/phenoLC.csv",
  sepType = "",
  betaPath =
    "rData/preprocessingMinfiEwasWater/metrics/beta_NomFilt_MSetF_Flt_Rxy_Ds_Rc.RData",
  mPath = "rData/preprocessingMinfiEwasWater/metrics/m_NomFilt_MSetF_Flt_Rxy_Ds_Rc.RData",
  cnPath =
    "rData/preprocessingMinfiEwasWater/metrics/cn_NomFilt_MSetF_Flt_Rxy_Ds_Rc.RData",
  SampleID = "Sample_Name",
  timeVar = "Timepoint",
  timepoints = "1,2",
  combineTimepoints = "1,2",
  outputPheno = "data/preprocessingPheno",
  outputRData = "rData/preprocessingPheno/metrics",
  outputRDataMerge = "rData/preprocessingPheno/mergeData",
  sexColumn = "Sex",
  outputLogs = "logs",
  outputDir = "data/preprocessingPheno",
  verbose = FALSE,
  logs = FALSE,
  saveOutputs = FALSE
)

```

**Arguments**

phenoFile	Character. Path to the phenotype CSV file.
sepType	Character. Field separator used in phenoFile. Use "" for a comma-separated file, "\\t" for a tab-delimited file, or another separator accepted by <code>utils::read.csv()</code> .
betaPath	Character. Path to the saved beta-value object. Both .RData and .rds files are supported.
mPath	Character. Path to the saved M-value object. Both .RData and .rds files are supported.
cnPath	Character. Path to the saved copy-number object. Both .RData and .rds files are supported.
SampleID	Character. Name of the phenotype column containing sample identifiers used to align phenotype and methylation data.
timeVar	Character. Name of the phenotype column containing timepoint labels.
timepoints	Character vector or comma-separated string of timepoints to retain and split into separate in-memory subsets.
combineTimepoints	Character vector or comma-separated string of timepoints to combine into the longitudinal phenotype-plus-beta object.
outputPheno	Character. Directory used for saved phenotype CSV files when <code>saveOutputs = TRUE</code> .

outputRData	Character. Directory used for saved metric .RData files when saveOutputs = TRUE.
outputRDataMerge	Character. Directory used for saved merged phenotype-plus-beta .RData files when saveOutputs = TRUE.
sexColumn	Character. Name of the phenotype sex column used when building Clock Foundation exports.
outputLogs	Character. Directory used for log files when logs = TRUE.
outputDir	Character. Directory used for Clock Foundation export files when saveOutputs = TRUE.
verbose	Logical. If TRUE, emit progress messages with message(). The default is FALSE.
logs	Logical. If TRUE, write the same progress messages to outputLogs. The default is FALSE.
saveOutputs	Logical. If TRUE, write the legacy CSV, ZIP, and .RData outputs to disk. The default is FALSE, so the function can be used in the more traditional in-memory Bioconductor style.

**Value**

A list with class "dnaEPICO\_preprocessingPheno".

**pheno** Phenotype table read from phenoFile.

**metricsData** Object returned by `loadMetricsPreprocessingPheno()` containing the beta-value, M-value, and copy-number matrices loaded from betaPath, mPath, and cnPath.

**timepointData** Object returned by `splitTimepointsPreprocessingPheno()` containing per-timepoint phenotype tables and methylation matrices.

**combinedData** Object returned by `combineTimepointsPreprocessingPheno()` containing the merged longitudinal phenotype-plus-beta object and the timepoint combination metadata.

**clockFoundation** Object returned by `buildClockFoundationInputsPreprocessingPheno()` containing the beta table and phenotype table prepared for Clock Foundation export.

**savedFiles** Object returned by `writePreprocessingPhenoOutputs()` when saveOutputs = TRUE, otherwise NULL.

**logFile** Resolved path to the optional log file, or NULL when logging was disabled.

See [dnaEPICO\\_preprocessingPheno](#) for a class-level overview.

**See Also**

[dnaEPICO\\_preprocessingPheno](#)

**Examples**

```
tmp <- tempdir()
pheno <- data.frame(
  Sample_Name = c("S1", "S2", "S3"),
  Timepoint = c("1", "1", "2"),
  Sex = c(0, 1, 0),
```

```

    stringsAsFactors = FALSE
  )
  beta <- matrix(
    c(0.10, 0.20, 0.30, 0.40, 0.50, 0.60),
    nrow = 2,
    dimnames = list(c("cg1", "cg2"), pheno$Sample_Name)
  )
  m <- beta * 10
  cn <- beta * 100
  pheno_file <- file.path(tmp, "pheno.csv")
  beta_path <- file.path(tmp, "beta.RData")
  m_path <- file.path(tmp, "m.RData")
  cn_path <- file.path(tmp, "cn.RData")
  utils::write.csv(pheno, pheno_file, row.names = FALSE)
  save(beta, file = beta_path)
  save(m, file = m_path)
  save(cn, file = cn_path)
  result <- preprocessingPheno(
    phenoFile = pheno_file,
    betaPath = beta_path,
    mPath = m_path,
    cnPath = cn_path,
    SampleID = "Sample_Name",
    timeVar = "Timepoint",
    timepoints = "1,2",
    combineTimepoints = "1,2",
    outputPheno = file.path(tmp, "data", "preprocessingPheno"),
    outputRData = file.path(tmp, "rData", "preprocessingPheno", "metrics"),
    outputRDataMerge = file.path(tmp, "rData", "preprocessingPheno", "mergeData"),
    sexColumn = "Sex",
    outputLogs = file.path(tmp, "logs"),
    outputDir = file.path(tmp, "clockFoundation"),
    saveOutputs = FALSE
  )
  stopifnot(inherits(result, "dnaEPICO_preprocessingPheno"))

```

---

```
print.dnaEPICO_dnamReport
```

*Print a DNA methylation report result*

---

## Description

Print a DNA methylation report result

## Usage

```
## S3 method for class 'dnaEPICO_dnamReport'
print(x, ...)
```

**Arguments**

x                    Object returned by `dnamReport()`.  
 ...                  Additional arguments ignored.

**Value**

Invisibly returns x.

---

readPhenotypeTargets    *Read phenotype targets for shared dnaEPICO workflows*

---

**Description**

Read the phenotype table used by shared dnaEPICO workflows, validate the sample identifier column, optionally subset the first nSamples, and return the targets as a base data.frame.

**Usage**

```
readPhenotypeTargets(
  phenoFile,
  sepType = "",
  nSamples = NA,
  SampleID = "Sample_Name",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_readPhenotypeTargets.txt"
)
```

**Arguments**

phenoFile	Character. Path to the phenotype table on disk.
sepType	Character. Field separator used in phenoFile. Use "" (default) for a standard comma-separated file, "\\t" for a tab-delimited file, or another single-character separator accepted by <code>utils::read.csv()</code> .
nSamples	Integer or NA. Number of rows to keep from the start of the phenotype table. The default NA reads and returns all rows.
SampleID	Character. Name of the column containing sample identifiers that will later be used to name methylation-array samples.
verbose	Logical. If TRUE, emit progress and preview messages with <code>message()</code> . The default is FALSE, so the function is quiet unless the user explicitly requests messages.
logs	Logical. If TRUE, write the same progress messages to a log file. The default is FALSE.
log_dir	Character or NULL. Directory where the log file should be written when <code>logs = TRUE</code> . If NULL, the current working directory is used.
log_file	Character. File name used when <code>logs = TRUE</code> . The default is "log_readPhenotypeTargets.txt".

**Value**

A data.frame containing the phenotype targets.

**Examples**

```
tmp <- tempdir()
pheno <- data.frame(
  Sample_Name = c("S1", "S2"),
  Sex = c("F", "M"),
  stringsAsFactors = FALSE
)
pheno_file <- file.path(tmp, "pheno.csv")
utils::write.csv(pheno, pheno_file, row.names = FALSE)
targets <- readPhenotypeTargets(
  phenoFile = pheno_file,
  SampleID = "Sample_Name"
)
stopifnot(is.data.frame(targets))
stopifnot(nrow(targets) == 2L)
```

---

readRGSetMinfiEwasWater

*Read IDAT files into an annotated RGChannelSet*

---

**Description**

Read methylation-array IDAT files with `minfi::read.metharray.exp()`, set sample names from the phenotype table, apply the requested annotation, and return the resulting `RGChannelSet`.

**Usage**

```
readRGSetMinfiEwasWater(
  idatFolder,
  targets,
  SampleID = "Sample_Name",
  arrayType = "IlluminaHumanMethylationEPICv2",
  annotationVersion = "20a1.hg38",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_readRGSetMinfiEwasWater.txt"
)
```

**Arguments**

idatFolder	Character. Directory containing the IDAT files.
targets	Data frame returned by readPhenotypeTargets() or an equivalent phenotype table.
SampleID	Character. Name of the phenotype column containing sample identifiers used to label the RGChannelSet.
arrayType	Character. Array name passed to Biobase::annotation(RGSet), for example "IlluminaHumanMethylationEPICv2".
annotationVersion	Character. Annotation build passed to Biobase::annotation(RGSet), for example "20a1.hg38" for EPIC v2 hg38 annotations or "ilmn12.hg19" for 450K hg19 annotations.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

An annotated RGChannelSet.

**Examples**

```
if (requireNamespace("minfiData", quietly = TRUE) &&
    requireNamespace("IlluminaHumanMethylation450kmanifest", quietly = TRUE) &&
    requireNamespace("IlluminaHumanMethylation450kanno.ilmn12.hg19", quietly = TRUE)) {
  ex <- dnaEPIC0::exampleMinfiIdatInputsDnaEpico(n = 4)
  rgset <- readRGSetMinfiEwasWater(
    idatFolder = ex$idatFolder,
    targets = ex$targets,
    SampleID = "Sample_Name",
    arrayType = ex$arrayType,
    annotationVersion = ex$annotationVersion
  )
  class(rgset)
}
```

---

renderDnamReport

*Render a prepared DNA methylation report*


---

**Description**

Render a prepared DNA methylation report

**Usage**

```
renderDnamReport(  
  preparedReport,  
  verbose = FALSE,  
  logs = FALSE,  
  logDir = NULL,  
  clean = TRUE  
)
```

**Arguments**

preparedReport	Object returned by prepareDnamReportInputs().
verbose	Logical. If TRUE, emit progress messages.
logs	Logical. If TRUE, write progress messages to a log file.
logDir	Character or NULL. Directory for optional log files.
clean	Logical. Retained for backwards compatibility.

**Value**

A list with class "dnaEPICO\_dnamReport\_render".

**Examples**

```
report_root <- file.path(tempdir(), "dnaepico-render-example")  
prepared <- prepareDnamReportInputs(  
  outputDir = file.path(report_root, "reports")  
)  
rendered <- renderDnamReport(prepared)  
rendered$status
```

---

splitTimepointsPreprocessingPheno

*Split phenotype and methylation data by timepoint*

---

**Description**

Align phenotype rows and metric matrices for each requested timepoint, and precompute the per-timepoint phenotype-plus-beta objects used by downstream modeling functions.

**Usage**

```
splitTimepointsPreprocessingPheno(
  pheno,
  metricsData,
  SampleID = "Sample_Name",
  timeVar = "Timepoint",
  timepoints = "1,2",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_splitTimepointsPreprocessingPheno.txt"
)
```

**Arguments**

pheno	Data frame containing phenotype information.
metricsData	Object returned by loadMetricsPreprocessingPheno().
SampleID	Character. Name of the sample identifier column in pheno.
timeVar	Character. Name of the timepoint column in pheno.
timepoints	Character vector or comma-separated string of timepoints to retain.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_preprocessingPheno\_timepoints" containing the parsed timepoints and aligned per-timepoint subsets.

**Examples**

```
ex <- dnaEPICO::examplePreprocessingPhenoStateDnaEpico()
timepoint_data <- splitTimepointsPreprocessingPheno(
  pheno = ex$pheno,
  metricsData = ex$metricsData,
  SampleID = "Sample_Name",
  timeVar = "Timepoint",
  timepoints = "1,2",
  verbose = FALSE,
  logs = FALSE
)
timepoint_data$timepoints
```

---

```
summarizeMethylationGLM_T1Models
```

*Summarize CpG-wise Gaussian GLM fits for one-timepoint analyses*

---

## Description

Extract phenotype-specific CpG coefficient tables from the fitted model object returned by `fitMethylationGLM_T1Models()`

## Usage

```
summarizeMethylationGLM_T1Models(
  modelResults,
  preparedData,
  summaryResidualSD = TRUE,
  summaryPval = NA,
  nCores = 1L,
  libPath = NULL,
  glmLibs = "glm2",
  chunkSize = NULL,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLM_T1.txt"
)
```

## Arguments

<code>modelResults</code>	Object returned by <code>fitMethylationGLM_T1Models()</code> .
<code>preparedData</code>	Object returned by <code>prepareMethylationGLM_T1Data()</code> .
<code>summaryResidualSD</code>	Logical. If TRUE, add residual standard deviations to each CpG summary row.
<code>summaryPval</code>	Numeric or NA. Optional p-value filter applied to the returned summary tables. NA keeps all rows.
<code>nCores</code>	Integer. Number of worker processes to use while extracting summary rows.
<code>libPath</code>	Character vector or NULL. Optional library paths forwarded to worker processes.
<code>glmLibs</code>	Character vector or comma-separated string of package names to check on worker processes. The default is "glm2".
<code>chunkSize</code>	Integer or NULL. Number of CpGs to process per parallel chunk. NULL chooses a value automatically.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

**Value**

A list with class "dnaEPIC0\_methylationGLM\_T1\_summaries" containing one CpG-level summary data frame per phenotype.

**Examples**

```
ex <- dnaEPIC0::exampleMethylationGLMStateDnaEpico()
summary_results <- summarizeMethylationGLM_T1Models(
  modelResults = ex$modelResults,
  preparedData = ex$preparedData,
  summaryResidualSD = TRUE,
  summaryPval = NA,
  nCores = 1,
  verbose = FALSE,
  logs = FALSE
)
names(summary_results$summaries)
```

---

```
summarizeMethylationGLMM_T1T2Models
```

*Summarize CpG-wise mixed-effects model fits for longitudinal analyses*

---

**Description**

Extract phenotype-specific fixed-effect tables from the fitted mixed-effects model object returned by fitMethylationGLMM\_T1T2Models().

**Usage**

```
summarizeMethylationGLMM_T1T2Models(
  modelResults,
  preparedData,
  summaryPval = NA,
  nCores = 1L,
  chunkSize = NULL,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_methylationGLMM_T1T2.txt"
)
```

**Arguments**

modelResults    Object returned by fitMethylationGLMM\_T1T2Models().  
 preparedData    Object returned by prepareMethylationGLMM\_T1T2Data().

summaryPval	Numeric or NA. Optional p-value filter applied to the returned summary tables. NA keeps all rows.
nCores	Integer. Number of worker processes to use while extracting summary rows.
chunkSize	Integer or NULL. Number of CpGs processed per parallel chunk. NULL chooses a value automatically.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_methylationGLMM\_T1T2\_summaries" containing one CpG-level summary data frame per phenotype.

**Examples**

```
ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
summary_results <- summarizeMethylationGLMM_T1T2Models(
  modelResults = ex$modelResults,
  preparedData = ex$preparedData,
  summaryPval = NA,
  nCores = 1,
  verbose = FALSE,
  logs = FALSE
)
names(summary_results$summaries)
```

---

```
summarizeTimepointsMethylationGLMM_T1T2
```

*Summarize phenotype values by timepoint for longitudinal methylation analyses*

---

**Description**

Summarize the requested phenotype variables by timepoint. Numeric phenotypes are reported with mean, standard deviation, and non-missing counts; non-numeric phenotypes are reported with non-missing counts and the observed levels.

**Usage**

```
summarizeTimepointsMethylationGLMM_T1T2(
  data,
  timeVar = "Timepoint",
  phenotypes,
```

```

    verbose = FALSE,
    logs = FALSE,
    log_dir = NULL,
    log_file = "log_methylationGLMM_T1T2.txt"
  )

```

### Arguments

<code>data</code>	Data frame containing the longitudinal phenotype-plus-beta data.
<code>timeVar</code>	Character. Name of the time variable.
<code>phenotypes</code>	Character vector or comma-separated string of phenotype variables to summarize.
<code>verbose</code>	Logical. If TRUE, emit progress messages with <code>message()</code> .
<code>logs</code>	Logical. If TRUE, write the same messages to a log file.
<code>log_dir</code>	Character or NULL. Directory used for the log file when <code>logs = TRUE</code> .
<code>log_file</code>	Character. File name used when <code>logs = TRUE</code> .

### Value

A data frame with one row per timepoint and summary columns for each requested phenotype.

### Examples

```

ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
timepoint_summary <- summarizeTimepointsMethylationGLMM_T1T2(
  data = ex$preparedData$data,
  timeVar = "Timepoint",
  phenotypes = "score",
  verbose = FALSE,
  logs = FALSE
)
nrow(timepoint_summary)

```

### Description

Read the phenotype table and a saved `RGChannelSet`, estimate surrogate variables from ENmix control probes, analyze their association with Sentrix chip and position factors, and return a structured in-memory result. Legacy CSV, `.RData`, text-summary, and figure outputs are written only when `saveOutputs = TRUE`.

**Usage**

```

svaEnmix(
  phenoFile = "data/preprocessingMinfiEwasWater/phenoLC.csv",
  rgsetData = "rData/preprocessingMinfiEwasWater/objects/RGSet.RData",
  sepType = "",
  outputLogs = "logs",
  nSamples = NA,
  SampleID = "Sample_Name",
  arrayType = "IlluminaHumanMethylationEPICv2",
  annotationVersion = "20a1.hg38",
  SentrixIDColumn = "Sentrix_ID",
  SentrixPositionColumn = "Sentrix_Position",
  ctrlSvaPercVar = 0.9,
  ctrlSvaFlag = 1,
  scriptLabel = "svaEnmix",
  tiffWidth = 2000,
  tiffHeight = 1000,
  tiffRes = 150,
  figureBaseDir = "figures",
  dataBaseDir = "data",
  rBaseDir = "rData",
  display = FALSE,
  verbose = FALSE,
  logs = FALSE,
  saveOutputs = FALSE
)

```

**Arguments**

phenoFile	Character. Path to the phenotype file with cell-composition data.
rgsetData	Character. Path to a saved RGChannelSet object. Both .RData and .rds files are supported.
sepType	Character. Field separator used in phenoFile. Use "" for a comma-separated file, "\\t" for a tab-delimited file, or another separator accepted by <code>utils::read.csv()</code> .
outputLogs	Character. Directory used for log files when <code>logs = TRUE</code> .
nSamples	Integer or NA. Number of rows to keep from the phenotype table. Use NA to keep all samples.
SampleID	Character. Name of the phenotype column containing sample identifiers.
arrayType	Character. Illumina array identifier assigned to <code>Biobase::annotation(RGSet)</code> .
annotationVersion	Character. Annotation build assigned to <code>Biobase::annotation(RGSet)</code> .
SentrixIDColumn	Character. Name of the chip identifier column in the phenotype data.
SentrixPositionColumn	Character. Name of the chip position column in the phenotype data.
ctrlSvaPercVar	Numeric. Proportion of control-probe variance explained when running <code>ENmix::ctrlsva()</code> .

<code>ctrlSvaFlag</code>	Integer. Control-probe flag passed to <code>ENmix::ctrlsva()</code> .
<code>scriptLabel</code>	Character. Label used to name output folders when <code>saveOutputs = TRUE</code> .
<code>tiffWidth</code>	Integer. Width of saved TIFF plots in pixels.
<code>tiffHeight</code>	Integer. Height of saved TIFF plots in pixels.
<code>tiffRes</code>	Integer. Resolution in DPI for saved TIFF plots.
<code>figureBaseDir</code>	Character. Base directory used for saved figure outputs when <code>saveOutputs = TRUE</code> .
<code>dataBaseDir</code>	Character. Base directory used for saved CSV and text outputs when <code>saveOutputs = TRUE</code> .
<code>rBaseDir</code>	Character. Base directory used for saved <code>.RData</code> outputs when <code>saveOutputs = TRUE</code> .
<code>display</code>	Logical. If <code>TRUE</code> , draw plots on the active graphics device.
<code>verbose</code>	Logical. If <code>TRUE</code> , emit progress messages with <code>message()</code> . The default is <code>FALSE</code> .
<code>logs</code>	Logical. If <code>TRUE</code> , write the same progress messages to <code>outputLogs</code> . The default is <code>FALSE</code> .
<code>saveOutputs</code>	Logical. If <code>TRUE</code> , write the legacy CSV, <code>.RData</code> , text, and TIFF outputs to disk. The default is <code>FALSE</code> .

### Value

A list with class `"dnaEPICO_svaEnmix"`.

**targets** Phenotype table read from `phenoFile` after any optional row subsetting.

**RGSet** Loaded `RGChannelSet` with sample names realigned to `targets[[SampleID]]`.

**svaData** Object returned by `estimateSvaEnmixControls()` containing the surrogate-variable matrix and the control-probe settings used to estimate it.

**mergedPheno** Phenotype table returned by `mergeSvaTargetsEnmix()` after the surrogate variables were appended as additional columns.

**analysisData** Object returned by `analyzeSvaEnmix()` containing the surrogate-variable association models, ANOVA tables, and Sentrix metadata.

**plotFiles** Named list describing the plot file paths requested for the SVA figures. When `saveOutputs = FALSE`, the entries are typically `NULL`.

**savedFiles** Object returned by `writeSvaEnmixOutputs()` when `saveOutputs = TRUE`, otherwise `NULL`.

**logFile** Resolved path to the optional log file, or `NULL` when logging was disabled.

See [dnaEPICO\\_svaEnmix](#) for a class-level overview.

### See Also

[dnaEPICO\\_svaEnmix](#)

**Examples**

```

tmp <- tempdir()
stopifnot(dir.exists(tmp))

if (requireNamespace("minfiData", quietly = TRUE)) {
  ex <- dnaEPICO::exampleMinfiBaseDataDnaEpico()
  pheno_file <- file.path(tmp, "pheno.csv")
  rgset_path <- file.path(tmp, "RGSet.RData")
  RGSet <- ex$RGSet
  utils::write.csv(ex$targets, pheno_file, row.names = FALSE)
  save(RGSet, file = rgset_path)
  sva_result <- svaEnmix(
    phenoFile = pheno_file,
    rgsetData = rgset_path,
    SampleID = "Sample_Name",
    arrayType = "IlluminaHumanMethylation450k",
    annotationVersion = "ilmn12.hg19",
    SentrixIDColumn = "Sentrix_ID",
    SentrixPositionColumn = "Sentrix_Position",
    outputLogs = file.path(tmp, "logs"),
    figureBaseDir = file.path(tmp, "figures"),
    dataBaseDir = file.path(tmp, "data"),
    rBaseDir = file.path(tmp, "rData"),
    saveOutputs = FALSE
  )
  stopifnot(inherits(sva_result, "dnaEPICO_svaEnmix"))
}

```

---

```
writeMethylationGLM_T1Outputs
```

*Write optional disk outputs for one-timepoint GLM analyses*

---

**Description**

Write optional serialized outputs, summary tables, significant-CpG tables, and annotated results from the one-timepoint GLM workflow.

**Usage**

```

writeMethylationGLM_T1Outputs(
  modelResults,
  modelSummaries,
  annotatedResults,
  significantCpGs = NULL,
  outputRData,
  summaryTxtDir,
  significantCpGDir,
  annotatedGLMOut,

```

```

    saveTxtSummaries = TRUE,
    saveSignificantCpGs = FALSE,
    verbose = FALSE,
    logs = FALSE,
    log_dir = NULL,
    log_file = "log_methylationGLM_T1.txt"
  )

```

## Arguments

**modelResults** Object returned by `fitMethylationGLM_T1Models()`.

**modelSummaries** Object returned by `summarizeMethylationGLM_T1Models()`.

**annotatedResults** Object returned by `annotateMethylationGLM_T1Summaries()` or a compatible data frame.

**significantCpGs** Object returned by `collectSignificantCpGsMethylationGLM_T1()` or `NULL`.

**outputRData** Character. Directory used for serialized model and summary outputs.

**summaryTxtDir** Character. Directory used for tab-delimited summary tables.

**significantCpGDir** Character. Directory used for significant-CpG coefficient tables.

**annotatedGLMOut** Character. Directory used for the annotated summary XLSX workbook.

**saveTxtSummaries** Logical. If `TRUE`, write tab-delimited summary tables.

**saveSignificantCpGs** Logical. If `TRUE`, write significant-CpG coefficient tables.

**verbose** Logical. If `TRUE`, emit progress messages with `message()`.

**logs** Logical. If `TRUE`, write the same messages to a log file.

**log\_dir** Character or `NULL`. Directory used for the log file when `logs = TRUE`.

**log\_file** Character. File name used when `logs = TRUE`.

## Value

A list with class `"dnaEPICO_methylationGLM_T1_paths"` containing the paths of the files written to disk.

## Examples

```

ex <- dnaEPICO:::exampleMethylationGLMStateDnaEpicco()
annotation_data <- annotateMethylationGLM_T1Summaries(
  modelSummaries = ex$modelSummaries,
  annotationObject = ex$annotationData,
  annotationCols = "Name,chr,pos",
  verbose = FALSE,
  logs = FALSE
)

```

```

)
significant_cpGs <- collectSignificantCpGsMethylationGLM_T1(
  modelResults = ex$modelResults,
  pvalThreshold = 1,
  verbose = FALSE,
  logs = FALSE
)
output_paths <- writeMethylationGLM_T1Outputs(
  modelResults = ex$modelResults,
  modelSummaries = ex$modelSummaries,
  annotatedResults = annotation_data,
  significantCpGs = significant_cpGs,
  outputRData = file.path(ex$tempDir, "models"),
  summaryTxtDir = file.path(ex$tempDir, "summary"),
  significantCpGDir = file.path(ex$tempDir, "significant"),
  annotatedGLMOut = file.path(ex$tempDir, "annotated"),
  saveTxtSummaries = TRUE,
  saveSignificantCpGs = TRUE,
  verbose = FALSE,
  logs = FALSE
)
names(output_paths)

```

---

```
writeMethylationGLMM_T1T2Outputs
```

*Write optional disk outputs for longitudinal mixed-effects analyses*

---

### Description

Write optional serialized outputs, summary tables, significant interaction tables, and annotated results from the longitudinal mixed-effects workflow.

### Usage

```

writeMethylationGLMM_T1T2Outputs(
  modelResults,
  modelSummaries,
  annotatedResults,
  significantInteractions = NULL,
  outputRData,
  summaryTxtDir,
  significantInteractionDir,
  annotatedLMEOut,
  saveTxtSummaries = TRUE,
  saveSignificantInteractions = FALSE,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,

```

```

    log_file = "log_methylationGLMM_T1T2.txt"
  )

```

### Arguments

**modelResults** Object returned by `fitMethylationGLMM_T1T2Models()`.

**modelSummaries** Object returned by `summarizeMethylationGLMM_T1T2Models()`.

**annotatedResults** Object returned by `annotateMethylationGLMM_T1T2Summaries()` or a compatible data frame.

**significantInteractions** Object returned by `collectSignificantInteractionsMethylationGLMM_T1T2()` or NULL.

**outputRData** Character. Directory used for serialized model and summary outputs.

**summaryTxtDir** Character. Directory used for tab-delimited summary tables.

**significantInteractionDir** Character. Directory used for significant interaction coefficient tables.

**annotatedLMEOut** Character. Directory used for the annotated summary XLSX workbook.

**saveTxtSummaries** Logical. If TRUE, write tab-delimited summary tables.

**saveSignificantInteractions** Logical. If TRUE, write significant interaction coefficient tables.

**verbose** Logical. If TRUE, emit progress messages with `message()`.

**logs** Logical. If TRUE, write the same messages to a log file.

**log\_dir** Character or NULL. Directory used for the log file when `logs = TRUE`.

**log\_file** Character. File name used when `logs = TRUE`.

### Value

A list with class `"dnaEPICO_methylationGLMM_T1T2_paths"` containing the paths of the files written to disk.

### Examples

```

ex <- dnaEPICO::exampleMethylationGLMMStateDnaEpico()
annotation_data <- annotateMethylationGLMM_T1T2Summaries(
  modelSummaries = ex$modelSummaries,
  annotationObject = ex$annotationData,
  annotationCols = "Name,chr,pos",
  verbose = FALSE,
  logs = FALSE
)
significant_hits <- collectSignificantInteractionsMethylationGLMM_T1T2(
  modelResults = ex$modelResults,
  pvalThreshold = 1,
  verbose = FALSE,

```

```

    logs = FALSE
  )
  output_paths <- writeMethylationGLMM_T1T2Outputs(
    modelResults = ex$modelResults,
    modelSummaries = ex$modelSummaries,
    annotatedResults = annotation_data,
    significantInteractions = significant_hits,
    outputRData = file.path(ex$tempDir, "models"),
    summaryTxtDir = file.path(ex$tempDir, "summary"),
    significantInteractionDir = file.path(ex$tempDir, "significant"),
    annotatedLMEOut = file.path(ex$tempDir, "annotated"),
    saveTxtSummaries = TRUE,
    saveSignificantInteractions = TRUE,
    verbose = FALSE,
    logs = FALSE
  )
  names(output_paths)

```

---

```
writePhenoLCMinfiEwasWater
```

*Write the merged phenotype plus cell-composition table*

---

## Description

Write the merged phenotype plus cell-composition table

## Usage

```

writePhenoLCMinfiEwasWater(
  lcData,
  file,
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_writePhenoLCMinfiEwasWater.txt"
)

```

## Arguments

lcData	Object returned by estimateLCMinfiEwasWater().
file	Character. Path to the CSV file to write.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

Invisibly returns file.

**Examples**

```
ref_file <- system.file("extdata", "saliva.txt", package = "dnaEPICO")
beta <- as.matrix(utils::read.table(ref_file))[1:20, , drop = FALSE]
colnames(beta) <- c("sample1", "sample2")
targets <- data.frame(
  Sample_Name = colnames(beta),
  Timepoint = c("T1", "T2"),
  stringsAsFactors = FALSE
)
lc_data <- estimateLCMinfiEwasWater(
  beta = beta,
  targets = targets,
  lcRef = "saliva",
  phenoOrder = "Sample_Name;Timepoint"
)
output_file <- file.path(tempdir(), "phenoLC.csv")
writePhenoLCMinfiEwasWater(lcData = lc_data, file = output_file)
file.exists(output_file)
```

---

writePreprocessingPhenoOutputs

*Write legacy preprocessingPheno outputs to disk*

---

**Description**

Write the legacy CSV, ZIP, and .RData outputs produced by preprocessingPheno(). This helper keeps file writing separate from the in-memory preprocessing steps.

**Usage**

```
writePreprocessingPhenoOutputs(
  preprocessingData,
  outputPheno = "data/preprocessingPheno",
  outputRData = "rData/preprocessingPheno/metrics",
  outputRDataMerge = "rData/preprocessingPheno/mergeData",
  outputDir = "data/preprocessingPheno",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_writePreprocessingPhenoOutputs.txt"
)
```

**Arguments**

preprocessingData	Object returned by preprocessingPheno() or a list with the same components.
outputPheno	Character. Directory used for saved phenotype CSV files.
outputRData	Character. Directory used for saved metric .RData files.
outputRDataMerge	Character. Directory used for saved merged phenotype-plus-beta .RData files.
outputDir	Character. Directory used for the Clock Foundation export files.
verbose	Logical. If TRUE, emit progress messages with message().
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_preprocessingPheno\_paths" containing the paths written to disk.

**Examples**

```
ex <- dnaEPICO::examplePreprocessingPhenoStateDnaEpico()
output_paths <- writePreprocessingPhenoOutputs(
  preprocessingData = ex$preprocessingData,
  outputPheno = file.path(ex$tempDir, "pheno"),
  outputRData = file.path(ex$tempDir, "metrics"),
  outputRDataMerge = file.path(ex$tempDir, "merge"),
  outputDir = file.path(ex$tempDir, "clock"),
  verbose = FALSE,
  logs = FALSE
)
names(output_paths)
```

---

writeSvaEnmixOutputs *Write svaEnmix outputs to disk*

---

**Description**

Write the legacy CSV, .RData, and text-summary outputs used by the original svaEnmix() workflow.

**Usage**

```
writeSvaEnmixOutputs(
  svaData,
  mergedPheno,
  analysisData = NULL,
  phenoFile = NULL,
  dataBaseDir = "data",
  rBaseDir = "rData",
  scriptLabel = "svaEnmix",
  verbose = FALSE,
  logs = FALSE,
  log_dir = NULL,
  log_file = "log_writeSvaEnmixOutputs.txt"
)
```

**Arguments**

svaData	Object returned by <code>estimateSvaEnmixControls()</code> .
mergedPheno	Phenotype data frame returned by <code>mergeSvaTargetsEnmix()</code> .
analysisData	Optional object returned by <code>analyzeSvaEnmix()</code> .
phenoFile	Character or NULL. When supplied, mergedPheno is written back to this path for legacy compatibility.
dataBaseDir	Character. Base directory used for saved data outputs.
rBaseDir	Character. Base directory used for saved .RData outputs.
scriptLabel	Character. Label used to create the output subdirectory.
verbose	Logical. If TRUE, emit progress messages with <code>message()</code> .
logs	Logical. If TRUE, write the same messages to a log file.
log_dir	Character or NULL. Directory used for the log file when logs = TRUE.
log_file	Character. File name used when logs = TRUE.

**Value**

A list with class "dnaEPICO\_svaEnmix\_paths" containing the paths written to disk.

**Examples**

```
ex <- dnaEPICO::exampleSvaAnalysisStateDnaEpic()
temp_dir <- tempdir()
output_paths <- writeSvaEnmixOutputs(
  svaData = list(sva = ex$sva),
  mergedPheno = ex$mergedPheno,
  analysisData = ex$analysisData,
  phenoFile = file.path(temp_dir, "phenoLC.csv"),
  dataBaseDir = file.path(temp_dir, "data"),
  rBaseDir = file.path(temp_dir, "rData"),
  scriptLabel = "svaEnmixExample",
```

```
    verbose = FALSE,  
    logs = FALSE  
  )  
  names(output_paths)
```

# Index

\* **package**  
dnaEPICO, 14

analyzeSvaEnmix, 3  
analyzeSvaEnmix(), 19, 79  
annotateMethylationGLM\_T1Summaries, 5  
annotateMethylationGLM\_T1Summaries(),  
16, 37  
annotateMethylationGLMM\_T1T2Summaries,  
6  
annotateMethylationGLMM\_T1T2Summaries(),  
17, 41  
assessSamplesMinfiEwasWater, 7  
assessSamplesMinfiEwasWater(), 18, 64

buildClockFoundationInputsPreprocessingPheno,  
8  
buildClockFoundationInputsPreprocessingPheno(),  
18, 67  
buildRawMinfiEwasWater, 9  
buildRawMinfiEwasWater(), 18, 64

collectSignificantCpGsMethylationGLM\_T1,  
10  
collectSignificantCpGsMethylationGLM\_T1(),  
16, 37  
collectSignificantInteractionsMethylationGLMM\_T1T2,  
11  
collectSignificantInteractionsMethylationGLMM\_T1T2(),  
17, 41  
combineTimepointsPreprocessingPheno,  
13  
combineTimepointsPreprocessingPheno(),  
18, 67

dnaEPICO, 14  
dnaEPICO-package (dnaEPICO), 14  
dnaEPICO\_dnamReport  
(dnaEPICO\_dnamReport-class), 15  
dnaEPICO\_dnamReport-class, 15  
dnaEPICO\_dnamReport\_prepared  
(dnaEPICO\_dnamReport\_prepared-class),  
15  
dnaEPICO\_dnamReport\_prepared-class, 15  
dnaEPICO\_dnamReport\_render  
(dnaEPICO\_dnamReport\_render-class),  
16  
dnaEPICO\_dnamReport\_render-class, 16  
dnaEPICO\_methylationGLM\_T1, 37  
dnaEPICO\_methylationGLM\_T1  
(dnaEPICO\_methylationGLM\_T1-class),  
16  
dnaEPICO\_methylationGLM\_T1-class, 16  
dnaEPICO\_methylationGLMM\_T1T2, 41  
dnaEPICO\_methylationGLMM\_T1T2  
(dnaEPICO\_methylationGLMM\_T1T2-class),  
17  
dnaEPICO\_methylationGLMM\_T1T2-class,  
17  
dnaEPICO\_preprocessingMinfiEwasWater,  
64  
dnaEPICO\_preprocessingMinfiEwasWater  
(dnaEPICO\_preprocessingMinfiEwasWater-class),  
17  
dnaEPICO\_preprocessingMinfiEwasWater-class,  
17  
dnaEPICO\_preprocessingPheno, 67  
dnaEPICO\_preprocessingPheno  
(dnaEPICO\_preprocessingPheno-class),  
18  
dnaEPICO\_preprocessingPheno-class, 18  
dnaEPICO\_svaEnmix, 79  
dnaEPICO\_svaEnmix  
(dnaEPICO\_svaEnmix-class), 19  
dnaEPICO\_svaEnmix-class, 19  
dnamReport, 19  
dnamReport(), 14, 15, 69

estimateLC, 22  
estimateLCMinfiEwasWater, 23

- estimateLCMinfiEwasWater(), [18](#), [64](#)
- estimateSvaEnmixControls, [24](#)
- estimateSvaEnmixControls(), [19](#), [79](#)
- extractMake, [25](#)
- extractMetricsMinfiEwasWater, [26](#)
- extractMetricsMinfiEwasWater(), [18](#), [64](#)
  
- filterProbesMinfiEwasWater, [27](#)
- filterProbesMinfiEwasWater(), [18](#), [64](#)
- filterSamplesMinfiEwasWater, [28](#)
- fitMethylationGLM\_T1Models, [30](#)
- fitMethylationGLM\_T1Models(), [16](#), [37](#)
- fitMethylationGLMM\_T1T2Models, [31](#)
- fitMethylationGLMM\_T1T2Models(), [17](#), [41](#)
  
- loadMetricsPreprocessingPheno, [32](#)
- loadMetricsPreprocessingPheno(), [18](#), [67](#)
  
- mergeSvaTargetsEnmix, [33](#)
- mergeSvaTargetsEnmix(), [19](#), [79](#)
- methylationGLM\_T1, [34](#)
- methylationGLM\_T1(), [14](#), [16](#), [17](#)
- methylationGLMM\_T1T2, [38](#)
- methylationGLMM\_T1T2(), [14](#), [17](#)
  
- normalizeMinfiEwasWater, [42](#)
- normalizeMinfiEwasWater(), [18](#), [64](#)
  
- plotAssessmentMinfiEwasWater, [43](#)
- plotCtrlMinfiEwasWater, [45](#)
- plotMethylationGLM\_T1Diagnostics, [46](#)
- plotMethylationGLM\_T1Diagnostics(), [16](#), [37](#)
- plotMethylationGLM\_T1Distributions, [47](#)
- plotMethylationGLM\_T1Distributions(), [16](#), [37](#)
- plotMethylationGLMM\_T1T2Diagnostics, [48](#)
- plotMethylationGLMM\_T1T2Diagnostics(), [17](#), [41](#)
- plotMetricsMinfiEwasWater, [50](#)
- plotNormalizationMinfiEwasWater, [51](#)
- plotRawDensityMinfiEwasWater, [52](#)
- plotSexMinfiEwasWater, [54](#)
- plotSvaEnmix, [55](#)
- predictSexMinfiEwasWater, [56](#)
- predictSexMinfiEwasWater(), [18](#), [64](#)
- prepareDnamReportInputs, [57](#)
- prepareDnamReportInputs(), [15](#), [16](#)
  
- prepareMethylationGLM\_T1Data, [58](#)
- prepareMethylationGLM\_T1Data(), [16](#), [37](#)
- prepareMethylationGLMM\_T1T2Data, [60](#)
- prepareMethylationGLMM\_T1T2Data(), [17](#), [41](#)
- preprocessingMinfiEwasWater, [62](#)
- preprocessingMinfiEwasWater(), [14](#), [17](#), [18](#)
- preprocessingPheno, [65](#)
- preprocessingPheno(), [14](#), [18](#)
- print.dnaEPICO\_dnamReport, [68](#)
  
- readPhenotypeTargets, [69](#)
- readRGSetMinfiEwasWater, [70](#)
- renderDnamReport, [71](#)
- renderDnamReport(), [16](#)
  
- splitTimepointsPreprocessingPheno, [72](#)
- splitTimepointsPreprocessingPheno(), [18](#), [67](#)
- summarizeMethylationGLM\_T1Models, [74](#)
- summarizeMethylationGLM\_T1Models(), [16](#), [37](#)
- summarizeMethylationGLMM\_T1T2Models, [75](#)
- summarizeMethylationGLMM\_T1T2Models(), [17](#), [41](#)
- summarizeTimepointsMethylationGLMM\_T1T2, [76](#)
- svaEnmix, [77](#)
- svaEnmix(), [14](#), [19](#)
  
- writeMethylationGLM\_T1Outputs, [80](#)
- writeMethylationGLM\_T1Outputs(), [16](#), [37](#)
- writeMethylationGLMM\_T1T2Outputs, [82](#)
- writeMethylationGLMM\_T1T2Outputs(), [17](#), [41](#)
- writePhenoLCMinfiEwasWater, [84](#)
- writePreprocessingPhenoOutputs, [85](#)
- writePreprocessingPhenoOutputs(), [18](#), [67](#)
- writeSvaEnmixOutputs, [86](#)
- writeSvaEnmixOutputs(), [19](#), [79](#)