

Package: DenMark (via r-universe)

October 25, 2026

Type Package

Title Cell-Density Correlated Genes from Spatial Transcriptomics

Version 0.9.0

Description Quantifies the dependence between local cell density and gene expression in single-cell resolution spatial transcriptomics data, such as data from the Xenium and MERFISH platforms. Cell locations form a point pattern and gene counts are its marks. After aggregation on a regular grid, cell counts and summed expression follow Poisson models driven by two coregionalized latent Gaussian fields with Matern covariance, approximated by Hilbert space basis functions; the correlation between the two fields measures the density-expression dependence. Models are fitted with 'Stan' through 'cmdstanr'. Functions cover the tissue boundary (an alpha shape of the cells), gridding, model fitting, posterior summaries, latent-field decomposition, spatial correlation ranges, and gene scans that call density-correlated genes.

License MIT + file LICENSE

URL <https://stagill.github.io/DenMark/>,
<https://github.com/StaGill/DenMark>

BugReports <https://github.com/StaGill/DenMark/issues>

Depends R (>= 4.1.0)

Imports cmdstanr (>= 0.5.0), posterior, ggplot2 (>= 3.4.0), rlang,
stats, tools, utils

Suggests future, future.apply, jsonlite, Matrix, patchwork,
SeuratObject, sf, testthat (>= 3.0.0), knitr, rmarkdown

Remotes stan-dev/cmdstanr

Additional_repositories <https://stan-dev.r-universe.dev>

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Roxygen list(markdown = TRUE)

Config/roxygen2/version 8.1.1

Repository <https://stagill.r-universe.dev>

Date/Publication 2026-10-25 20:18:38 UTC

RemoteUrl <https://github.com/StaGill/DenMark>

RemoteRef HEAD

RemoteSha 99a1899a6b4b731ec18002580ccc6ca792c80f1e

Contents

buildgridpp	3
DenMark	4
denmark_alpha_shape	6
denmark_annotate	7
denmark_check_hsgp	8
denmark_decay	9
denmark_draws	10
denmark_fields	11
denmark_filter_genes	12
denmark_from_seurat	12
denmark_hsgp_indices	13
denmark_init	14
denmark_model	15
denmark_prepare	16
denmark_save	17
denmark_scan	18
denmark_simulate	19
denmark_summary	21
denmark_toy_data	22
DenMarkmain	22
grid_pts_centroid	24
matern32	24
plot_denmark_boundary	25
plot_denmark_decay	26
plot_denmark_fields	27
plot_denmark_grid	27
plot_denmark_scan	28
q2.5	29

Index	30
--------------	-----------

buildgridpp	<i>Discretize cell locations and expression on a regular grid</i>
-------------	---

Description

Each cell is assigned to exactly one grid cell (cells on an interior edge go to the upper or right cell; cells on the outer boundary are kept).

Usage

```
buildgridpp(grid_size, dataset, gene = NULL, bbox = NULL, boundary = NULL)
```

Arguments

grid_size	Number of grid cells per axis, or <code>c(nx, ny)</code> .
dataset	Data frame with columns <code>x</code> , <code>y</code> and the expression column(s), one row per cell.
gene	Expression column to use (default: the first non-coordinate column).
bbox	Optional bounding box <code>c(xmin, xmax, ymin, ymax)</code> ; the default is the data range, or the extent of boundary when one is given. Cells outside the box are dropped.
boundary	Optional tissue polygon from <code>denmark_alpha_shape()</code> . The grid then spans the tissue, and every grid cell gets its area inside the tissue (<code>area</code>) and a type: interior, boundary or exterior.

Value

A list. The vectors are ordered with `x` varying fastest, that is `grid_index = i + (j - 1) * nx`, identical to `as.vector(Ymat)`. The first three elements keep the order of the original DenMark function, so `res[[1]]`, `res[[2]]` and `res[[3]]` still work.

loc_mark_mat	<code>N x 2</code> matrix with the columns point (= <code>Y</code>) and mark (= <code>M</code>).
Ymat	<code>nx x ny</code> matrix of cell counts (rows = <code>x</code> , columns = <code>y</code>).
Mmat	<code>nx x ny</code> matrix of summed expression.
grid	Data frame with the columns <code>grid_index</code> , <code>i</code> , <code>j</code> , <code>x</code> , <code>y</code> (centroids), <code>Y</code> , <code>M</code> , <code>area</code> (area inside the tissue; the full grid cell area without a boundary) and <code>type</code> .
bbox, step, grid_area, grid_size, gene, boundary	Bounding box, grid steps <code>c(dx, dy)</code> , area of one full grid cell, <code>c(nx, ny)</code> , the gene name and the tissue polygon (NULL without one).

Examples

```
toy <- denmark_toy_data()
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
g <- buildgridpp(grid_size = 10, dataset = tumor, gene = "geneA")
dim(g$Ymat)
head(g$grid)
```

```
tissue <- denmark_alpha_shape(toy$x, toy$y)
gb <- buildgridpp(grid_size = 10, dataset = tumor, gene = "geneA",
                  boundary = tissue)
table(gb$grid$type)
```

DenMark

Fit DenMark to one gene

Description

Grids the data, builds the HSGP inputs, compiles the Stan model (once per session), sets valid initial values and samples with cmdstanr.

Usage

```
DenMark(
  dataset,
  gene = NULL,
  grid_size = 10,
  basis = 25,
  boundfactor = 1.5,
  bbox = NULL,
  chains = 4,
  parallel_chains = chains,
  iter_warmup = 1000,
  iter_sampling = 1000,
  adapt_delta = 0.95,
  max_treedepth = 10,
  seed = 2024,
  refresh = 400,
  init = NULL,
  rho0 = 0.3,
  stan_file = NULL,
  output_dir = NULL,
  keep_fit = TRUE,
  quiet = FALSE,
  L1 = NULL,
  L2 = NULL,
  boundary = NULL,
  min_area_frac = 0,
  ...
)

## S3 method for class 'denmark'
print(x, digits = 3, ...)
```

Arguments

dataset	Data frame with columns x, y and the expression column(s), one row per cell; expression values are raw counts.
gene	Name of the expression column (default: the first column other than x and y).
grid_size	Grid cells per axis, or c(nx, ny).
basis	HSGP basis functions per axis, or c(m1, m2).
boundfactor	HSGP boundary factor (≥ 1).
bbox	Optional region of interest c(xmin, xmax, ymin, ymax).
chains, parallel_chains, iter_warmup, iter_sampling, adapt_delta, max_treedepth, seed, refresh	Passed to the cmdstanr \$sample() method.
init	NULL (valid defaults from denmark_init()) or anything accepted by cmdstanr.
rho0	Starting value of rho.
stan_file	Optional path to a custom Stan file with the parameters of DenMark.stan; with a boundary it must declare vector[n] log_grid_area.
output_dir	Optional folder for the CmdStan CSV files (created if needed).
keep_fit	Keep the CmdStanMCMC object (set FALSE in large scans).
quiet	Suppress the sampler progress output.
L1, L2	Optional half-ranges of the region of interest; only checked against the data (see denmark_prepare()).
boundary	Optional tissue polygon from denmark_alpha_shape() ; grid cells outside the tissue are dropped and boundary grid cells use their clipped area (variable-area model).
min_area_frac	See denmark_prepare() .
...	For DenMark(), further arguments passed to \$sample(); ignored by print().
x	A denmark object.
digits	Number of significant digits in the printed summary table.

Details

The DenMark model (DenMark.stan) has two dependent latent fields, $\log \lambda_1(g) = \log |g| + \beta_0 + a_{11} w_1(g)$ and $\log \lambda_2(g) = \beta_1 + \log(Y(g) + 0.05) + a_{21} w_1(g) + a_{22} w_2(g)$; $\rho = \text{Astar}[2, 1]$ is the density-expression correlation.

A tissue section is rarely a rectangle. With boundary (an alpha shape of all cells of the section, from [denmark_alpha_shape\(\)](#)), the grid cells outside the tissue are dropped and every remaining grid cell enters with its area inside the tissue, through the variable-area model DenMark-vararea.stan (the same model with one log_grid_area per grid cell). Without boundary, the whole bounding box is gridded with a common grid area.

The print method reports rho, the sampler diagnostics and the posterior summary.

Value

An object of class `denmark`: a list with the cmdstanr fit (`fit`), `gene`, `stan_data`, `grid`, `info`, the sampler settings, the posterior summary (see `denmark_summary()`), the sampler diagnostics, the run time in minutes, the HSGP check `hsgp` (see `denmark_check_hsgp()`) and, for backward compatibility with `DenMarkmain()`, `rhat`, `Ymat` and `Mmat`. With `keep_fit = FALSE` the fit is dropped and the draws of the key parameters are kept in `draws_cache`.

See Also

`denmark_scan()` to fit many genes.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10, chains = 2,
               iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)

fit
```

denmark_alpha_shape	<i>Tissue boundary as an alpha shape of the cell locations</i>
---------------------	--

Description

The Delaunay triangulation of the cells is computed and every triangle whose circumradius is smaller than α is kept; the union of the kept triangles is the tissue region. This follows the boundary step of the real-data analyses: compute it from all cells of the section (not only the cell type analyzed), so that the density of one cell type is modeled over the whole tissue rather than over its own convex hull.

Usage

```
denmark_alpha_shape(
  x,
  y,
  alpha = NULL,
  alpha_mult = 3,
  keep = c("largest", "all"),
  fill_holes = TRUE
)
```

Arguments

x, y	Cell coordinates (all cells of the section).
alpha	Circumradius threshold; the default is alpha_mult times the median Delaunay edge length.
alpha_mult	Multiplier for the default alpha (3, as in the analyses).
keep	"largest" keeps the largest connected piece of tissue, "all" keeps every piece.
fill_holes	Fill holes inside the tissue.

Value

An sfc polygon (package sf) with the attribute "alpha".

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
attr(tissue, "alpha")
sf::st_area(tissue)
```

denmark_annotate	<i>Write DenMark scan results into a Seurat object</i>
------------------	--

Description

Adds, for every feature of the assay, the posterior mean of rho, its credible interval and the call from [denmark_scan\(\)](#) to the feature metadata (object[[assay]][[]]). Features that were not scanned get NA.

Usage

```
denmark_annotate(object, scan, assay = NULL, prefix = "denmark_")
```

Arguments

object	A Seurat object.
scan	Data frame returned by denmark_scan() .
assay	Assay whose feature metadata receive the results (default: the default assay).
prefix	Prefix of the new columns: rho, rho_lower, rho_upper and class.

Value

The Seurat object with the new feature metadata.

See Also[denmark_from_seurat\(\)](#)**Examples**

```

counts <- Matrix::Matrix(matrix(rpois(30, 2), 3, 10,
                                dimnames = list(c("geneA", "geneB", "geneC"), NULL)),
                           sparse = TRUE)
colnames(counts) <- paste0("cell", 1:10)
obj <- SeuratObject::CreateSeuratObject(counts = counts)
scan <- data.frame(gene = c("geneA", "geneC"), rho_mean = c(0.6, -0.6),
                   rho_lower = c(0.3, -0.8), rho_upper = c(0.8, -0.3),
                   class = c("PCG", "NCG"))
obj <- denmark_annotate(obj, scan)
obj[["RNA"]][[[]]]

```

denmark_check_hsgp

*Rule-of-thumb check of the HSGP approximation***Description**

Uses the one-dimensional Matern-3/2 recommendations of Riutort-Mayol et al. (2023, Statistics and Computing): with S the half-range of the domain, c the boundary factor and m the number of basis functions, the smallest length-scale that is well approximated is $\text{ell_min} = 3.42 * c * S / m$, and the boundary factor should satisfy $c \geq \max(1.2, 4.5 * \text{ell} / S)$. Applied per axis; a message is issued when the posterior median length-scale is below ell_min (increase basis) or c is too small. The rule targets an accurate approximation of the latent function itself and is conservative for estimating ρ ; treat it as guidance.

Usage

```
denmark_check_hsgp(obj, warn = TRUE)
```

Arguments

obj	A denmark object from DenMark() .
warn	Issue a message when the rule is not met.

Value

Data frame with one row per field and axis: field, axis, ell_median, ell_min, c_needed, basis_ok and boundary_ok.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10,
               chains = 2, iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)
denmark_check_hsgp(fit, warn = FALSE)
```

denmark_decay

*Spatial correlation decay of the two latent processes***Description**

Density process: $r_1(d) = M(d; e_{l11})$. Expression process: $r_2(d) = [a_{21}^2 M(d; e_{l11}) + a_{22}^2 M(d; e_{l12})] / (a_{21}^2 + a_{22}^2)$ with M the Matern-3/2 correlation. The practical range is the distance at which the correlation falls to threshold (default 0.05), computed per posterior draw.

Usage

```
denmark_decay(obj, distances = NULL, probs = c(0.025, 0.975), threshold = 0.05)
```

Arguments

obj	A denmark object from DenMark() .
distances	Distances at which to evaluate the curves (data units); default: 200 points from 0 to the maximum distance between grid centroids.
probs	Lower and upper probabilities of the posterior band.
threshold	Correlation level that defines the practical range.

Value

A list with curves (data frame: field, distance, posterior mean, lower and upper band), ranges (data frame: posterior mean, median and interval of the practical range of each field), range_draws (the practical ranges per draw) and threshold.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10,
               chains = 2, iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)
```

```
decay <- denmark_decay(fit)
decay$ranges
```

denmark_draws

Harmonized posterior draws

Description

One row per posterior draw with the columns beta0, beta1, ell1, ell2, rho, sigma1, sigma2 and the coregionalization coefficients $a_{11} = \sigma_1$, $a_{21} = \sigma_2 * \rho$ and $a_{22} = \sigma_2 * \sqrt{1 - \rho^2}$.

Usage

```
denmark_draws(obj)
```

Arguments

obj A denmark object from [DenMark\(\)](#).

Value

Data frame with one row per posterior draw.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10,
               chains = 2, iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)
draws <- denmark_draws(fit)
quantile(draws$rho, c(0.025, 0.5, 0.975))
```

denmark_fields	<i>Latent-field decomposition for every grid cell</i>
----------------	---

Description

Posterior means (and optional quantiles) of

- $\eta_1 = \beta_0 + a_{11} w_1$, the log cell intensity per unit area;
- $a_{11} w_1$, the density latent field;
- $\eta_2 = \beta_1 + a_{21} w_1 + a_{22} w_2$, the log expression per cell;
- $a_{21} w_1$, the density-shared part of expression;
- $a_{22} w_2$, the expression-specific part.

Usage

```
denmark_fields(obj, probs = NULL)
```

Arguments

obj	A denmark object fitted with <code>keep_fit = TRUE</code> .
probs	Optional probabilities; for each, the posterior quantiles of the five fields are added in columns such as <code>eta1_q2.5</code> .

Details

The full linear predictors are $\log_{\lambda 1} = \log|g| + \eta_1$ and $\log_{\lambda 2} = \log(Y + 0.05) + \eta_2$.

Value

Data frame with one row per grid cell: `grid_index`, `i`, `j`, `x`, `y`, `Y`, `M`, the five fields, `log_lambda1` and `log_lambda2`.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10,
               chains = 2, iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)
fields <- denmark_fields(fit)
head(fields)
```

denmark_filter_genes *Quality-control filter for genes*

Description

Keeps genes whose variance across cells is at or above the `var_quantile` quantile of the gene variances and that are detected (`count > 0`) in at least a fraction `min_nonzero` of the cells. With the defaults this is the filter used in the DenMark paper: the genes in the lowest 5 percent of variance are dropped, and at least 5 percent of the cells must have a non-zero count.

Usage

```
denmark_filter_genes(expr, var_quantile = 0.05, min_nonzero = 0.05)
```

Arguments

<code>expr</code>	Cells x genes matrix or data frame of counts (numeric columns only), with the gene names as column names.
<code>var_quantile</code>	Quantile of the gene variances below which genes are dropped.
<code>min_nonzero</code>	Minimum fraction of cells with a non-zero count.

Value

Character vector of the genes kept.

Examples

```
toy <- denmark_toy_data()
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
denmark_filter_genes(tumor[, c("geneA", "geneB", "geneC")])
```

denmark_from_seurat *Cell table from a Seurat object*

Description

Extracts what DenMark needs from a Seurat object with imaging-based spatial data, such as Xenium, MERSCOPE or CosMx data loaded with `LoadXenium()`, `LoadVizgen()` or `LoadNanostring()`: the cell centroids, a cell-type label and the raw counts, one row per cell. Compute the tissue boundary from all rows with `denmark_alpha_shape()`, then analyze the cells of one type.

Usage

```
denmark_from_seurat(object, celltype, genes = NULL, assay = NULL, fov = NULL)
```

Arguments

object	A Seurat object with a field of view holding cell centroids.
celltype	Name of the column of the cell metadata that holds the cell types, for example the result of a clustering or an annotation.
genes	Genes to include (default: all features of the assay). Each gene becomes a dense column, so select genes for whole-transcriptome panels.
assay	Assay holding the raw counts (default: the default assay).
fov	Field of view holding the cell centroids (default: the first).

Value

Data frame with the columns x, y (in the units of the field of view, micrometers for Xenium), celltype and one count column per gene, with the cell names as row names.

See Also

[denmark_annotate\(\)](#) to write the results of [denmark_scan\(\)](#) back into the object.

Examples

```
# a small Seurat object built from the toy data
toy <- denmark_toy_data()
counts <- Matrix::Matrix(t(as.matrix(toy[, c("geneA", "geneB", "geneC")])),
  sparse = TRUE)
colnames(counts) <- paste0("cell", seq_len(nrow(toy)))
obj <- SeuratObject::CreateSeuratObject(counts = counts)
obj$celltype <- toy$celltype
obj[["fov"]] <- SeuratObject::CreateFOV(
  SeuratObject::CreateCentroids(data.frame(x = toy$x, y = toy$y, cell = colnames(counts))),
  type = "centroids", assay = "RNA")

cells <- denmark_from_seurat(obj, celltype = "celltype")
head(cells)
```

denmark_hsgp_indices *HSGP basis-function index matrix*

Description

All pairs of basis-function indices, 1..m1 x 1..m2, with the first index varying fastest, as used by the Stan models.

Usage

```
denmark_hsgp_indices(basis)
```

Arguments

`basis` Number of basis functions per axis, or `c(m1, m2)`.

Value

Integer matrix with `m1 * m2` rows and two columns, without dimnames.

Examples

```
denmark_hsgp_indices(3)
dim(denmark_hsgp_indices(c(10, 8)))
```

<code>denmark_init</code>	<i>Initial values for DenMark</i>
---------------------------	-----------------------------------

Description

Returns a function (as expected by `cmdstanr`) giving valid starting values. The Cholesky factor of the correlation matrix must be lower triangular with unit-norm rows: $A_{star} = \begin{bmatrix} 1 & 0 \\ \rho_0 & \sqrt{1 - \rho_0^2} \end{bmatrix}$. The length-scales start at the prior mean (`mrange`) and the intercepts at their empirical values (for β_0 , the number of cells per unit of tissue area), which shortens warm-up.

Usage

```
denmark_init(stan_data, rho0 = 0.3)
```

Arguments

`stan_data` Stan data list, as in `denmark_prepare()`\$`stan_data`.
`rho0` Starting value of ρ , in $(-1, 1)$.

Value

A function without arguments that returns a list of initial values.

Examples

```
toy <- denmark_toy_data()
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
prep <- denmark_prepare(tumor, gene = "geneA", grid_size = 10, basis = 10)
init <- denmark_init(prepare$stan_data)
str(init())
```

denmark_model	<i>Compile (once) and return the DenMark cmdstanr model</i>
---------------	---

Description

DenMark.stan, or DenMark-vararea.stan with vararea = TRUE, is looked up in `getOption("denmark.stan_dir")` and then in the stan folder of the installed package. The compiled model is cached for the R session. The executables of the packaged models are written to `tools::R_user_dir("DenMark", which = "cache")` (one subfolder per package version, because the library may be read-only), so that later sessions reuse them; any other Stan file is compiled next to itself.

Usage

```
denmark_model(
  stan_file = NULL,
  vararea = FALSE,
  force_recompile = FALSE,
  quiet = TRUE
)
```

Arguments

stan_file	Optional path to a Stan file, compiled instead of the packaged model.
vararea	Use the variable-area model DenMark-vararea.stan, the same model with one area offset <code>log_grid_area</code> per grid cell (used with a tissue boundary).
force_recompile	Recompile even if a compiled model is available.
quiet	Suppress the compiler output.

Value

A cmdstanr `CmdStanModel` object.

Examples

```
mod <- denmark_model()
mod$exe_file()
```

denmark_prepare

Build the Stan data list for DenMark

Description

The grid is centered at the origin, so that the region of interest becomes $[-L1, L1] \times [-L2, L2]$; the grid centroids are the HSGP input coordinates and the HSGP domain is $\text{boundfactor} * c(L1, L2)$. With a tissue boundary, the grid cells outside the tissue are removed first and the region of interest is that of the retained grid.

Usage

```
denmark_prepare(
  dataset,
  gene = NULL,
  grid_size = 10,
  basis = 25,
  boundfactor = 1.5,
  bbox = NULL,
  L1 = NULL,
  L2 = NULL,
  boundary = NULL,
  min_area_frac = 0
)
```

Arguments

dataset	Data frame with columns x, y and the expression column(s), one row per cell.
gene	Expression column (default: the first non-coordinate column).
grid_size	Grid cells per axis, or c(nx, ny).
basis	Number of HSGP basis functions per axis, or c(m1, m2).
boundfactor	HSGP boundary factor c (≥ 1 ; 1.5 is a good default).
bbox	Optional region of interest c(xmin, xmax, ymin, ymax); the default is the data range, or the extent of boundary when one is given.
L1, L2	Optional half-ranges. Only used to warn if they disagree with the data (the old interface required them; they are now computed).
boundary	Optional tissue polygon from denmark_alpha_shape() . Grid cells outside the tissue are removed, and each remaining grid cell enters with its own area inside the tissue: log_grid_area becomes a vector and the variable-area model DenMark-vararea.stan is used.
min_area_frac	Drop boundary grid cells whose area inside the tissue is below this fraction of a full grid cell (default 0: keep all).

Value

A list with the elements

stan_data	Data list for the Stan models: n, y1 (cell counts), y2 (summed expression), d, mstar, coords (centered grid centroids), log_grid_area (one value, or one per grid cell with a boundary), mrange, indices, L and is_centered_PHI.
grid	The grid from <code>buildgridpp()</code> (without the exterior grid cells when a boundary is given) with the centered coordinates cx and cy.
info	Gene, grid geometry, HSGP settings, number of cells, the count matrices Ymat and Mmat, and for the tissue boundary vararea, boundary, n_grid_total, n_interior and n_boundary.

Examples

```
toy <- denmark_toy_data()
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
prep <- denmark_prepare(tumor, gene = "geneA", grid_size = 10, basis = 10)
str(prepare$stan_data[c("n", "mstar", "L", "log_grid_area", "mrangle")])

tissue <- denmark_alpha_shape(toy$x, toy$y)
prep_b <- denmark_prepare(tumor, gene = "geneA", grid_size = 10, basis = 10,
                          boundary = tissue)
prep_b$info[c("n_grid_total", "n_interior", "n_boundary")]
summary(exp(prepare_b$stan_data$log_grid_area))
```

denmark_save	<i>Save and load a DenMark fit</i>
--------------	------------------------------------

Description

cmdstanr keeps posterior draws in CSV files (by default in a temporary folder that disappears when R closes). `denmark_save()` reads everything into memory first, so the saved object can be reloaded and post-processed in a later session.

Usage

```
denmark_save(obj, file)

denmark_load(file)
```

Arguments

obj	A denmark object from <code>DenMark()</code> .
file	Path of the .rds file.

Value

denmark_save() returns file invisibly; denmark_load() returns the denmark object.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10,
               chains = 2, iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)
f <- tempfile(fileext = ".rds")
denmark_save(fit, f)
fit2 <- denmark_load(f)
denmark_summary(fit2)
```

denmark_scan

Fit DenMark gene by gene and call density-correlated genes (DCG)

Description

A gene is a positively density-correlated gene (PCG) when the cred_level credible interval of rho lies above 0, a negatively density-correlated gene (NCG) when it lies below 0, and "not DCG" otherwise.

Usage

```
denmark_scan(
  data,
  genes = NULL,
  save_dir = NULL,
  cred_level = 0.95,
  quiet = TRUE,
  verbose = TRUE,
  workers = 1,
  ...
)
```

Arguments

data	Data frame with columns x, y and one column per gene (cells in rows).
genes	Genes to scan (default: all non-coordinate columns).
save_dir	Optional folder; one .rds file per gene is written, and genes already present are loaded instead of refitted (resumable scans).
cred_level	Credible level used to call DCGs.

quiet	Suppress the sampler output, passed to <code>DenMark()</code> .
verbose	Report one message per gene with rho and its credible interval.
workers	Number of genes fitted at the same time, each in its own R session (needs the <code>future.apply</code> package and an installed <code>DenMark</code>). Every gene keeps its seed, so the results equal those of the sequential scan. Each fit also runs its chains in parallel (<code>parallel_chains</code> , by default <code>chains</code>), so a scan uses up to <code>workers * chains</code> cores. For a cluster, <code>system.file("slurm", package = "DenMark")</code> holds a template that fits one gene per SLURM array task.
...	Further arguments passed to <code>DenMark()</code> (<code>grid_size</code> , <code>basis</code> , ...).

Value

Data frame with one row per gene, sorted by decreasing posterior mean of rho: `gene`, `rho_mean`, `rho_median`, `rho_lower`, `rho_upper`, `prob_positive`, `total_counts`, `max_rhat`, `min_ess_bulk`, `divergences`, `time_min`, `error` (NA unless the fit failed), `class` (PCG, NCG or not DCG) and `converged` (maximum R-hat below 1.01 and no divergent transitions). The credible level is stored in `attr("cred_level")`.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
scan <- denmark_scan(tumor, genes = c("geneA", "geneC"), boundary = tissue,
  grid_size = 10,
  basis = 10, chains = 2, iter_warmup = 300,
  iter_sampling = 300, seed = 2024, verbose = FALSE)
scan[, c("gene", "rho_mean", "rho_lower", "rho_upper", "class")]
```

denmark_simulate	<i>Simulate a marked point pattern from the DenMark data-generating process</i>
------------------	---

Description

Cells of the analyzed type follow an inhomogeneous Poisson process with $\log \lambda_1(s) = \beta_0 + \alpha_1 w_1(s)$, and each cell's count for gene k is Poisson with $\log \mu_k(s) = \beta_1 + \sigma_2 * (\rho_k w_1(s) + \sqrt{1 - \rho_k^2} w_2(s))$.

Usage

```
denmark_simulate(
  rho = c(geneA = 0.6, geneB = 0, geneC = -0.6),
  width = 6,
  height = 4,
  n_cells = 5000,
  ell1 = 1,
```

```

ell2 = 1,
a11 = 1,
sigma2 = 0.8,
beta1 = 0.5,
seed = 1,
tissue = c("rectangle", "irregular"),
n_other = 0,
celltype = "Target"
)

```

Arguments

rho	Named vector of true correlations, one gene per element.
width, height	Size of the window [0, width] x [0, height].
n_cells	Number of cells of the analyzed type: the thinned point pattern is subsampled to n_cells cells when it has more.
ell1, ell2	Length-scales of the density and expression fields.
a11	Standard deviation of the log cell intensity.
sigma2	Standard deviation of the log expression.
beta1	Intercept of the log expression.
seed	Random seed, set with <code>set.seed()</code> ; this resets the global random number stream.
tissue	"rectangle" (cells fill the window) or "irregular" (cells lie in a lobed tissue section).
n_other	Number of cells of other types; together with the analyzed cells they define the tissue boundary.
celltype	Label of the analyzed cell type (used when n_other > 0; the other cells are labeled "Other").

Details

The fields are smooth, zero-mean, unit-variance Gaussian fields with approximately Matern-3/2 covariance, drawn by random Fourier features. The fields are standardized, and each $w2k$ is orthogonalized against $w1$, on a regular lattice over the tissue (that is, over space rather than over the cells), so that the spatial correlation of the two fields equals ρ_k , which is what ρ measures in the model.

With `tissue = "irregular"`, the cells lie in a lobed tissue section inside the window, so that the analysis needs a tissue boundary (see [denmark_alpha_shape\(\)](#)). The cells of other types are drawn from the candidate locations that the thinning rejected, so they are densest where the analyzed cells are sparse; their gene counts are Poisson with mean 0.3.

Value

Data frame with the columns `x`, `y`, `celltype` (when `n_other > 0`) and one count column per gene; the true values are stored in `attr("truth")`, including the true tissue outline for `tissue = "irregular"`.

Examples

```
sim <- denmark_simulate(rho = c(up = 0.5, down = -0.5), n_cells = 500)
head(sim)
attr(sim, "truth")$rho

sec <- denmark_simulate(n_cells = 1000, tissue = "irregular", n_other = 800,
                        celltype = "Tumor")
table(sec$celltype)
```

denmark_summary	<i>Posterior summary of the key DenMark parameters</i>
-----------------	--

Description

The key parameters are β_0 , β_1 , $e_{11}[1]$, $e_{11}[2]$, $A_{star}[2,1]$ ($= \rho$), $\sigma[1]$ and $\sigma[2]$.

Usage

```
denmark_summary(
  obj,
  variables = NULL,
  probs = c(0.025, 0.25, 0.5, 0.75, 0.975)
)
```

Arguments

obj	A denmark object from <code>DenMark()</code> .
variables	Optional override of the variables to summarize.
probs	Probabilities of the posterior quantiles to report.

Value

Data frame with the columns variable, mean, sd, one column per quantile (q2.5, q25, ...), rhat, ess_bulk, ess_tail and a descriptive label.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10,
               chains = 2, iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)
denmark_summary(fit)
```

denmark_toy_data	<i>Toy data with a known answer</i>
------------------	-------------------------------------

Description

A simulated tissue section: 3,449 cells of the analyzed type ("Tumor") and 4,000 cells of other types ("Other") in an irregular, lobed tissue. The Tumor cells carry three genes whose density-expression correlations are known: geneA ($\rho = 0.6$), geneB ($\rho = 0$) and geneC ($\rho = -0.6$). The other cells only help delineate the tissue: compute the boundary with `denmark_alpha_shape()` from all cells, then analyze the Tumor cells. The examples, the tests and the vignette use these data, so every result can be checked against the truth.

Usage

```
denmark_toy_data()
```

Details

The file `inst/extdata/denmark_toy.csv.gz` was generated by `data-raw/make_toy_data.R` with `denmark_simulate()`, using `rho = c(geneA = 0.6, geneB = 0, geneC = -0.6)`, `width = 6`, `height = 4`, `n_cells = 5000`, `seed = 1`, `tissue = "irregular"`, `n_other = 4000`, `celltype = "Tumor"` and the defaults for the other arguments (`ell1 = 1`, `ell2 = 1`, `a11 = 1`, `sigma2 = 0.8`, `beta1 = 0.5`); the coordinates were rounded to six decimals. The Python package reads a byte copy of the same file with `denmark.datasets.load_toy()`.

Value

Data frame with one row per cell and the columns `x` and `y` (cell locations in the window $[0, 6] \times [0, 4]$), `celltype` ("Tumor" or "Other") and `geneA`, `geneB` and `geneC` (counts).

Examples

```
toy <- denmark_toy_data()
head(toy)
table(toy$celltype)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
```

DenMarkmain

Backward-compatible wrapper of the original DenMark interface

Description

Keeps scripts written for the original interface running. L1 and L2 are no longer needed (they are computed from the data) and are only checked.

Usage

```
DenMarkmain(
  grid_size,
  dataset,
  basis,
  boundfactor,
  L1 = NULL,
  L2 = NULL,
  denmark_version = "M2",
  ...
)
```

Arguments

<code>grid_size</code>	Grid cells per axis, or <code>c(nx, ny)</code> .
<code>dataset</code>	Data frame with columns <code>x</code> , <code>y</code> and one expression column.
<code>basis</code>	HSGP basis functions per axis, or <code>c(m1, m2)</code> .
<code>boundfactor</code>	HSGP boundary factor (≥ 1).
<code>L1, L2</code>	Optional half-ranges, only checked against the data.
<code>denmark_version</code>	"M2" (the default) fits the DenMark model; any other value is taken as the path to a Stan file (as in the earlier tutorial) and passed to <code>DenMark()</code> as <code>stan_file</code> .
<code>...</code>	Further arguments passed to <code>DenMark()</code> .

Value

An object of class `denmark`, see `DenMark()`.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- toy[toy$celltype == "Tumor", c("x", "y", "geneA")]
fit <- DenMarkmain(grid_size = 10, dataset = tumor,
  basis = 10, boundfactor = 1.5, boundary = tissue,
  chains = 2, iter_warmup = 300, iter_sampling = 300,
  seed = 2024, quiet = TRUE)
fit$rhat
```

grid_pts_centroid	<i>Grid centroids, grid area and number of grids</i>
-------------------	--

Description

Grid centroids, grid area and number of grids

Usage

```
grid_pts_centroid(grid_size, dataset, bbox = NULL)
```

Arguments

grid_size	Number of grid cells per axis, or c(nx, ny).
dataset	Data frame with the columns x and y (if these are missing, the first two columns are used).
bbox	Optional bounding box c(xmin, xmax, ymin, ymax); the default is the data range.

Value

list(grid_points, grid_area, N, step, bbox); the first three elements keep the order of the original function. grid_points are the true centroids of the grid cells (x varying fastest).

Examples

```
toy <- denmark_toy_data()
geom <- grid_pts_centroid(grid_size = 10, dataset = toy)
c(grid_area = geom$grid_area, n_grids = geom$N)
```

matern32	<i>Matern (nu = 3/2) covariance</i>
----------	-------------------------------------

Description

Matern covariance with smoothness 3/2, in the same parameterization as the Stan models: $C(d) = \sigma^2 (1 + \sqrt{3} d / \rho)$. The argument name rho (the length-scale ell) is kept for backward compatibility.

Usage

```
matern32(d, sigma = 1, rho)
```

Arguments

d	Distances.
sigma	Marginal standard deviation.
rho	Length-scale ell; not the density-expression correlation.

Value

Numeric vector of covariances, one per element of d.

Examples

```
matern32(c(0, 0.5, 1, 2), rho = 1)
```

plot_denmark_boundary *Tissue boundary and grid classification*

Description

Left: all cells (gray), the cells of the analyzed type (colored) and the alpha-shape tissue boundary. Right (when grid is given): the grid cells colored by type (interior, boundary, exterior) with the boundary outline.

Usage

```
plot_denmark_boundary(
  cells,
  boundary,
  grid = NULL,
  highlight = NULL,
  xlab = "x",
  ylab = "y",
  point_size = 0.15
)
```

Arguments

cells	Data frame with the columns x and y (all cells).
boundary	Polygon from denmark_alpha_shape() .
grid	Optional output of <code>buildgridpp(..., boundary = boundary)</code> .
highlight	Optional logical vector, one element per row of cells: TRUE for the cells of the analyzed type.
xlab, ylab	Axis labels.
point_size	Size of the cell points.

Value

A ggplot object; with grid, the two panels combined with patchwork, or a list of two ggplot objects when patchwork is not installed.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
g <- buildgridpp(grid_size = 10, dataset = tumor, gene = "geneA",
                 boundary = tissue)
plot_denmark_boundary(toy, tissue, grid = g,
                     highlight = toy$celltype == "Tumor")
```

plot_denmark_decay	<i>Spatial correlation decay curves with posterior bands</i>
--------------------	--

Description

Posterior mean and band of the spatial correlation of the density and expression fields, with the threshold that defines the practical range as a dashed line.

Usage

```
plot_denmark_decay(obj, decay = NULL, xlab = "Distance")
```

Arguments

obj	A denmark object from DenMark() .
decay	Optional output of denmark_decay() ; computed from obj when NULL.
xlab	Label of the distance axis.

Value

A ggplot object.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
              grid_size = 10, basis = 10,
              chains = 2, iter_warmup = 300, iter_sampling = 300,
              seed = 2024, quiet = TRUE)
plot_denmark_decay(fit)
```

plot_denmark_fields	<i>Latent-field decomposition maps (posterior means)</i>
---------------------	--

Description

Row 1 (cell density): $\beta_0 + a_{11} w_1$ and $a_{11} w_1$, on a shared color scale. Row 2 (expression): $\beta_1 + a_{21} w_1 + a_{22} w_2$, $a_{21} w_1$ and $a_{22} w_2$, on a shared color scale.

Usage

```
plot_denmark_fields(obj, fields = NULL, xlab = "x", ylab = "y")
```

Arguments

obj	A denmark object fitted with <code>keep_fit = TRUE</code> .
fields	Optional output of <code>denmark_fields()</code> ; computed from obj when NULL.
xlab, ylab	Axis labels.

Value

A patchwork figure, or a list of five ggplot objects when patchwork is not installed.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
fit <- DenMark(tumor, gene = "geneA", boundary = tissue,
               grid_size = 10, basis = 10,
               chains = 2, iter_warmup = 300, iter_sampling = 300,
               seed = 2024, quiet = TRUE)
plot_denmark_fields(fit)
```

plot_denmark_grid	<i>Gridded cell counts (Y) and summed expression (M)</i>
-------------------	--

Description

Tile maps of the cell count and the summed expression of each grid cell. With a tissue boundary, only the grid cells inside the tissue are drawn, together with the tissue outline.

Usage

```
plot_denmark_grid(obj, xlab = "x", ylab = "y")
```

Arguments

obj A denmark fit, or the result of `buildgridpp()`.
 xlab, ylab Axis labels.

Value

The two panels combined with patchwork, or a list of two ggplot objects when patchwork is not installed.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
plot_denmark_grid(buildgridpp(grid_size = 10, dataset = tumor, gene = "geneA",
                              boundary = tissue))
```

plot_denmark_scan	<i>Forest plot of rho across genes from denmark_scan()</i>
-------------------	--

Description

Posterior mean and credible interval of rho for each gene, colored by the DCG call.

Usage

```
plot_denmark_scan(scan, top = NULL)
```

Arguments

scan Data frame returned by `denmark_scan()`.
 top Optional number of genes to show, those with the largest absolute posterior mean of rho.

Value

A ggplot object.

Examples

```
toy <- denmark_toy_data()
tissue <- denmark_alpha_shape(toy$x, toy$y)
tumor <- subset(toy, celltype == "Tumor", select = -celltype)
scan <- denmark_scan(tumor, genes = c("geneA", "geneC"), boundary = tissue,
                    grid_size = 10,
                    basis = 10, chains = 2, iter_warmup = 300,
                    iter_sampling = 300, seed = 2024, verbose = FALSE)
plot_denmark_scan(scan)
```

q2.5

*Posterior quantile helpers***Description**

Shorthands for the 2.5th, 25th, 50th, 75th and 97.5th percentiles of a vector of posterior draws, kept for compatibility with earlier scripts.

Usage

q2.5(x)

q25(x)

q50(x)

q75(x)

q97.5(x)

Arguments

x Numeric vector, for example the posterior draws of one parameter.

Value

A single unnamed number: the sample quantile of x from stats::quantile() with its default method.

Examples

```
x <- 0:100
c(q2.5(x), q25(x), q50(x), q75(x), q97.5(x))
```

Index

buildgridpp, 3
buildgridpp(), 17, 28

DenMark, 4
DenMark(), 8–10, 17, 19, 21, 23, 26
denmark_alpha_shape, 6
denmark_alpha_shape(), 3, 5, 12, 16, 20, 22, 25
denmark_annotate, 7
denmark_annotate(), 13
denmark_check_hsgp, 8
denmark_check_hsgp(), 6
denmark_decay, 9
denmark_decay(), 26
denmark_draws, 10
denmark_fields, 11
denmark_fields(), 27
denmark_filter_genes, 12
denmark_from_seurat, 12
denmark_from_seurat(), 8
denmark_hsgp_indices, 13
denmark_init, 14
denmark_init(), 5
denmark_load(denmark_save), 17
denmark_model, 15
denmark_prepare, 16
denmark_prepare(), 5
denmark_save, 17
denmark_scan, 18
denmark_scan(), 6, 7, 13, 28
denmark_simulate, 19
denmark_simulate(), 22
denmark_summary, 21
denmark_summary(), 6
denmark_toy_data, 22
DenMarkmain, 22
DenMarkmain(), 6

grid_pts_centroid, 24

matern32, 24

plot_denmark_boundary, 25
plot_denmark_decay, 26
plot_denmark_fields, 27
plot_denmark_grid, 27
plot_denmark_scan, 28
print.denmark(DenMark), 4

q2.5, 29
q25 (q2.5), 29
q50 (q2.5), 29
q75 (q2.5), 29
q97.5 (q2.5), 29