Package 'fastfocal'

September 23, 2025

Type Package

Title Fast Multiscale Raster Extraction and Moving Window Analysis with FFT

Version 0.1.3

Date 2025-09-09

Description Provides fast moving-window (``focal'') and buffer-based extraction for raster data using the 'terra' package. Automatically selects between a 'C++' backend (via 'terra') and a Fast Fourier Transform (FFT) backend depending on problem size. The FFT backend supports sum and mean, while other statistics (e.g., median, min, max, standard deviation) are handled by the 'terra' backend. Supports multiple kernel types (e.g., circle, rectangle, gaussian), with NA handling consistent with 'terra' via 'na.rm' and 'na.policy'. Operates on 'SpatRaster' objects and returns results with the same geometry.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports terra, graphics, grDevices, stats

Suggests testthat (>= 3.0.0), knitr, rmarkdown, dplyr, withr, spelling

VignetteBuilder knitr

URL https://hoyiwan.github.io/fastfocal/, https://github.com/hoyiwan/fastfocal, https://doi.org/10.5281/zenodo.17074691

BugReports https://github.com/hoyiwan/fastfocal/issues

Config/testthat/edition 3

RoxygenNote 7.3.2

Language en-US

NeedsCompilation no

Author Ho Yi Wan [aut, cre] (ORCID: <https://orcid.org/0000-0002-2146-8257>)

Maintainer Ho Yi Wan <hoyiwan@gmail.com>

Repository CRAN

Date/Publication 2025-09-23 10:50:02 UTC

Contents

`fastextract` *Fast raster extraction at points (buffered)*

Description

Extracts summary statistics from a SpatRaster at point locations, optionally using buffered extraction with custom kernel windows.

Usage

```
fastextract(x, y, d = 0, w = "circle", fun = "mean", na.rm = TRUE)
```

Arguments

`x`	SpatRaster. Input raster (single- or multi-layer).
`y`	SpatVector. Points or polygons.
`d`	numeric or numeric vector. Buffer radius/radii in map units.
`w`	character. Window type for the buffer kernel when `d > 0` (currently passed through to **terra**; e.g., "circle", "rectangle").
`fun`	character or function. Summary function: "mean", "sum", "min", "max", "sd", or "median"; or a user function.
`na.rm`	logical. Whether to remove NAs when computing summaries.

Details

- If `d > 0`, a buffer of radius `d` (map units) is created around each point and the summary is computed over raster cells intersecting the buffer.
- If `d == 0`, values are taken at the point locations (no buffering).
- If `y` is a polygon layer, the summary is computed over polygon areas.

Value

A data.frame of extracted values. When `d` has multiple values, rows are stacked by scale with a `scale_m` column indicating the radius.

Examples

```
r <- terra::rast(nrows = 10, ncols = 10, xmin = 0, xmax = 100, ymin = 0, ymax = 100)
terra::values(r) <- seq_len(terra::ncell(r))

pts <- terra::vect(
  matrix(c(10, 10,
           50, 50), ncol = 2, byrow = TRUE),
  type = "points",
  crs  = terra::crs(r)
)

# Mean over a 20-unit circular neighborhood around each point
res <- fastextract(r, pts, d = 20, w = "circle", fun = "mean")
head(res)
```

`fastfocal` *Fast focal smoothing with FFT auto-switching*

Description

Applies a focal operation to a `SpatRaster` using either a 'C++' backend (via **terra**) or an 'FFT' backend. Window types include rectangle, circle, gaussian, pareto, idw, exponential, triangular, cosine, logistic, cauchy, quartic, epanechnikov, or you may pass a numeric matrix as the kernel.

Usage

```
fastfocal(
  x,
  d,
  w = "circle",
  fun = "mean",
  engine = "auto",
  na.rm = TRUE,
  na.policy = c("omit", "all"),
  pad = c("none", "auto"),
  ...
)
```

Arguments

`x`	SpatRaster. Input raster (1+ layers).
`d`	numeric. Radius/size in map units (ignored if `w` is a matrix).
`w`	character or numeric matrix. Window type, or a custom kernel matrix.
`fun`	character. One of "mean","sum","min","max","sd","median".
`engine`	character. "auto" (default), "cpp", or "fft".
`na.rm`	logical. Remove NAs before applying the summary function.

na.policy	character. "omit" (default) leaves NA centers as NA; "all" fills centers when neighbors exist (FFT path respects this; C++ path emulates center handling after the call).
pad	character. "none" or "auto" (pad to next 5-smooth sizes for FFT).
...	Extra args to terra::focal() for the 'C++' path.

Details

The 'FFT' backend uses masked convolution with proper NA semantics and can pad to "5-smooth" sizes for stable speed. With engine = "auto", the function chooses between 'C++' and 'FFT' based on a simple window-size heuristic.

Value

terra::SpatRaster with the same geometry as x.

Examples

```
set.seed(1)
r <- terra::rast(nrows = 12, ncols = 12, xmin = 0, xmax = 12, ymin = 0, ymax = 12)
terra::values(r) <- stats::runif(terra::ncell(r))

# Mean with a small circular window (d is in map units; here res = 1)
m_circ <- fastfocal(r, d = 2, w = "circle", fun = "mean")

# Same idea using a custom 3x3 box kernel (uniform mean)
k3 <- matrix(1, 3, 3)
m_box <- fastfocal(r, w = k3, fun = "mean")

# Tiny numeric summaries (keeps examples fast & quiet for CRAN)
as.numeric(terra::global(m_circ, "mean", na.rm = TRUE))
as.numeric(terra::global(m_box,  "mean", na.rm = TRUE))
```

---

fastfocal_weights *Generate weight matrix for focal operations using map units*

Description

Builds an unnormalized (or normalized) kernel from map units. Circle uses a center-distance rule (include if center <= d). **Gaussian interprets d as sigma in map units and truncates at 3 sigma**, matching terra::focalMat(..., type = "Gauss").

Usage

```
fastfocal_weights(x, d, w = "circle", normalize = TRUE, plot = FALSE)
```

Arguments

x	SpatRaster (used for resolution; assumes square pixels).
d	numeric. Radius in map units for most kernels; **sigma** in map units for "gaussian"/"Gauss".
w	character. One of: "rectangle","circle","circular","gaussian","Gauss","pareto","idw", "exponential","triangular","cosine","logistic","cauchy","quartic","epanechnikov".
normalize	logical. If TRUE (default), scale weights to sum to 1.
plot	logical. If TRUE, plots the kernel.

Value

numeric matrix of weights.

Examples

```
# Small raster (resolution = 1 map unit)
r <- terra::rast(nrows = 5, ncols = 5, xmin = 0, xmax = 5, ymin = 0, ymax = 5)

# Circle: d is a radius in map units -> here cell_radius = 2 -> 5x5 kernel
Kc <- fastfocal_weights(r, d = 2, w = "circle", normalize = TRUE)
dim(Kc)            # 5 x 5
round(sum(Kc), 6)  # ~1

# Gaussian: d is sigma in map units, truncated at 3 sigmas
Kg <- fastfocal_weights(r, d = 1, w = "gaussian", normalize = TRUE)
dim(Kg)            # 7 x 7 (since 2*ceil(3*sigma) + 1)
round(sum(Kg), 6)  # ~1

# \donttest{
# Quick visualization (kept out of CRAN's main run)
fastfocal_weights(r, d = 2, w = "circle", normalize = TRUE, plot = TRUE)
# }
```

Index