

# Package ‘SpatialDecon’

April 12, 2022

**Title** Deconvolution of mixed cells from spatial and/or bulk gene expression data

**Version** 1.4.3

**Description** Using spatial or bulk gene expression data, estimates abundance of mixed cell types within each observation. Based on “Advances in mixed cell deconvolution enable quantification of cell types in spatially-resolved gene expression data”, Danaher (2020). Designed for use with the NanoString GeoMx platform, but applicable to any gene expression data.

**Depends** R (>= 4.0.0)

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** TRUE

**RoxygenNote** 7.1.2

**Imports** grDevices, stats, utils, graphics, SeuratObject, Biobase, GeomxTools, repmis, methods, Matrix

**Suggests** testthat, knitr, rmarkdown, qpdf

**biocViews** ImmunoOncology, FeatureExtraction, GeneExpression, Transcriptomics

**VignetteBuilder** knitr

**BugReports** <https://github.com/Nanostring-Biostats/SpatialDecon/issues>

**git\_url** <https://git.bioconductor.org/packages/SpatialDecon>

**git\_branch** RELEASE\_3\_14

**git\_last\_commit** 72c2c83

**git\_last\_commit\_date** 2021-11-02

**Date/Publication** 2022-04-12

**Author** Nicole Ortogero [cre],  
Patrick Danaher [aut],  
Maddy Griswold [aut]

**Maintainer** Nicole Ortogero <nortogero@nanostring.com>

**R topics documented:**

|  |           |
|--|-----------|
| SpatialDecon-package                           | 2         |
| cellcols                                       | 4         |
| collapseCellTypes                              | 4         |
| create_profile_matrix                          | 5         |
| derive_GeoMx_background                        | 7         |
| download_profile_matrix                        | 7         |
| florets  | 8         |
| mean.resid.sd                                  | 10        |
| mergeTumorIntoX                                | 10        |
| mini_geomx_dataset                             | 11        |
| mini_singleCell_dataset                        | 12        |
| nsclc  | 12        |
| reverseDecon                                   | 13        |
| runCollapseCellTypes                           | 14        |
| runCollapseCellTypes,NanoStringGeoMxSet-method | 15        |
| runErrorModel                                  | 16        |
| runMergeTumorIntoX                             | 16        |
| runMergeTumorIntoX,NanoStringGeoMxSet-method   | 17        |
| runReverseDecon                                | 18        |
| runReverseDecon,NanoStringGeoMxSet-method      | 19        |
| runspatialdecon                                | 20        |
| runspatialdecon,NanoStringGeoMxSet-method      | 21        |
| runspatialdecon,Seurat-method                  | 24        |
| safeTME  | 26        |
| safeTME.matches                                | 26        |
| spatialdecon                                   | 27        |
| TIL_barplot                                    | 29        |
| <b>Index</b>                                   | <b>31</b> |

---

SpatialDecon-package    *SpatialDecon: A package for computing the notorious bar statistic.*

---

**Description**

The SpatialDecon package estimates mixed cell type abundance in the regions of spatially-resolved gene expression studies, using the method of Danaher & Kim (2020), "Advances in mixed cell deconvolution enable quantification of cell types in spatially-resolved gene expression data." It is also appropriate to apply to bulk gene expression data.

**functions**

Functions to help set up deconvolution:

- `derive_GeoMx_background` Estimates the background levels from GeoMx experiments
- `collapseCellTypes` reformats deconvolution results to merge closely-related cell types

- `download_profile_matrix` Downloads a cell profile matrix.
- `safeTME`: a data object, a matrix of immune cell profiles for use in tumor-immune deconvolution.

Deconvolution functions:

- `spatialdecon` runs the core deconvolution function
- `reverseDecon` runs a transposed/reverse deconvolution problem, fitting the data as a function of cell abundance estimates. Used to measure genes' dependency on cell mixing and to calculate gene residuals from cell mixing.

Plotting functions:

- `florets` Plot cell abundance on a specified x-y space, with each point a cockscomb plot showing the cell abundances of that region/sample.
- `TIL_barplot` Plot abundances of tumor infiltrating lymphocytes (TILs) estimated from the `safeTME` cell profile matrix

## Examples

```
data(mini_geomx_dataset)
data(safeTME)
data(safeTME.matches)
# estimate background:
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
# run basic decon:
res0 <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
  bg = mini_geomx_dataset$bg,
  X = safeTME
)
# run decon with bells and whistles:
res <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
  bg = mini_geomx_dataset$bg,
  X = safeTME,
  cellmerges = safeTME.matches,
  cell_counts = mini_geomx_dataset$annot$nuclei,
  is_pure_tumor = mini_geomx_dataset$annot$AOI.name == "Tumor"
)
```

---

|          |  |
|----------|--|
| cellcols | <i>Default colors for the cell types in the safeTME matrix</i> |
|----------|--|

---

**Description**

A named vector of colors, giving colors for the cell types of the safeTME matrix.

**Usage**

```
cellcols
```

**Format**

A named vector

---

|                   |  |
|-------------------|--|
| collapseCellTypes | <i>Collapse related cell types within a deconvolution result</i> |
|-------------------|--|

---

**Description**

Given the input of an SpatialDecon result output and a list of which cell types to combine, returns a reshaped deconvolution result object with the specified cell types merged.

**Usage**

```
collapseCellTypes(fit, matching)
```

**Arguments**

|          |   |
|----------|---|
| fit      | The object (a list) returned by the SpatialDecon algorithm  |
| matching | A list object holding the mapping from beta's cell names to official cell names. See str(safeTME.matches) for an example. |

**Value**

A reshaped deconvolution result object

## Examples

```
data(mini_geomx_dataset)
data(safeTME)
data(safeTME.matches)
# estimate background:
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
# run basic decon:
res0 <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
  bg = mini_geomx_dataset$bg,
  X = safeTME
)
res1 <- collapseCellTypes(
  fit = res0,
  matching = safeTME.matches
)
```

---

create\_profile\_matrix *Create Custom Cell Profile Matrix*

---

## Description

Create custom cell profile matrix using single cell data. The average gene expression for each cell type is returned.

## Usage

```
create_profile_matrix(
  mtx,
  cellAnnots,
  cellTypeCol,
  cellNameCol,
  matrixName = "Custom",
  outDir = "./",
  geneList = NULL,
  normalize = FALSE,
  scalingFactor = 5,
  minCellNum = 15,
  minGenes = 100,
  discardCellTypes = FALSE
)
```

**Arguments**

|                  |  |
|------------------|--|
| mtx              | cell x gene count matrix   |
| cellAnnots       | cell annotations with cell type and cell name as columns                                       |
| cellTypeCol      | column containing cell type  |
| cellNameCol      | column containing cell ID/name   |
| matrixName       | name of final profile matrix   |
| outDir           | path to desired output directory, set to NULL if matrix should not be written                  |
| geneList         | gene list to filter profile matrix to  |
| normalize        | Should data be normalized? (TRUE/FALSE) if TRUE data will be normalized using total gene count |
| scalingFactor    | what should all values be multiplied by for final matrix, set to 1 if no scaling is wanted     |
| minCellNum       | minimum number of cells of one type needed to create profile, exclusive                        |
| minGenes         | minimum number of genes expressed in a cell, exclusive   |
| discardCellTypes | should cell types be filtered for types like mitotic, doublet, low quality, unknown, etc.      |

**Value**

A custom cell profile matrix genes (rows) by cell types (columns), matrix gets written to disk and outDir

**Examples**

```

cellNames <- paste0("Cell", seq_len(1500))
geneNames <- paste0("Gene", seq_len(1500))
mtx <- matrix(data=sample(size = length(cellNames)*length(geneNames),
                        replace = TRUE,
                        x = c(0,seq_len(100)),
                        prob = c(0.6784, rep(0.0075, 15), rep(0.005, 25),
                                rep(0.002, 25), rep(0.001, 35))),
             ncol = length(cellNames), nrow = length(geneNames),
             dimnames = list(geneNames, cellNames))
cellAnnots <- as.data.frame(cbind(CellID=cellNames,
                                cellType=sample(size = length(cellNames),
                                                replace = TRUE,
                                                x = c("A", "B", "C", "D"),
                                                prob = c(0.1, 0.4, 0.3, 0.2))))

table(cellAnnots$cellType)
profile_matrix <- create_profile_matrix(mtx = mtx,
                                       cellAnnots = cellAnnots,
                                       cellTypeCol = "cellType",
                                       cellNameCol = "CellID",
                                       minGenes = 10,
                                       scalingFactor = 1)

head(profile_matrix)

```

---

`derive_GeoMx_background`*Derive background at the scale of the normalized data for GeoMx data*

---

**Description**

Estimates per-datapoint background levels from a GeoMx experiment. In studies with two or more probe pools, different probes will have different background levels. This function provides a convenient way to account for this phenomenon.

**Usage**

```
derive_GeoMx_background(norm, probepool, negnames)
```

**Arguments**

|                        |   |
|------------------------|---|
| <code>norm</code>      | Matrix of normalized data, genes in rows and segments in columns. Must include negprobes, and must have rownames. |
| <code>probepool</code> | Vector of probe pool names for each gene, aligned to the rows of "norm".  |
| <code>negnames</code>  | Names of all negProbes in the dataset. Must be at least one neg.name within each probe pool.                      |

**Value**

A matrix of expected background values, in the same scale and dimensions as the "norm" argument.

**Examples**

```
data(mini_geomx_dataset)
# estimate background:
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
```

---

`download_profile_matrix`*Download a cell profile matrix*

---

**Description**

Download a cell profile matrix from the online library

**Usage**

```
download_profile_matrix(species, age_group, matrixname)
```

**Arguments**

|            |   |
|------------|---|
| species    | species of profile matrix   |
| age_group  | age_group of profile matrix, if fetal mouse please add the developmental stage separated with /, i.e. Fetal/E14.5 |
| matrixname | name of profile matrix  |

**Details**

Valid matrices can be found on the github site <https://github.com/Nanostring-Biostats/CellProfileLibrary/tree/NewProfileMatrices>

**Value**

A cell profile matrix, suggested cell groups, and paper metadata

**Examples**

```
download_profile_matrix(species = "Human", age_group = "Adult", matrixname = "Colon_HCA")
head(profile_matrix)
print(cellGroups)
print(metadata)
```

---

florets

*Draw coxcomb plots as points in a graphics window*

---

**Description**

Draws a scatterplot where each point is a circular barplot, intended to show decon results

**Usage**

```
florets(
  x,
  y,
  b,
  col = NULL,
  legendwindow = FALSE,
  rescale.by.sqrt = TRUE,
  border = NA,
  add = FALSE,
  cex = 1,
  bty = "n",
  xaxt = "n",
```



```

    yaxt = "n",
    xlab = "",
    ylab = "",
    ...
)

```

### Arguments

|                              |  |
|------------------------------|--|
| <code>x</code>               | Vector of x coordinates  |
| <code>y</code>               | Vector of y coordinates  |
| <code>b</code>               | matrix or cell abundances, with columns aligned with the elements of x and y                                       |
| <code>col</code>             | vector of colors, aligned to the rows of b.  |
| <code>legendwindow</code>    | Logical. If TRUE, the function draws a color legend in a new window  |
| <code>rescale.by.sqrt</code> | Logical, for whether to rescale b by its square root to make value proportional to shape area, not shape length.   |
| <code>border</code>          | Color of pie segment border, defaults to NA/none   |
| <code>add</code>             | Logical. If TRUE, the function draws florets atop an existing graphics device (TRUE) or call a new device (FALSE). |
| <code>cex</code>             | Floret size. Florets are scaled relative to the range of x and y; this further scales up or down.                  |
| <code>bty</code>             | bty argument passed to plot()  |
| <code>xaxt</code>            | xaxt argument passed to plot()   |
| <code>yaxt</code>            | yaxt argument passed to plot()   |
| <code>xlab</code>            | xlab, defaults to ""   |
| <code>ylab</code>            | ylab, defaults to ""   |
| <code>...</code>             | additional arguments passed to plot()  |

### Value

Draws a coxcomb plot, returns no data.

### Examples

```

data(mini_geomx_dataset)
data(safeTME)
# estimate background:
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
# run basic decon:
res0 <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
  bg = mini_geomx_dataset$bg,

```

```

    X = safeTME
  )
  # draw florets:
  florets(
    x = mini_geomx_dataset$annot$x,
    y = mini_geomx_dataset$annot$y,
    b = res0$beta, cex = 2
  )

```

---

mean.resid.sd

*Genes' biological variability in immune deconvolution from TCGA.*

---

### Description

Genes' biological SDs, as estimated from immune deconvolution from TCGA. Used to weight genes in spatialdecon.

### Usage

```
mean.resid.sd
```

### Format

A named vector giving SDs of 1179 genes.

---

mergeTumorIntoX

*Estimate a tumor-specific profile and merge it with the pre-specified cell profile matrix (X)*

---

### Description

Given the input of "tumor-only" AOI's, estimates an collection of tumor-specific expression profiles and merges them with the immune cell expression training matrix. The process:

1. log2/normalized data from tumor-only AOIs is clustered with hclust, and cutree() is used to define clusters.
2. Each cluster's geomean profile is merged into the immune cell profile matrix.

### Usage

```
mergeTumorIntoX(norm, bg, pure_tumor_ids, X, K = 10)
```

**Arguments**

|                |   |
|----------------|---|
| norm           | matrix of normalized data   |
| bg             | matrix of expected background, on the scale of norm.  |
| pure_tumor_ids | Vector identifying columns of norm that are pure tumor. Can be indices, logicals or column names. |
| X              | The training matrix   |
| K              | the number of clusters to fit   |

**Value**

an updated X matrix with new columns, "tumor.1", "tumor.2", ...

**Examples**

```

data(mini_geomx_dataset)
data(safeTME)
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
safeTME.with.tumor <- mergeTumorIntoX(
  norm = mini_geomx_dataset$norm,
  bg = mini_geomx_dataset$bg,
  pure_tumor_ids = mini_geomx_dataset$annot$AOI.name == "Tumor",
  X = safeTME,
  K = 3
)

```

---

mini\_geomx\_dataset      *Small example GeoMx data*

---

**Description**

A miniature GeoMx dataset used by the spatialdecon examples.

**Usage**

```
mini_geomx_dataset
```

**Format**

A list with the following elements:

- normalized: normalized data matrix
- raw: raw data matrix
- annot: AOI annotation data frame

---

mini\_singleCell\_dataset

*Mini human colon single cell dataset*

---

### **Description**

Random 250 cells and most informative genes (CV > 10) between cell types from Kinchen, J. et al. Structural Remodeling of the Human Colonic Mesenchyme in Inflammatory Bowel Disease. Cell 175, 372-386.e17 (2018).

### **Usage**

mini\_singleCell\_dataset

### **Format**

A list with the following elements:

- mtx: sparse count matrix
- annots: cell type annotation data frame

---

nslc

*Large example GeoMx data*

---

### **Description**

A GeoMx dataset with dense AOIs gridded over a NSCLC tumor. Each AOI is split into tumor and microenvironment segments.

### **Usage**

nslc

### **Format**

GeoMxSet Object

---

|              |                              |
|--------------|------------------------------|
| reverseDecon | <i>Reverse deconvolution</i> |
|--------------|------------------------------|

---

### Description

Performs "reverse deconvolution", modelling each gene expression's ~ cell scores. Returns a matrix of "fitted" expression values, a matrix of residuals, a matrix of reverse decon coefficients for genes \* cells.

### Usage

```
reverseDecon(norm, beta, epsilon = NULL)
```

### Arguments

|         |  |
|---------|--|
| norm    | Matrix of normalized data, with genes in rows and observations in columns  |
| beta    | Matrix of cell abundance estimates, with cells in rows and observations in columns. Columns are aligned to "norm".                                   |
| epsilon | All y and yhat values are thresholded up to this point when performing decon. Essentially says, "ignore variability in counts below this threshold." |

### Value

A list:

- coefs, a matrix of coefficients for genes \* cells, where element i,j is interpreted as "every unit increase in cell score j is expected to increase expression of gene i by \_".
- yhat, a matrix of fitted values, in the same dimension as norm
- resids, a matrix of log2-scale residuals from the reverse decon fit, in the same dimension as norm
- cors, a vector giving each gene's correlation between fitted and observed expression
- resid.sd, a vector of each gene's residual SD, a metric of how much variability genes have independent of cell mixing.

### Examples

```
data(mini_geomx_dataset)
data(safeTME)
# estimate background:
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
# run basic decon:
res0 <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
```

```

    bg = mini_geomx_dataset$bg,
    X = safeTME
  )
# run reverse decon:
rdecon <- reverseDecon(
  norm = mini_geomx_dataset$norm,
  beta = res0$beta
)

```

---

runCollapseCellTypes    *Run collapseCellTypes*

---

### Description

Runs collapseCellTypes from an S4 object

### Usage

```
runCollapseCellTypes(object, ...)
```

### Arguments

|        |  |
|--------|--|
| object | An S4 object such as a GeoMxSet object |
| ...    | Arguments passed to collapseCellTypes  |

### Value

A reshaped deconvolution result object

### Examples

```

library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "/demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

target_demoData <- normalize(target_demoData, "quant")

# run basic decon:
res0 <- runspatialdecon(object = target_demoData,
  norm_elt = "exprs_norm",
  raw_elt = "exprs")

# run reverse decon:
target_demoData <- runReverseDecon(object = target_demoData,
  norm_elt = "exprs_norm",
  beta = pData(res0)$beta)

```



---

|               |   |
|---------------|---|
| runErrorModel | <i>Apply error model to estimate technical SD from raw counts</i> |
|---------------|---|

---

**Description**

Based on raw counts, uses past data to estimate each raw count's log-scale SD from technical noise. Specifies different error models for different platforms.

**Usage**

```
runErrorModel(counts, platform = "general")
```

**Arguments**

|          |  |
|----------|--|
| counts   | vector or matrix of raw counts   |
| platform | String specifying which platform was used to create "rawCounts". Default to "dsp", for digital spatial profiler/ aka GeoMx. Other options include "ncounter", "rsem", "quantile", and "st" for spatial transcriptomics/visium. |

**Value**

a matrix of log2-scale SDs

**Examples**

```
library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

sd_from_noise <- runErrorModel(counts = exprs(target_demoData), platform = "dsp")
wts <- 1 / sd_from_noise
```

---

|                    |                            |
|--------------------|----------------------------|
| runMergeTumorIntoX | <i>Run MergeTumorIntoX</i> |
|--------------------|----------------------------|

---

**Description**

Runs mergeTumorIntoX from an S4 object

**Usage**

```
runMergeTumorIntoX(object, ...)
```



**Arguments**

object            An S4 object such as a GeoMxSet object  
 ...               Arguments passed to mergeTumorIntoX

**Value**

updated X matrix with new columns, "tumor.1", "tumor.2", ...

**Examples**

```
library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "/demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

target_demoData <- normalize(target_demoData, "quant")

data(safeTME)
tumor.ids <- as.logical(sample(x = c("TRUE","FALSE"), size = 88, replace = TRUE))
safeTME.with.tumor <- runMergeTumorIntoX(object = target_demoData,
                                         X = safeTME,
                                         K = 3,
                                         pure_tumor_ids = tumor.ids,
                                         norm_elt = "exprs_norm")
```

---

runMergeTumorIntoX,NanoStringGeoMxSet-method

*Run mergeTumorIntoX on a NanostringGeomxSet object*

---

**Description**

A wrapper for applying mergeTumorIntoX to a NanostringGeomxSet object.

**Usage**

```
## S4 method for signature 'NanoStringGeoMxSet'
runMergeTumorIntoX(object, X, K = 10, pure_tumor_ids = NULL, norm_elt = NULL)
```

**Arguments**

object            A NanostringGeomxSet object.  
 X                 The training matrix  
 K                 the number of clusters to fit  
 pure\_tumor\_ids   Vector identifying columns of norm that are pure tumor. Can be indices, logicals  
                   or column names.  
 norm\_elt         norm data element in assayData

**Value**

an updated X matrix with new columns, "tumor.1", "tumor.2", ...

**Examples**

```
library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "/demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

target_demoData <- normalize(target_demoData, "quant")

data(safeTME)
tumor.ids <- as.logical(sample(x = c("TRUE", "FALSE"), size = 88, replace = TRUE))
safeTME.with.tumor <- runMergeTumorIntoX(object = target_demoData,
                                         X = safeTME,
                                         K = 3,
                                         pure_tumor_ids = tumor.ids,
                                         norm_elt = "exprs_norm")
```

---

runReverseDecon

*Run Reversedecon*


---

**Description**

Runs reversedecon from an S4 object

**Usage**

```
runReverseDecon(object, ...)
```

**Arguments**

|        |  |
|--------|--|
| object | An S4 object such as a GeoMxSet object |
| ...    | Arguments passed to reversedecon       |

**Value**

list containing modeled gene expression's ~ cell scores

**Examples**

```

library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "/demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

target_demoData <- normalize(target_demoData, "quant")

# run basic decon:
res0 <- runspatialdecon(object = target_demoData,
                       norm_elt = "exprs_norm",
                       raw_elt = "exprs")

# run reverse decon:
target_demoData <- runReverseDecon(object = target_demoData,
                                   norm_elt = "exprs_norm",
                                   beta = pData(res0)$beta)

```

---

runReverseDecon,NanoStringGeoMxSet-method

*Run reversedecon on a NanostringGeomxSet object*

---

**Description**

A wrapper for applying reversedecon to a NanostringGeomxSet object.

**Usage**

```

## S4 method for signature 'NanoStringGeoMxSet'
runReverseDecon(object, norm_elt = NULL, beta, epsilon = NULL)

```

**Arguments**

|          |  |
|----------|--|
| object   | A NanostringGeomxSet object.   |
| norm_elt | normalized data element in assayData.  |
| beta     | Matrix of cell abundance estimates, with cells in columns and observations in rows. Columns are aligned to "norm".                                   |
| epsilon  | All y and yhat values are thresholded up to this point when performing decon. Essentially says, "ignore variability in counts below this threshold." |

**Value**

a valid GeoMx S4 object including the following items:

- in fData
  - coefs, a matrix of coefficients for genes \* cells, where element i,j is interpreted as "every unit increase in cell score j is expected to increase expression of gene i by \_".
  - cors, a vector giving each gene's correlation between fitted and observed expression
  - resid.sd, a vector of each gene's residual SD, a metric of how much variability genes have independent of cell mixing.
- in assayData
  - yhat, a matrix of fitted values, in the same dimension as norm
  - resids, a matrix of log2-scale residuals from the reverse decon fit, in the same dimension as norm

**Examples**

```
library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "/demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

target_demoData <- normalize(target_demoData, "quant")

# run basic decon:
res0 <- runspatialdecon(object = target_demoData,
                       norm_elt = "exprs_norm",
                       raw_elt = "exprs")

# run reverse decon:
target_demoData <- runReverseDecon(object = target_demoData,
                                   norm_elt = "exprs_norm",
                                   beta = pData(res0)$beta)
```

---

runspatialdecon

*Run spatialdecon*

---

**Description**

Runs spatialdecon from an S4 object

**Usage**

```
runspatialdecon(object, ...)
```

**Arguments**

object            An S4 object such as a Seurat object or a GeoMxSet object  
 ...                Arguments passed to spatialdecon

**Value**

decon results in list or in GeoMxSet object

**Examples**

```
##Seurat
# get dataset
con <- gzcon(url("https://github.com/almaan/her2st/raw/master/data/ST-cnts/G1.tsv.gz"))
txt <- readLines(con)
temp <- read.table(textConnection(txt), sep = "\t", header = TRUE, row.names = 1)
# parse data
raw = t(as.matrix(temp))
norm = sweep(raw, 2, colSums(raw), "/" ) * mean(colSums(raw))
x = as.numeric(substr(rownames(temp), 1, unlist(gregexpr("x", rownames(temp))) - 1))
y = -as.numeric(substr(rownames(temp),
                      unlist(gregexpr("x", rownames(temp))) + 1, nchar(rownames(temp))))
# put into a seurat object:
andersson_g1 = SeuratObject::CreateSeuratObject(counts = raw, assay="Spatial")
andersson_g1@meta.data$x = x
andersson_g1@meta.data$y = y

res <- runspatialdecon(andersson_g1)
str(res)

##GeomxTools
library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "/demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

target_demoData <- normalize(target_demoData, "quant")

demoData <- runspatialdecon(object = target_demoData,
                          norm_elt = "exprs_norm",
                          raw_elt = "exprs")
```

---

runspatialdecon, NanoStringGeoMxSet-method

*Run spatialdecon on a NanostringGeomxSet object*

---

**Description**

A wrapper for applying spatialdecon to a NanostringGeomxSet object.

**Usage**

```
## S4 method for signature 'NanoStringGeoMxSet'
runspatialdecon(
  object,
  X = NULL,
  norm_elt = NULL,
  raw_elt = NULL,
  wts = NULL,
  resid_thresh = 3,
  lower_thresh = 0.5,
  align_genes = TRUE,
  is_pure_tumor = NULL,
  n_tumor_clusters = 10,
  cell_counts = NULL,
  cellmerges = NULL,
  maxit = 1000
)
```

**Arguments**

|                               |   |
|-------------------------------|---|
| <code>object</code>           | A NanostringGeomxSet object.  |
| <code>X</code>                | Cell profile matrix. If NULL, the safeTME matrix is used.   |
| <code>norm_elt</code>         | normalized data element in assayData  |
| <code>raw_elt</code>          | raw data element in assayData   |
| <code>wts</code>              | Optional, a matrix of weights.  |
| <code>resid_thresh</code>     | A scalar, sets a threshold on how extreme individual data points' values can be (in log2 units) before getting flagged as outliers and set to NA.   |
| <code>lower_thresh</code>     | A scalar. Before log2-scale residuals are calculated, both observed and fitted values get thresholded up to this value. Prevents log2-scale residuals from becoming extreme in points near zero.  |
| <code>align_genes</code>      | Logical. If TRUE, then Y, X, bg, and wts are row-aligned by shared genes.   |
| <code>is_pure_tumor</code>    | A logical vector denoting whether each AOI consists of pure tumor. If specified, then the algorithm will derive a tumor expression profile and merge it with the immune profiles matrix.  |
| <code>n_tumor_clusters</code> | Number of tumor-specific columns to merge into the cell profile matrix. Has an impact only when <code>is_pure_tumor</code> argument is used to indicate pure tumor AOIs. Takes this many clusters from the pure-tumor AOI data and gets the average expression profile in each cluster. Default 10. |
| <code>cell_counts</code>      | Number of cells estimated to be within each sample. If provided alongside <code>norm_factors</code> , then the algorithm will additionally output cell abundance estimates on the scale of cell counts.   |
| <code>cellmerges</code>       | A list object holding the mapping from beta's cell names to combined cell names. If left NULL, then defaults to a mapping of granular immune cell definitions to broader categories.  |
| <code>maxit</code>            | Maximum number of iterations. Default 1000.   |

**Value**

if not given cellmerges and cell\_counts, a valid GeoMx S4 object including the following items

- In pData
  - beta: matrix of cell abundance estimates, cells in rows and observations in columns
  - p: matrix of p-values for  $H_0: \beta == 0$
  - t: matrix of t-statistics for  $H_0: \beta == 0$
  - se: matrix of standard errors of beta values
  - prop\_of\_all: rescaling of beta to sum to 1 in each observation
  - prop\_of\_nontumor: rescaling of beta to sum to 1 in each observation, excluding tumor abundance estimates
  - sigmas: covariance matrices of each observation's beta estimates
- In assayData
  - yhat: a matrix of fitted values
  - resids: a matrix of residuals from the model fit. ( $\log_2(\text{pmax}(y, \text{lower\_thresh})) - \log_2(\text{pmax}(xb, \text{lower\_thresh}))$ ).
- In experimentData
  - SpatialDeconMatrix: the cell profile matrix used in the decon fit.

if given cellmerges, the valid GeoMx S4 object will additionally include the following items

- In pData
  - beta.granular: cell abundances prior to combining closely-related cell types
  - sigma.granular: sigmas prior to combining closely-related cell types

if given cell\_counts, the valid GeoMx S4 object will additionally include the following items

- In pData
  - cell.counts: beta rescaled to estimate cell numbers, based on prop\_of\_all and nuclei count

if given both cellmerges and cell\_counts, the valid GeoMx S4 object will additionally include the following items

- In pData
  - cell.counts.granular: cell.counts prior to combining closely-related cell types

**Examples**

```
library(GeomxTools)
datadir <- system.file("extdata", "DSP_NGS_Example_Data", package = "GeomxTools")
demoData <- readRDS(file.path(datadir, "/demoData.rds"))

demoData <- shiftCountsOne(demoData)
target_demoData <- aggregateCounts(demoData)

target_demoData <- normalize(target_demoData, "quant")

demoData <- runspatialdecon(object = target_demoData,
                           norm_elt = "exprs_norm",
                           raw_elt = "exprs")
```

---

 runspatialdecon,Seurat-method

*Run spatialdecon on a Seurat object*


---

## Description

A wrapper for applying spatialdecon to the Spatial data element in a Seurat object. Unlike spatialdecon, which expects a normalized data matrix, this function operates on raw counts. Scaling for total cells

## Usage

```
## S4 method for signature 'Seurat'
runspatialdecon(
  object,
  X = NULL,
  bg = 0.1,
  wts = NULL,
  resid_thresh = 3,
  lower_thresh = 0.5,
  align_genes = TRUE,
  is_pure_tumor = NULL,
  n_tumor_clusters = 10,
  cell_counts = NULL,
  cellmerges = NULL,
  maxit = 1000
)
```

## Arguments

|               |  |
|---------------|--|
| object        | A seurat object. Must include a "Spatial" element in the "assays" slot.  |
| X             | Cell profile matrix. If NULL, the safeTME matrix is used.  |
| bg            | Expected background counts. Either a scalar applied equally to all points in the count matrix, or a matrix with the same dimensions as the count matrix in <code>GetAssayData(object, assay = "Spatial")</code> . Recommended to use a small non-zero value, default of 0.1. |
| wts           | Optional, a matrix of weights.   |
| resid_thresh  | A scalar, sets a threshold on how extreme individual data points' values can be (in log2 units) before getting flagged as outliers and set to NA.  |
| lower_thresh  | A scalar. Before log2-scale residuals are calculated, both observed and fitted values get thresholded up to this value. Prevents log2-scale residuals from becoming extreme in points near zero.   |
| align_genes   | Logical. If TRUE, then Y, X, bg, and wts are row-aligned by shared genes.  |
| is_pure_tumor | A logical vector denoting whether each AOI consists of pure tumor. If specified, then the algorithm will derive a tumor expression profile and merge it with the immune profiles matrix.   |



|                  |  |
|------------------|--|
| n_tumor_clusters | Number of tumor-specific columns to merge into the cell profile matrix. Has an impact only when is_pure_tumor argument is used to indicate pure tumor AOIs. Takes this many clusters from the pure-tumor AOI data and gets the average expression profile in each cluster. Default 10. |
| cell_counts      | Number of cells estimated to be within each sample. If provided alongside norm_factors, then the algorithm will additionally output cell abundance estimates on the scale of cell counts.  |
| cellmerges       | A list object holding the mapping from beta's cell names to combined cell names. If left NULL, then defaults to a mapping of granular immune cell definitions to broader categories.   |
| maxit            | Maximum number of iterations. Default 1000.  |

### Value

if not given cellmerges and cell\_counts, a list including the following items:

- beta: matrix of cell abundance estimates, cells in rows and observations in columns
- p: matrix of p-values for  $H_0: \beta == 0$
- t: matrix of t-statistics for  $H_0: \beta == 0$
- se: matrix of standard errors of beta values
- prop\_of\_all: rescaling of beta to sum to 1 in each observation
- prop\_of\_nontumor: rescaling of beta to sum to 1 in each observation, excluding tumor abundance estimates
- yhat: a matrix of fitted values
- resids: a matrix of residuals from the model fit. ( $\log_2(\text{pmax}(y, \text{lower\_thresh})) - \log_2(\text{pmax}(xb, \text{lower\_thresh}))$ ).
- X: the cell profile matrix used in the decon fit.
- sigmas: covariance matrices of each observation's beta estimates

if given cellmerges, the list will additionally include the following items

- beta.granular: cell abundances prior to combining closely-related cell types
- sigma.granular: sigmas prior to combining closely-related cell types

if given cell\_counts, the list will additionally include the following items

- cell.counts: beta rescaled to estimate cell numbers, based on prop\_of\_all and nuclei count

if given both cellmerges and cell\_counts, the list will additionally include the following items

- cell.counts.granular: cell.counts prior to combining closely-related cell types

**Examples**

```

# get dataset
con <- gzcon(url("https://github.com/almaan/her2st/raw/master/data/ST-cnts/G1.tsv.gz"))
txt <- readLines(con)
temp <- read.table(textConnection(txt), sep = "\t", header = TRUE, row.names = 1)
# parse data
raw = t(as.matrix(temp))
norm = sweep(raw, 2, colSums(raw), "/" * mean(colSums(raw)))
x = as.numeric(substr(rownames(temp), 1, unlist(gregexpr("x", rownames(temp))) - 1))
y = -as.numeric(substr(rownames(temp),
                      unlist(gregexpr("x", rownames(temp))) + 1, nchar(rownames(temp))))
# put into a seurat object:
andersson_g1 = SeuratObject::CreateSeuratObject(counts = raw, assay="Spatial")
andersson_g1@meta.data$x = x
andersson_g1@meta.data$y = y

res <- runspatialdecon(andersson_g1)
str(res)

```

safeTME

*SafeTME matrix***Description**

A matrix of expression profiles of 906 genes over 18 cell types.

**Usage**

```
safeTME
```

**Format**

A matrix with 906 genes (rows) and 18 cell types (columns)

safeTME.matches

*Mapping from granularly-defined cell populations to broadened cell populations***Description**

Mapping from granularly-defined cell populations to broadened cell populations, for use by the `convertCellTypes` function.

**Usage**

```
safeTME.matches
```

**Format**

A list. Each element of the list contains the granular cell types that roll up to a single coarse cell type.

---

spatialdecon

*Mixed cell deconvolution of spatially-resolved gene expression data*


---

**Description**

Runs the spatialdecon algorithm with added optional functionalities. Workflow is:

1. compute weights from raw data
2. Estimate a tumor profile and merge it into the cell profiles matrix
3. run deconvolution once
4. remove poorly-fit genes from first round of decon
5. re-run decon with cleaned-up gene set
6. combine closely-related cell types
7. compute p-values
8. rescale abundance estimates, to proportions of total, proportions of immune, cell counts

**Usage**

```
spatialdecon(
  norm,
  bg,
  X = NULL,
  raw = NULL,
  wts = NULL,
  resid_thresh = 3,
  lower_thresh = 0.5,
  align_genes = TRUE,
  is_pure_tumor = NULL,
  n_tumor_clusters = 10,
  cell_counts = NULL,
  cellmerges = NULL,
  maxit = 1000
)
```

**Arguments**

|      |  |
|------|--|
| norm | p-length expression vector or p * N expression matrix - the actual (linear-scale) data |
| bg   | Same dimension as norm: the background expected at each data point.                    |
| X    | Cell profile matrix. If NULL, the safeTME matrix is used.                              |

|                               |   |
|-------------------------------|---|
| <code>raw</code>              | Optional for using an error model to weight the data points. $p$ -length expression vector or $p * N$ expression matrix - the raw (linear-scale) data   |
| <code>wts</code>              | Optional, a matrix of weights.  |
| <code>resid_thresh</code>     | A scalar, sets a threshold on how extreme individual data points' values can be (in $\log_2$ units) before getting flagged as outliers and set to NA.   |
| <code>lower_thresh</code>     | A scalar. Before $\log_2$ -scale residuals are calculated, both observed and fitted values get thresholded up to this value. Prevents $\log_2$ -scale residuals from becoming extreme in points near zero.  |
| <code>align_genes</code>      | Logical. If TRUE, then <code>Y</code> , <code>X</code> , <code>bg</code> , and <code>wts</code> are row-aligned by shared genes.  |
| <code>is_pure_tumor</code>    | A logical vector denoting whether each AOI consists of pure tumor. If specified, then the algorithm will derive a tumor expression profile and merge it with the immune profiles matrix.  |
| <code>n_tumor_clusters</code> | Number of tumor-specific columns to merge into the cell profile matrix. Has an impact only when <code>is_pure_tumor</code> argument is used to indicate pure tumor AOIs. Takes this many clusters from the pure-tumor AOI data and gets the average expression profile in each cluster. Default 10. |
| <code>cell_counts</code>      | Number of cells estimated to be within each sample. If provided alongside <code>norm_factors</code> , then the algorithm will additionally output cell abundance estimates on the scale of cell counts.   |
| <code>cellmerges</code>       | A list object holding the mapping from beta's cell names to combined cell names. If left NULL, then defaults to a mapping of granular immune cell definitions to broader categories.  |
| <code>maxit</code>            | Maximum number of iterations. Default 1000.   |

## Value

a list:

- `beta`: matrix of cell abundance estimates, cells in rows and observations in columns
- `sigmas`: covariance matrices of each observation's beta estimates
- `p`: matrix of p-values for  $H_0: \beta == 0$
- `t`: matrix of t-statistics for  $H_0: \beta == 0$
- `se`: matrix of standard errors of beta values
- `prop_of_all`: rescaling of beta to sum to 1 in each observation
- `prop_of_nontumor`: rescaling of beta to sum to 1 in each observation, excluding tumor abundance estimates
- `cell.counts`: beta rescaled to estimate cell numbers, based on `prop_of_all` and nuclei count
- `beta.granular`: cell abundances prior to combining closely-related cell types
- `sigma.granular`: sigmas prior to combining closely-related cell types
- `cell.counts.granular`: `cell.counts` prior to combining closely-related cell types
- `resids`: a matrix of residuals from the model fit. ( $\log_2(\text{pmax}(y, \text{lower\_thresh})) - \log_2(\text{pmax}(xb, \text{lower\_thresh}))$ ).
- `X`: the cell profile matrix used in the decon fit.

**Examples**

```

data(mini_geomx_dataset)
data(safeTME)
data(safeTME.matches)
# estimate background:
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
# run basic decon:
res0 <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
  bg = mini_geomx_dataset$bg,
  X = safeTME
)
# run decon with bells and whistles:
res <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
  bg = mini_geomx_dataset$bg,
  X = safeTME,
  cellmerges = safeTME.matches,
  cell_counts = mini_geomx_dataset$annot$nuclei,
  is_pure_tumor = mini_geomx_dataset$annot$AOI.name == "Tumor"
)

```

---

TIL\_barplot

*Barplot of abundance estimates*


---

**Description**

Draw barplot of the "betas" from a decon fit

**Usage**

```
TIL_barplot(mat, draw_legend = FALSE, main = "", col = NULL, ...)
```

**Arguments**

|             |  |
|-------------|--|
| mat         | Matrix of cell proportions or abundances, in the same dimensions output by spatialdecon (cells in rows, observations in columns). User is free to re-order columns/observations in whatever order is best for display. |
| draw_legend | Logical. If TRUE, the function draws a legend in a new plot frame.   |
| main        | Title for barplot  |
| col         | Vector of colors for cell types. Defaults to pre-set colors for the safeTME cell types.  |
| ...         | Arguments passed to barplot()  |

**Value**

Draws a barplot.

**Examples**

```
data(mini_geomx_dataset)
data(safeTME)
# estimate background:
mini_geomx_dataset$bg <- derive_GeoMx_background(
  norm = mini_geomx_dataset$normalized,
  probepool = rep(1, nrow(mini_geomx_dataset$normalized)),
  negnames = "NegProbe"
)
# run basic decon:
res0 <- spatialdecon(
  norm = mini_geomx_dataset$normalized,
  bg = mini_geomx_dataset$bg,
  X = safeTME
)
# run barplot:
TIL_barplot(mat = res0$beta)
# run barplot and draw a color legend
TIL_barplot(mat = res0$beta, draw_legend = TRUE)
```

# Index

## \* datasets

- cellcols, [4](#)
- mean.resid.sd, [10](#)
- mini\_geomx\_dataset, [11](#)
- mini\_singleCell\_dataset, [12](#)
- nsclc, [12](#)
- safeTME, [26](#)
- safeTME.matches, [26](#)

- cellcols, [4](#)
- collapseCellTypes, [4](#)
- create\_profile\_matrix, [5](#)

- derive\_GeoMx\_background, [7](#)
- download\_profile\_matrix, [7](#)

- florets, [8](#)

- mean.resid.sd, [10](#)
- mergeTumorIntoX, [10](#)
- mini\_geomx\_dataset, [11](#)
- mini\_singleCell\_dataset, [12](#)

- nsclc, [12](#)

- reverseDecon, [13](#)
- runCollapseCellTypes, [14](#)
- runCollapseCellTypes,NanoStringGeoMxSet-method, [15](#)
- runErrorModel, [16](#)
- runMergeTumorIntoX, [16](#)
- runMergeTumorIntoX,NanoStringGeoMxSet-method, [17](#)
- runReverseDecon, [18](#)
- runReverseDecon,NanoStringGeoMxSet-method, [19](#)
- runspatialdecon, [20](#)
- runspatialdecon,NanoStringGeoMxSet-method, [21](#)
- runspatialdecon,Seurat-method, [24](#)

- safeTME, [26](#)
- safeTME.matches, [26](#)
- spatialdecon, [27](#)
- SpatialDecon-package, [2](#)
- TIL\_barplot, [29](#)