# Package 'rainfarmr'

April 9, 2019

**Title** Stochastic Precipitation Downscaling with the RainFARM Method

**Version** 0.1

**URL** https://github.com/jhardenberg/rainfarmr

**BugReports** https://github.com/jhardenberg/rainfarmr

**Description**

An implementation of the RainFARM (Rainfall Filtered Autoregressive Model) stochastic precipitation downscaling method (Rebora et al. (2006) <doi:10.1175/JHM517.1>). Adapted for climate downscaling according to D'Onofrio et al. (2018) <doi:10.1175/JHM-D-13-096.1> and for complex topography as in Terzago et al. (2018) <doi:10.5194/nhess-18-2825-2018>. The RainFARM method is based on the extrapolation to small scales of the Fourier spectrum of a large-scale precipitation field, using a fixed logarithmic slope and random phases at small scales, followed by a nonlinear transformation of the resulting linearly correlated stochastic field. RainFARM allows to generate ensembles of spatially downscaled precipitation fields which conserve precipitation at large scales and whose statistical properties are consistent with the small-scale statistics of observed precipitation, based only on knowledge of the large-scale precipitation field.

**Depends** R (>= 3.1.0)

**License** Apache License 2.0

**LazyData** true

**RoxygenNote** 6.1.1

**Encoding** UTF-8

**NeedsCompilation** no

**Author** Jost von Hardenberg [aut, cre, cph]
(<https://orcid.org/0000-0002-5312-8070>)

**Maintainer** Jost von Hardenberg <j.vonhardenberg@isac.cnr.it>

**Repository** CRAN

**Date/Publication** 2019-04-09 12:10:03 UTC

# R topics documented:



---

`agg` *Aggregation using box-averaging*

---

## Description

Aggregates an input field `z` to an array at coarser resolutions `c(nas,nas)` by box averaging. The input array can contain missing values.

## Usage

```
agg(z, nas)
```

## Arguments

| | |
|---|---|
| `z` | matrix containing the fine field to aggregate. |
| `nas` | target dimension for the output field. |

## Value

The coarse field obtained by box averaging.

## Author(s)

Jost von Hardenberg, `<j.vonhardenberg@isac.cnr.it>`

## Examples

```
z <- rnorm(64 * 64)
dim(z) <- c(64, 64)
za <- agg(z, 4)
dim(za)
# [1] 4 4
```

---

`downscale` *Downscale a precipitation field*

---

## Description

Downscales an input precipitation matrix using a metagaussian spectral field `f` previously generated with `initmetagauss()`. The target resolution is defined by the dimensions of `f`. An optional weights array can be specified.

## Usage

```
downscale(r, f, weights = 1, fglob = FALSE, fsmooth = TRUE)
```

## Arguments

| | |
|---|---|
| `r` | matrix of precipitation data to downscale. |
| `f` | matrix containing a complex spectrum generated by the `initmetagauss()` function. |
| `weights` | matrix with climatological weights generated with the `rfweights()` function. |
| `fglob` | logical to conserve global average over domain. |
| `fsmooth` | logical to use smoothing for conservation. If neither option is set precipitation is conserved over each coarse box. |

## Value

The downscaled field, with the same dimensions as `f`.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
# Make some sample synthetic rainfall data
r <- exp(rnorm(4 * 4))
dim(r) <- c(4, 4)
r
#           [,1]      [,2]      [,3]      [,4]
# [1,] 1.8459816 1.8536550 2.1600665 1.3102116
# [2,] 1.3851011 1.4647348 0.2708219 0.4571810
```

```
# [3,] 0.2492451 0.8227134 0.4790567 1.9320403
# [4,] 0.5985922 3.3065678 2.1282795 0.6849944
# Create help field `f` with logarithmic slope 1.7
# with `dim(f) = c(8 * 4 ,8 * 4)`
f <- initmetagauss(1.7, 8 * 4)
rd <- downscale(r, f, fsmooth=FALSE)
# Verify that downscaled data maintained original box averages
agg(rd, 4)
#           [,1]      [,2]      [,3]      [,4]
# [1,] 1.8459816 1.8536550 2.1600665 1.3102116
# [2,] 1.3851011 1.4647348 0.2708219 0.4571810
# [3,] 0.2492451 0.8227134 0.4790567 1.9320403
# [4,] 0.5985922 3.3065678 2.1282795 0.6849944
```

---

`fft2d` *Compute spatial Fourier power spectrum*

---

## Description

The Fourier power spectrum of the input field is computed and averaged over shells (bins with width 1) of the modulus of the wavenumber.

## Usage

```
fft2d(z)
```

## Arguments

| | |
|---|---|
| `z` | matrix or array of input data with spatial dimensions `c(N, N)`. If `z` is an array the power spectra are averaged over the third dimension. The first two dimensions must be the same (the input fields must be square). |

## Value

Spectral power (average square of absolute value of spectral amplitudes) for wavenumbers `k=1:(N/2)`.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
# Make a synthetic rainfall field with prescribed logarithmic spectral slope
f = initmetagauss(1.7, 64)
r = metagauss(f)
# Check slope of the resulting field
fx <- fft2d(r)
fitslope(fx)
# 1.640373
```

## fitslope *Compute logarithmic slope of a spatial power spectrum*

### Description

Uses a linear fit to derive the log-log slope of a Fourier power spectrum.

### Usage

```
fitslope(fx, kmin = 1, kmax = length(fx))
```

### Arguments

| | |
|---|---|
| fx | vector containing input power spectrum starting from k=1. |
| kmin | minimum wavenumber for logarithmic fit range. |
| kmax | maximum wavenumber for logarithmic fit range. |

### Value

The spatial spectral slope minus one. The slope is returned as the logarithmic slope of k*|A(k)|^2 where |A(k)|^2 are the squared spectral amplitudes provided in input..

### Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

### Examples

```
# Make a synthetic rainfall field with prescribed logarithmic spectral slope
f = initmetagauss(1.7, 64)
r = metagauss(f)
# Check spectral slope of the resulting field
fx <- fft2d(r)
print(fitslope(fx))
# 1.640373
```

## gaussianize *Gaussianize field using rank ordering*

### Description

The amplitudes of the input field are rank ordered and substituted with the corresponding rank-ordered samples from a Normal distribution.

## Usage

```
gaussianize(z)
```

## Arguments

| | |
|---|---|
| z | matrix containing the input field to be Gaussianized. |

## Value

The Gaussianized field with the same dimensions as the input field.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
# Make some sample synthetic rainfall data
r <- exp(rnorm(64 * 64))
dim(r) <- c(64, 64)
# Gaussianize and check standard deviation
g <- gaussianize(r)
sd(g)
# [1] 1
```

---

# initmetagauss *Generate the spectral amplitudes for a metagaussian field*

---

## Description

A Fourier spectrum with prescribed logarithimic slope and zero phases is constructed.

## Usage

```
initmetagauss(sx, ns)
```

## Arguments

| | |
|---|---|
| sx | spectral slope for the output field. The convention is that this is the slope of k*\|A(k)\|^2. |
| ns | size of the output field. |

## Value

Output complex field in Fourier space with specified spectral slope, with dimensions c(ns, ns).

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
# Make a synthetic rainfall field with prescribed logarithmic spectral slope
f = initmetagauss(1.7, 64)
r = metagauss(f)
# Check slope of the resulting field
fx <- fft2d(r)
fitslope(fx)
# 1.640373
```

---

`interpola` *Interpolate field using nearest neighbors*

---

## Description

Interpolates a square input matrix to a finer resolution `ns` using nearest neighbours.

## Usage

```
interpola(z, ns)
```

## Arguments

| | |
|---|---|
| `z` | matrix containing the input field at coarse resolution. |
| `ns` | the target size. |

## Value

The resulting fine-scale field with dimensions `c(ns, ns)`.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
za <- rnorm(4 * 4)
dim(za) <- c(4, 4)
z <- interpola(za, 16)
dim(z)
# [1] 16 16
```

`lon_lat_fine` *Linear interpolation of longitude and latitude vectors to higher resolution*

## Description

Longitude and latitude vectors are interpolated higher resolution increasing dimensions by a factor `nf`. The grid spacings of the first two and of the last two elements are used to extrapolate at the boundaries.

## Usage

```
lon_lat_fine(lon, lat, nf)
```

## Arguments

| | |
|---|---|
| `lon` | vector of longitudes. |
| `lat` | vector of latitudes. |
| `nf` | factor by which to increase resolution. |

## Value

List with elements `lon` and `lat` with resolution increased by factor `nf`.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
lon <- 5:9
lat <- 43:47
nf <- 4
grid <- lon_lat_fine(lon, lat, nf)
grid$lon
# [1] 4.625 4.875 5.125 5.375 5.625 5.875 6.125 6.375 6.625 6.875 7.125 7.375
# [13] 7.625 7.875 8.125 8.375 8.625 8.875 9.125 9.375
grid$lat
# [1] 42.625 42.875 43.125 43.375 43.625 43.875 44.125 44.375 44.625 44.875
# [11] 45.125 45.375 45.625 45.875 46.125 46.375 46.625 46.875 47.125 47.375
```

mergespec *Spectral merging of a coarse field and of a fine field at a given wavenumber*

## Description

The input fields are transformed to Fourier space, their Fourier spectra are merged at wavenumber kmax after adapting the variance of the fine field and an inverse Fourier transform is performed.

## Usage

```
mergespec(ra, r, kmax = 0)
```

## Arguments

| | |
|---|---|
| ra | matrix containing a coarse field of size c(nas, nas). |
| r | matrix containing a field at higher resolution. |
| kmax | wavenumber to use for merging (default nas/2). |

## Value

The merged field, in physical space.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
# Make a coarse field with power-law Fourier spectrum
fa <- initmetagauss(1.7, 8)
ra <- metagauss(fa)
# Make a fine power-law Fourier spectrum
f <- initmetagauss(1.7, 32)
r <- metagauss(f)
# Merge the two fields in spectral space
rm <- mergespec(ra, r, kmax = 4)
# Check spectral slope of the resulting field
fx <- fft2d(rm)
fitslope(fx)
# 1.678554
```

## metagauss *Generate a metagaussian field*

### Description

Random Fourier phases are added to the input spectrum `f` and an inverse FFT transform to real space is performed.

### Usage

```
metagauss(f)
```

### Arguments

| | |
|---|---|
| `f` | matrix with complex spectral amplitudes generated with the `initmetagauss()` function. |

### Value

A metagaussian field with random Fourier phases.

### Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

### Examples

```
f = initmetagauss(1.7, 64)
z = metagauss(f)
sd(z)
# [1] 1.000122
```

## rainfarm *Perform RainFARM downscaling*

### Description

The input array is downscaled to finer spatial resolution using the RainFARM stochastic precipitation downscaling method. Orographic correction weights can be applied as described in Terzago et al. (2018) doi: 10.5194/nhess1828252018. Precipitation can be conserved globally (`fglob`), using convolution (`fsmooth`) or over the original coarse-scale boxes.

### Usage

```
rainfarm(r, slope, nf, weights = 1, fglob = FALSE, fsmooth = TRUE,
  verbose = FALSE)
```

## Arguments

| | |
|---|---|
| r | matrix or array with large-scale field to downscale. Can be a three-dimensional array with multiple frames at different times. Spatial downscaling is performed separately for each element of the third dimension of r. |
| slope | spatial spectral slope. |
| nf | refinement factor for spatial downscaling. |
| weights | matrix with weights for orographic downscaling generated by the rfweights() function. |
| fglob | logical to conserve global average over domain. |
| fsmooth | logical to use smoothing for conservation. If neither fsmooth or fglob is set precipitation is conserved over each coarse pixel of the input field. |
| verbose | logical to provide some progress report. |

## Value

The downscaled array.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## References

Terzago, S. et al. (2018). NHESS 18(11), 2825–2840 doi: 10.5194/nhess1828252018; D'Onofrio et al. (2014). J of Hydrometeorology 15, 830-843 doi: 10.1175/JHMD13096.1; Rebora et. al. (2006), JHM 7, 724 doi: 10.1175/JHM517.1.

## Examples

```
# Make some sample synthetic rainfall data
r <- exp(rnorm(4 * 4 * 10))
dim(r) <- c(4, 4, 10)
r[ , , 1]
#           [,1]      [,2]      [,3]      [,4]
# [1,] 1.8459816 1.8536550 2.1600665 1.3102116
# [2,] 1.3851011 1.4647348 0.2708219 0.4571810
# [3,] 0.2492451 0.8227134 0.4790567 1.9320403
# [4,] 0.5985922 3.3065678 2.1282795 0.6849944
# Downscale with spectral slope=1.7 to size 32x32
rd <- rainfarm(r, 1.7, 8, fsmooth=FALSE)
# Verify that downscaled data maintained original box averages
agg(rd[ , , 1], 4)
#           [,1]      [,2]      [,3]      [,4]
# [1,] 1.8459816 1.8536550 2.1600665 1.3102116
# [2,] 1.3851011 1.4647348 0.2708219 0.4571810
# [3,] 0.2492451 0.8227134 0.4790567 1.9320403
# [4,] 0.5985922 3.3065678 2.1282795 0.6849944
```

`remapcon` *Conservative remapping*

## Description

Implements conservative remapping, weighting with the overlap area between pixels.

## Usage

```
remapcon(x, y, z, xo, yo)
```

## Arguments

| | |
|---|---|
| `x` | vector of input longitudes |
| `y` | vector of input latitudes |
| `z` | matrix of input data |
| `xo` | vector of target longitudes |
| `yo` | vector of target latitudes |

## Value

A remapped matrix of dimensions `c(length(xo), length(yo))`

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
z <- 1:(31*51)
dim(z) <- c(31, 51)
x <- seq(4, 10, 0.2)
y <- seq(30, 40, 0.2)
xo <- seq(5, 6, 0.5)
yo <- seq(35, 37, 0.5)
zo <- remapcon(x, y, z, xo, yo)
zo
#       [,1]  [,2]  [,3]   [,4]   [,5]
# [1,] 781.0 858.5 936.0 1013.5 1091.0
# [2,] 783.5 861.0 938.5 1016.0 1093.5
# [3,] 786.0 863.5 941.0 1018.5 1096.0
```

`rfweights` *Derive weights from a fine-scale precipitation climatology*

## Description

Weights for downscaling are computed interpolating a fine-scale climatology to the target grid and dividing it by an averaged version of itself. A suitable climatology could be represented for example by a fine-scale precipitation climatology from a high-resolution regional climate model (see e.g. Terzago et al. (2018) doi: 10.5194/nhess1828252018), a local high-resolution gridded climatology from observations, or a reconstruction such as those which can be downloaded from the WORLDCLIM or CHELSA websites. The latter data could be converted to NetCDF format using for example the GDAL tools.

## Usage

```
rfweights(z, lon, lat, lonc, latc, nf, fsmooth = TRUE)
```

## Arguments

| | |
|---|---|
| `z` | matrix with a spatial field of fine-scale precipitation climatology. |
| `lon` | vector of longitudes of the high-resolution climatology |
| `lat` | vector of latitudes of the high-resolution climatology |
| `lonc` | vector of longitudes of the coarse field to downscale |
| `latc` | vector of latitudes of the coarse field to downscale |
| `nf` | refinement factor for downscaling (the coarse resolution is increased by this factor). The number of longitudes and latitudes is expected to be equal. |
| `fsmooth` | logical to compute weights against a smooth average. If false box averaging is used. |

## Value

The matrix of weights with dimensions `c(lonc*nf, latc*nf)`

## Author(s)

Jost von Hardenberg, `<j.vonhardenberg@isac.cnr.it>`

## References

Terzago, S. et al. (2018). NHESS 18(11), 2825–2840 doi: 10.5194/nhess1828252018.

## Examples

```
# Make synthetic fine-scale precipitation climatology
z <- exp(metagauss(initmetagauss(1.7, 64)))
# Specify lon and lat of the input
lon <- seq(10,17.875,0.125)
lat <- seq(40,47.875,0.125)
# Specify lon and lat of the coarse field and the downscaling factor
lonc <- seq(12,15.5,0.5)
latc <- seq(42,45.5,0.5)
nf <- 4
ww <- rfweights(z, lon, lat, lonc, latc, nf)
```

---

# smoothconv *Smoothening using convolution with a circular kernel*

---

## Description

The input field is convolved with a circular kernel with equal weights. Takes into account missing values.

## Usage

```
smoothconv(z, nas)
```

## Arguments

| | |
|---|---|
| z | matrix with the input field to smoothen, with dimensions c(ns, ns) |
| nas | the smoothing kernel uses a radius (ns/nas)/2 |

## Value

The smoothened field.

## Author(s)

Jost von Hardenberg, <j.vonhardenberg@isac.cnr.it>

## Examples

```
z <- rnorm(64 * 64)
dim(z) <- c(64, 64)
zs <- smoothconv(z, 8)
sd(zs)
# [1] 0.07910996
```

# Index