

```
@Manual{casnet-package,  
  title = {casnet: A toolbox for studying Complex Adaptive Systems and NETworks},  
  author = {Fred Hasselman},  
  year = {2017},  
  note = {R package version 0.1.3},  
  url = {https://github.com/FredHasselmann/casnet},  
}
```

# Package ‘casnet’

July 17, 2017

**Type** Package

**Title** A Toolbox for Studying Complex Adaptive Systems and NETworks

**Version** 0.1.3

**Maintainer** Fred Hasselman <f.hasselmann@pwo.ru.nl>

**Description** A collection of analytic tools for studying signals recorded from complex adaptive systems or networks:

- Recurrence Quantification Analyses (CrossRQA, Categorical RQA, Chromatic RQA, Anisotropic RQA).
- Fluctuation Analyses (DFA varieties, PSD slope, SDA, Multifractal DFA, Wavelet Singularity Spectrum).
- Coupling Analyses (Cross Conformal Mapping, Detection of Coupling Direction, CRQA).
- Network based time series analyses (Recurrence Networks, Multifractal Spectrum Networks, Multiplex Networks).

**YEAR** 2017

**License** GPL-3 + file LICENSE

**Depends** R (>= 2.10)

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 6.1.1

**Roxygen** list(markdown = TRUE)

**Imports** dplyr,  
fractal,  
ggplot2,  
gridExtra,  
ifultools,  
igraph,  
lattice,  
latticeExtra,  
Matrix,  
nonlinearTseries,  
plyr,  
pracma,  
proxy,  
scales,  
xts,  
zoo,

tidyr,  
rio,  
gtable,  
reshape2,  
sapa,  
signal,  
lubridate,  
magrittr,  
tseries,  
parallel,  
DescTools,  
pROC,  
purrr,  
rpart,  
rlang ( $\geq 0.1.2$ ),  
infotheo,  
methods,  
RColorBrewer,  
tibble,  
broom,  
callr,  
readr,  
paletteer,  
ggraph,  
boot,  
invctr,  
cowplot,  
raster,  
mice,  
imputeTS

**Suggests** knitr,  
rmarkdown,  
graphics,  
grDevices,  
stats,  
utils,  
devtools,  
tidyverse,  
testthat,  
roxygen2

**VignetteBuilder** knitr

**Language** en-US

**URL** <https://github.com/FredHasselman/casnet>

**BugReports** <https://github.com/FredHasselman/casnet/issues>

R topics documented:

add_alpha . . . . .	5
as.numeric_character . . . . .	5
as.numeric_discrete . . . . .	6

# R topics documented:

3

as.numeric_factor . . . . .	6
bandReplace . . . . .	7
crqa_cl . . . . .	8
crqa_diagProfile . . . . .	11
crqa_parameters . . . . .	12
crqa_radius . . . . .	13
crqa_rp . . . . .	15
dc_ccp . . . . .	16
dc_d . . . . .	17
dc_f . . . . .	18
dc_f2 . . . . .	19
dc_win . . . . .	19
di2bi . . . . .	21
di2we . . . . .	22
dist_hamming . . . . .	22
elascer . . . . .	23
est_emDim . . . . .	24
est_emLag . . . . .	25
factor_obs_exp . . . . .	26
fd_allan . . . . .	26
fd_boxcount2D . . . . .	27
fd_dfa . . . . .	29
fd_mfdfa . . . . .	30
fd_psd . . . . .	31
fd_RR . . . . .	32
fd_sda . . . . .	33
fd_sev . . . . .	34
flight_Cauchy . . . . .	35
flight_LevyPareto . . . . .	36
flight_Rayleigh . . . . .	37
getColours . . . . .	38
get_os . . . . .	38
gg_plotHolder . . . . .	39
gg_theme . . . . .	40
growth_ac . . . . .	40
growth_ac_cond . . . . .	41
layout_as_spiral . . . . .	42
make_spiral_focus . . . . .	43
make_spiral_graph . . . . .	45
mif . . . . .	46
mif_interlayer . . . . .	47
mi_mat . . . . .	48
noise_fBm . . . . .	49
noise_fGn . . . . .	49
noise_powerlaw . . . . .	50
plotDC_ccp . . . . .	50
plotDC_lvl . . . . .	51
plotDC_res . . . . .	52
plotFD_loglog . . . . .	53
plotNET_BA . . . . .	54
plotNET_groupColour . . . . .	54
plotNET_groupWeight . . . . .	55

plotNET_prep . . . . .	56
plotNET_SW . . . . .	57
plotRED_acf . . . . .	58
plotRED_mif . . . . .	59
plotSUR_hist . . . . .	59
plotTS_multi . . . . .	60
repmat . . . . .	61
rn . . . . .	62
RNG . . . . .	63
rn_plot . . . . .	64
rn_recSpec . . . . .	65
rn_scaleoGram . . . . .	66
rp . . . . .	66
rp_copy_attributes . . . . .	68
rp_lineDist . . . . .	68
rp_nzdiags . . . . .	69
rp_plot . . . . .	70
rp_size . . . . .	71
sa2fd_dfa . . . . .	72
sa2fd_psd . . . . .	73
sa2fd_sda . . . . .	74
set_command_line_rp . . . . .	74
ssg_gwf2long . . . . .	75
ssg_winning . . . . .	75
SWtestE . . . . .	76
ts_center . . . . .	76
ts_changeindex . . . . .	77
ts_checkfix . . . . .	77
ts_detrend . . . . .	79
ts_diff . . . . .	79
ts_discrete . . . . .	81
ts_duration . . . . .	82
ts_embed . . . . .	83
ts_integrate . . . . .	84
ts_levels . . . . .	84
ts_peaks . . . . .	86
ts_permtest_block . . . . .	86
ts_permtest_transmat . . . . .	87
ts_rasterize . . . . .	88
ts_sd . . . . .	89
ts_slice . . . . .	90
ts_standardise . . . . .	90
ts_sumorder . . . . .	91
ts_symbolic . . . . .	92
ts_transmat . . . . .	93
ts_trimfill . . . . .	94
ts_windower . . . . .	95

*add\_alpha*

5

---

<code>add_alpha</code>	<i>Add transparency to a colour</i>
------------------------	-------------------------------------

---

### Description

Add transparency to a colour

### Usage

```
add_alpha(col, alpha = 1)
```

### Arguments

<code>col</code>	A colour name, hexadecimal string or positive integer <code>i</code> , such that <code>palette()[i]</code>
<code>alpha</code>	Alpha transparency value

### Value

An rgb colour with transparency

---

<code>as.numeric_character</code>	<i>Character vector to named numeric vector</i>
-----------------------------------	---

---

### Description

Converts a character vector to a named numeric vector, with the character elements as names.

### Usage

```
as.numeric_character(x, sortUnique = FALSE, keepNA = FALSE)
```

### Arguments

<code>x</code>	A character vector
<code>sortUnique</code>	Should the unique character values be sorted? (default = FALSE)
<code>keepNA</code>	Keep NA values (TRUE), or remove them (default = FALSE)

### Value

A named numeric vector

### Examples

```
f <- letters
as.numeric_character(f)
```

---

as.numeric_discrete	<i>Discrete (factor or character) to numeric vector</i>
---------------------	---

---

**Description**

Converts a factor with numeric levels, or, a character vector with numeric values to a numeric vector using `as.numeric_factor`, or, `as.numeric_character` respectively. If an unnamed numeric vector is passed, it will be returned as a named numeric vector.

**Usage**

```
as.numeric_discrete(x, keepNA = FALSE)
```

**Arguments**

x	A factor with levels that are numeric, or, a character vector representing numbers.
keepNA	Keep NA values (TRUE), or remove them (default = FALSE)

**Value**

A numeric vector with factor levels / numeric character values as names.

**Examples**

```
f <- factor(round(runif(10,0,9)))
as.numeric_factor(f)

# Add NAs
f <- factor(c(round(runif(9,0,9)),NA))
as.numeric_factor(f)
as.numeric_factor(f, keepNA = TRUE)
```

---

as.numeric_factor	<i>Numeric factor to numeric vector</i>
-------------------	---

---

**Description**

Converts a factor with numeric levels to a numeric vector, using the values of the levels.

**Usage**

```
as.numeric_factor(x, keepNA = FALSE)
```

*bandReplace*

7

### Arguments

**x** A factor based on numeric values.  
**keepNA** Keep NA values (TRUE), or remove them (default = FALSE)

### Value

A numeric vector with factor levels as names.

### Examples

```
f <- factor(round(runif(10,0,9)))
as.numeric_factor(f)

# Add NAs
f <- factor(c(round(runif(9,0,9)),NA))
as.numeric_factor(f)
as.numeric_factor(f, keepNA = TRUE)
```

---

<code>bandReplace</code>	<i>Replace matrix diagonals</i>
--------------------------	---------------------------------

---

### Description

Sets a band of matrix diagonals to any given value

### Usage

```
bandReplace(mat, lower, upper, value = NA, silent = TRUE)
```

### Arguments

**mat** A Matrix  
**lower** Lower diagonal to be included in the band (should be  $\leq 0$ )  
**upper** Upper diagonal to be included in the band (should be  $\geq 0$ )  
**value** A single value to replace all values in the selected band (default = NA)  
**silent** Operate in silence, only (some) warnings will be shown (default = TRUE)

### Value

A matrix in which the values in the selected diagonals have been replaced

### Author(s)

Fred Hasselman

### See Also

Other Distance matrix operations (recurrence plot): [di2bi](#), [di2we](#), [dist\\_hamming](#), [rp\\_lineDist](#), [rp\\_nzdiags](#), [rp\\_plot](#), [rp\\_size](#), [rp](#)

**Examples**

```
# Create a 10 by 10 matrix
library(Matrix)
m <- Matrix(rnorm(10),10,10)

bandReplace(m,-1,1,0) # Replace diagonal and adjacent bands with 0 (Theiler window of 1)
```

crqa\_cl

*Fast (C)RQA (command line crp)***Description**

This function will run the **commandline Recurrence Plots** executable provided by Norbert Marwan.

**Usage**

```
crqa_cl(y1, y2 = NULL, emDim = 1, emLag = 1, emRad = NA,
        DLmin = 2, VLmin = 2, theiler = 0, win = min(length(y1),
        ifelse(is.null(y2), (length(y1) + 1), length(y2))), na.rm = TRUE),
        step = win, JRP = FALSE, distNorm = c("EUCLIDEAN", "MAX", "MIN",
        "OP")[[1]], standardise = c("none", "mean.sd", "median.mad")[1],
        returnMeasures = TRUE, returnRPvector = FALSE,
        returnLineDist = FALSE, doPlot = c("noplot", "rp", "distmat")[[1]],
        path_to_rp = getOption("casnet.path_to_rp"), saveOut = FALSE,
        path_out = NULL, file_ID = NULL, silent = TRUE,
        surrogateTest = FALSE, targetValue = 0.05, useParallel = FALSE,
        ...)
```

**Arguments**

y1	Time series 1
y2	Time series 2 for Cross Recurrence Analysis (default = NULL)
emDim	Embedding dimensions (default = 1)
emLag	Embedding lag (default = 1)
emRad	Radius on distance matrix (default = 1)
DLmin	Minimum length of diagonal structure to be considered a line (default = 2)
VLmin	Minimum length of vertical structure to be considered a line (default = 2)
theiler	Theiler window (default = 0)
win	Window to calculate the (C)RQA (default = minimum of length of y1 or y2)
step	Stepsize for sliding windows (default = size of win, so no sliding window)
JRP	Whether to calculate a Joint Recurrence Plot (default = FALSE)
distNorm	One of "EUCLIDEAN" (default), "MAX", "MIN", or "OP" for an Order Pattern recurrence matrix
standardise	Standardise data: "none" (default), "mean.sd", or "median.mad"
returnMeasures	Return the (C)RQA measures? (default = TRUE)
returnRPvector	Return the recurrent points in a dataframe? (default = FALSE)



returnLineDist	Return the distribution of diagonal and horizontal line length distances (default = FALSE)
doPlot	Produce a plot of the recurrence matrix by calling <code>rp_plot()</code> , values can be "rp" (the thresholded recurrence matrix), "distmat" (the unthresholded recurrence matrix) or "noplot" (default = "noplot")
path_to_rp	Path to the command line executable (default = path set during installation, use <code>getOption("casnet.path_to_rp")</code> to see)
saveOut	Save the output to files? If TRUE and path_out = NA, the current working directory will be used (default = FALSE)
path_out	Path to save output if saveOut = TRUE (default = NULL)
file_ID	A file ID which will be a prefix to the filename if saveOut = TRUE (default = NULL, an integer will be added to the file name to ensure unique files)
silent	Do not display any messages (default = TRUE)
surrogateTest	Perform surrogate tests. If TRUE, will run surrogate tests using default settings for a two-sided test of $H_0$ : <i>The data generating process is a rescaled linear Gaussian process</i> at $\alpha = .05$ (arguments ns = 39, fft = TRUE, amplitude = TRUE)
targetValue	A value passed to <code>crqa_radius(..., type="fixed", targetMeasure="RR")</code> if <code>is.na(emRad)==TRUE</code> . This is useful for windowed analysis, it will estimate a new radius for each window.
useParallel	Speed up calculations by using the parallel processing options provided by parallel to assign a separate process/core for each window in windowed (C)RQA analysis using <code>purrr::map2()</code> to assign data and <code>parallel::detectCores()</code> with <code>logical = TRUE</code> to decide on the available cores (default = FALSE)
...	Additional parameters (currently not used)

## Details

The rp executable is installed when the function is called for the first time and is renamed to rp, from a platform specific filename downloaded from <http://tocsy.pik-potsdam.de/commandline-rp.php> or extracted from an archive located in the directory: `... \casnet\commandline_rp\`. The file is copied to the directory: `... \casnet\exec\`. The latter location is stored as an option and can be read by calling `getOption("casnet.path_to_rp")`.

## Value

A list object containing 1-3 elements, depending on arguments requesting output.

1. rqa\_measures - A list of the (C)RQA measures returned if `returnMeasures = TRUE`:

- RR = 'Recurrence rate'
- DET = 'Determinism'
- DET\_RR = 'Ratio DET/RR'
- LAM = 'Laminarity'
- LAM\_DET = 'Ratio LAM/DET'
- L\_max = 'maximal diagonal line length'
- L\_mean = 'mean diagonal line length'
- L\_entr = 'Entropy of diagonal line length distribution'
- DIV = 'Divergence (1/L\_max)'
- V\_max = 'maximal vertical line length'

- TT = 'Trapping time'
  - V\_entr = 'Entropy of vertical line length distribution'
  - T1 = 'Recurrence times 1st type'
  - T2 = 'Recurrence times 2nd type'
  - W\_max = 'Max interval length'
  - W\_mean = 'Mean of interval lengths'
  - W\_entr = 'Entropy of interval length distribution'
  - W\_prob = 'Probability of interval'
  - F\_min = 'F min'
2. rqa\_rpvector - The radius thresholded distance matrix (recurrence matrix), which can be visualised as a recurrence plot by calling `rp_plot()`. If a sliding window analysis is conducted this will be a list of matrices and could potentially grow too large to handle. It is recommended you save the output to disk by setting `saveOut = TRUE`.
  3. rqa\_diagdist - The distribution of diagonal line lengths

### Troubleshooting

Some notes on resolving errors with rp. The script will first try to download the correct executable, if that fails it will try to extract the file from a .zip archive in `... \casnet \commandline_rp \crp_cl.zip`. If that fails, the copy will have failed. It should be relatively easy to get `crqa_cl()` working using custom settings:

- *Copy failed* - Every time the function `crqa_cl()` is called it will check whether a log file `rp_instal_log.txt` is present in the `... \casnet \exec \` directory. If you delete the `rp_instal_log.txt` file, and call the function, another attempt will be made to download a copy of the executable.
- *Copy still fails and/or no permission to copy* - If you cannot access the directory `... \casnet \commandline_rp \`, download the appropriate executable from the **commandline Recurrence Plots** page and copy to a directory you do have the rights to: *execute* commands, *write* and *read* files. Make sure you rename the file to `rp` (`rp.exe` on Windows OS). Then, either pass the path to `rp` as the argument `path_to_rp` in the `crqa_cl(..., path_to_rp = "YOUR_PATH")` function call, or, as a more permanent solution, set the `path_to_rp` option by calling `options(casnet.path_to_rp="YOUR_PATH")`.
- *Error in execution of rp* - This can have a variety of causes, the `rp` executable is called using `system2()` and makes use of the `normalizePath()` function with argument `mustWork = FALSE`. Problems caused by specific OS, machine, or, locale problems (e.g. the winslash can be reported as an **issue on Github**). One execution error that occurs when the OS is not recognised properly can be resolved by checking `getOption("casnet.rp_prefix")`. On Windows OS this should return an empty character vector, on Linux or macOS it should return `"/"`. You can manually set the correct prefix by calling `options(casnet.rp_prefix="CORRECT OS PREFIX")` and fill in the prefix that is correct for your OS

### Note

The platform specific `rp` command line executables were created by Norbert Marwan and obtained under a Creative Commons License from the website of the Potsdam Institute for Climate Impact Research at <http://tocsy.pik-potsdam.de/>.

The full copyright statement on the website is as follows:

(C) 2004-2017 SOME RIGHTS RESERVED

University of Potsdam, Interdisciplinary Center for Dynamics of Complex Systems, Germany

Potsdam Institute for Climate Impact Research, Transdisciplinary Concepts and Methods, Germany

This work is licensed under a [Creative Commons Attribution-NonCommercial-NoDerivs 2.0 Germany License](#).

More information about recurrence analysis can be found on the [Recurrence Plot](#) website.

### See Also

Other Recurrence Quantification Analysis: [crqa\\_parameters](#), [crqa\\_radius](#), [crqa\\_rp\\_measures](#), [crqa\\_rp](#)

---

crqa_diagProfile	<i>Diagonal Recurrence Profile</i>
------------------	------------------------------------

---

### Description

Diagonal Recurrence Profile

### Usage

```
crqa_diagProfile(RM, diagWin = NULL, xname = "X-axis",
  yname = "Y-axis", DLmin = 2, VLmin = 2, HLmin = 2,
  DLmax = length(Matrix::diag(RM)) - 1,
  VLmax = length(Matrix::diag(RM)) - 1,
  HLmax = length(Matrix::diag(RM)) - 1, doShuffle = FALSE, y1 = NA,
  y2 = NA, Nshuffle = 19, AUTO = NULL, chromatic = FALSE,
  matrices = FALSE, doPlot = TRUE)
```

### Arguments

RM	A binary recurrence matrix
diagWin	Window around the line of synchrony
xname	Label for x-axis
yname	Label for y-axis
DLmin	Minimal diagonal line length (default = 2)
VLmin	Minimal vertical line length (default = 2)
HLmin	Minimal horizontal line length (default = 2)
DLmax	Maximal diagonal line length (default = length of diagonal -1)
VLmax	Maximal vertical line length (default = length of diagonal -1)
HLmax	Maximal horizontal line length (default = length of diagonal -1)
doShuffle	Should a shuffled baseline be calculated (default = FALSE)
y1	The original y1 time series
y2	The original y2 time series
Nshuffle	How many shuffled versions to make up the baseline? The default is 19, which is the minimum for a one-sided surrogate test.
AUTO	Auto-recurrence? (default = FALSE)
chromatic	Force chromatic RQA? (default = FALSE)
matrices	Return matrices? (default = FALSE)
doPlot	Plot

**Value**

A plot and/or the data for the plot

---

crqa_parameters	<i>Find optimal (C)RQA parameters</i>
-----------------	---------------------------------------

---

**Description**

A wrapper for various algorithms used to find optimal value pair for the embedding delay and the number of embedding dimensions

**Usage**

```
crqa_parameters(y, lagMethods = c("first.minimum", "global.minimum",
  "max.lag"), estimateDimensions = "preferSmallestInLargestHood",
  maxDim = 10, emLag = NULL, maxLag = floor(length(y)/(maxDim + 1)),
  nnSizes = 2, nnRadius = 5, nnThres = 10, theiler = 0,
  doPlot = TRUE, silent = TRUE, ...)
```

**Arguments**

y	A numeric vector or time series
lagMethods	A character vector with one or more of the following strings: "first.minimum", "global.minimum" If emLag represents a valid lag this value will be reported as "user.lag" (default = c("first.minimum", "global.minimum", "max.lag"))
estimateDimensions	Decide on an optimal embedding dimension relative to the values in maxDim and lagMethods, according to a number of preferences passed as a character vector. The order in which the preferences appear in the vector affects the selection procedure, with index 1 being most important preference. The following options are available: <ul style="list-style-type: none"> <li>• preferNone - No optimal number will be picked all other preferences will be ignored</li> <li>• preferSmallestDim - Pick smallest number of dimensions associated with a percentage NN below nnThres</li> <li>• preferSmallestNN - Pick the number of dimensions that is associated with the smallest percentage NN below nnThres</li> <li>• preferSmallestLag - If the value of nnThres does not lead to a unique preference for a pair of dimension and lag values, use the pair with the smallest lag</li> <li>• preferSmallestInLargestHood - The default option: If no unique pair can be found, prefer pairs with smallest values for lag, dimensions, percentage NN for the largest NN size</li> </ul>
maxDim	Maximum number of embedding dimensions to consider (default = 10)
emLag	Optimal embedding lag (delay), e.g., provided by an optimising algorithm. If NULL the lags based on the mutual information in lagMethods will be reported. If a numeric value representing a valid lag is passed, this value will be used to estimate the number of dimensions (default = NULL)

*crqa\_radius*

13

<code>maxLag</code>	Maximum embedding lag to consider. Default value is: <code>floor(length(y)/(maxDim+1))</code>
<code>nnSizes</code>	Points whose distance is <code>nnSize</code> times further apart than the estimated size of the attractor will be declared false neighbours. See the argument <code>atol</code> in <code>fractal::FNN()</code> (default = 2)
<code>nnRadius</code>	If the ratio of the distance between two points in successive dimensions is larger than <code>nnRadius</code> , the points are declared false neighbours. See the argument <code>rto1</code> in <code>fractal::FNN()</code> (default = 5)
<code>nnThres</code>	Threshold value representing the percentage of Nearest Neighbours that would be acceptable when using <code>N</code> surrogate dimensions. The smallest number of surrogate dimensions that yield a value below the threshold will be considered optimal (default = 10)
<code>theiler</code>	Theiler window on distance matrix (default = 0)
<code>doPlot</code>	Produce a diagnostic plot the results (default = TRUE)
<code>silent</code>	Silent-ish mode
<code>...</code>	Other parameters passed to <code>nonlinearTseries::timeLag()</code>

**Details**

A number of functions are called to determine optimal parameters for delay embedding a time series:

- Embedding lag ( $\tau$ , `emLag`): The default is to call `casnet::est_emLag()`, which is a wrapper around `nonlinearTseries::timeLag()` with `technique="ami"` to get lags based on the mutual information function.
- Embedding dimension (`m`, `emDim`): The default is to call `casnet::est_emDim()`, which is a wrapper around `fractal::FNN()`

**Value**

A list object containing the optimal values (as indicated by the user) and iteration history.

**See Also**

Other Recurrence Quantification Analysis: `crqa_cl`, `crqa_radius`, `crqa_rp_measures`, `crqa_rp`

---

<code>crqa_radius</code>	<i>Find fixed or optimal radius</i>
--------------------------	-------------------------------------

---

**Description**

Find fixed or optimal radius

**Usage**

```
crqa_radius(RM = NULL, y1 = NULL, y2 = NULL, emLag = 1,
  emDim = 1, type = c("fixed", "optimal")[1], startRadius = NULL,
  eachRadius = 1, targetMeasure = c("RR", "DET", "LAM", "T1",
  "all")[1], targetValue = 0.05, tol = 0.1, maxIter = 100,
  theiler = -1, histIter = FALSE, noiseLevel = 0.75,
  noiseType = c("normal", "uniform")[1], plotROC = FALSE,
  standardise = c("mean.sd", "median.mad", "none")[3],
  radiusOnFail = c("tiny", "huge", "percentile")[1], silent = FALSE)
```

**Arguments**

RM	Unthresholded Recurrence Matrix
y1	A numeric vector or time series
y2	A numeric vector or time series
emLag	Delay to use for embedding
emDim	Number of embedding dimensions
type	Either "fixed" (default) or "optimal", "fixed" will search for a radius that is close to the value for the targetMeasure in targetValue, "optimal" will optimise the radius for the targetMeasure, targetValue is ignored.
startRadius	If type = "fixed" this is the starting value for the radius (default = percentile of unique distances in RM given by targetValue). If type = "optimal" this will be a range of radius values (in normalised SD units) that will be considered (default = seq(0, 2, by=.01))
eachRadius	If type = "optimal" this is the number of signal and noise series that will be generated for each level in startRadius (default = 1)
targetMeasure	If type = "optimal", it must be a character vector indicating which recurrence measure to optimise the radius for, options are "RR" (default), "DET", "LAM", "T1", and "all". The option targetMeasure = "all" will report all the optimal values obtained from one realisation of startRadius * eachRadius signal and noise series.
targetValue	When argument type is set to "fixed", the value represents the target value for the measure in targetMeasure (default = RR = .05).
tol	Tolerance for achieving targetValue for targetMeasure (default = 0.1)
maxIter	If type = "fixed": Maximum number of iterations to reach targetValue.
theiler	Size of theiler window (default 0)
histIter	Return iteration history? (default = FALSE)
noiseLevel	Noise level to construct the signal + noiseLevel * $N(\mu = 0, \sigma = 1)$ (default = 0.75)
noiseType	Type
plotROC	Generates an ROC plot if type = "optimal"
standardise	Standardise
radiusOnFail	Radius to return when search fails "tiny" = 0 + ,Machine.double.eps, this will likely cause a matrix full of zeros. "huge" = 1 + max. distance in RM, which will give a matrix full of ones, "percentile" = quantile(RM, prob = targetValue) of distances greater than 0.
silent	Silent-ish

**Value**

A dataframe listing settings used to search for the radius, the radius found given the settings and the recurrence rate produced by the radius (either 1 row or the entire iteration history)

**See Also**

Other Recurrence Quantification Analysis: [crqa\\_cl](#), [crqa\\_parameters](#), [crqa\\_rp\\_measures](#), [crqa\\_rp](#)

crqa\_rp

*Get bootstrapped (C)RQA measures based on a recurrence matrix***Description**

A zoo of measures based on singular recurrent points, diagonal, vertical and horizontal line structures will be calculated.

**Usage**

```
crqa_rp(RM, emRad = NA, DLmin = 2, VLmin = 2, HLmin = 2,
        DLmax = length(Matrix::diag(RM)) - 1,
        VLmax = length(Matrix::diag(RM)) - 1,
        HLmax = length(Matrix::diag(RM)) - 1, AUTO = NULL, theiler = NULL,
        chromatic = FALSE, matrices = FALSE, doHalf = FALSE,
        Nboot = NULL, CL = 0.95, targetValue = 0.05, doParallel = FALSE,
        silent = TRUE)
```

**Arguments**

RM	A distance matrix, or a matrix of zeroes and ones (you must set emRad = NA)
emRad	Threshold for distance value that counts as a recurrence
DLmin	Minimal diagonal line length (default = 2)
VLmin	Minimal vertical line length (default = 2)
HLmin	Minimal horizontal line length (default = 2)
DLmax	Maximal diagonal line length (default = length of diagonal -1)
VLmax	Maximal vertical line length (default = length of diagonal -1)
HLmax	Maximal horizontal line length (default = length of diagonal -1)
AUTO	Auto-recurrence? (default = FALSE)
theiler	= Use a theiler window around the line of identity / synchronisation to remove high auto-correlation at short time-lags (default = 0)
chromatic	Force chromatic RQA? (default = FALSE)
matrices	Return matrices? (default = FALSE)
doHalf	Analyse half of the matrix? (default = FALSE)
Nboot	How many bootstrap replications? (default = NULL)
CL	Confidence limit for bootstrap results (default = .95)
targetValue	A value passed to <code>crqa_radius(..., type="fixed", targetMeasure="RR", tol = .2)</code> if <code>is.na(emRad)==TRUE</code> , it will estimate a radius (default = .05).
doParallel	Speed up calculations by using the parallel processing options provided by <code>parallel</code> to assign a separate process/core for each window in windowed (C)RQA analysis using <code>purrr::map2()</code> to assign data and <code>parallel::detectCores()</code> with <code>logical = TRUE</code> to decide on the available cores (default = FALSE)
silent	Do not display any messages (default = TRUE)

**Value**

A list object containing (C)RQA measures (and matrices if requested)

**See Also**

Other Recurrence Quantification Analysis: [crqa\\_cl](#), [crqa\\_parameters](#), [crqa\\_radius](#), [crqa\\_rp\\_measures](#)

---

dc_ccp	<i>Cumulative Complexity Peaks (CCP)</i>
--------	--

---

**Description**

Computes significant peaks in the dynamic complexity time series. Example: Schiepek, Tomin-schek & Heinzel, 2014.

**Usage**

```
dc_ccp(df_win, alpha_item = 0.05, alpha_time = 0.05, doPlot = FALSE,
       useVarNames = TRUE, colOrder = TRUE, useTimeVector = NA,
       timeStamp = "01-01-1999")
```

**Arguments**

df_win	A data frame containing series of Dynamic Complexity values obtained by running function <a href="#">dc_win()</a>
alpha_item	The significance level of the one-sided Z-test used to determine which peaks are $> 0$ .
alpha_time	The significance level of the one-sided Z-test used to determine if the number of significant peaks (as determined by <code>alpha_item</code> ) at a specific time stamp are $> 0$ .
doPlot	If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible <a href="#">ggplot2::ggplot()</a> object. (default = FALSE)
useVarNames	Use the column names of df as variable names in the Complexity Resonance Diagram (default = TRUE)
colOrder	If TRUE, the order of the columns in df determines the of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by by mean value of Dynamic Complexity (default = TRUE)
useTimeVector	Parameter used for plotting. A vector of length <code>NROW(df)</code> , containing date/time information (default = NA)
timeStamp	If useTimeVector is not NA, a character string that can be passed to <a href="#">lubridate::stamp()</a> to format the the dates/times passed in useTimeVector (default = "01-01-1999")

**Value**

A list with a dataframe of binary complexity peak indices and a cumulative complexity peak index, a CCP diagram.

**Author(s)**

Merlijn Olthof  
Fred Hasselman



## References

Schiepek, G., & Strunk, G. (2010). The identification of critical fluctuations and phase transitions in short term and coarse-grained time series-a method for the real-time monitoring of human change processes. *Biological cybernetics*, 102(3), 197-207.

Schiepek, G. (2003). A Dynamic Systems Approach to Clinical Case Formulation. *European Journal of Psychological Assessment*, 19, 175-184.

Haken, H. & Schiepek, G. (2006, 2. Aufl. 2010). *Synergetik in der Psychologie. Selbstorganisation verstehen und gestalten*. Göttingen: Hogrefe.

Schiepek, G. K., Tominschek, I., & Heinzl, S. (2014). Self-organization in psychotherapy: testing the synergetic model of change processes. *Frontiers in psychology*, 5, 1089.

## See Also

Other Dynamic Complexity functions: [dc\\_d](#), [dc\\_f](#), [dc\\_win](#), [plotDC\\_ccp](#), [plotDC\\_lvl](#), [plotDC\\_res](#)

---

dc_d	<i>Distribution Uniformity</i>
------	--------------------------------

---

## Description

Distribution Uniformity is one of two components of which the product is the Dynamic Complexity measure.

## Usage

```
dc_d(df, win = NROW(df), scale_min, scale_max, doPlot = FALSE,
     useVarNames = TRUE, colOrder = TRUE, useTimeVector = NA,
     timeStamp = "01-01-1999")
```

## Arguments

df	A dataframe containing multivariate time series data from 1 person. Rows should indicate time, columns should indicate the time series variables. All time series in df should be on the same scale, an error will be thrown if the range of the time series in df is not [scale_min, scale_max].
win	Size of window in which to calculate Dynamic Complexity. If win < NROW(df) the window will move along the time series with a stepsize of 1 (default = NROW(df))
scale_min	The theoretical minimum value of the scale. Used to calculate expected values, so it is important to set this to the correct value.
scale_max	The theoretical maximum value of the scale. Used to calculate expected values, so it is important to set this to the correct value.
doPlot	If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE)
useVarNames	Use the column names of df as variable names in the Complexity Resonance Diagram (default = TRUE)
colOrder	If TRUE, the order of the columns in df determines the order of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by mean value of Dynamic Complexity (default = TRUE)

18

*dc\_f*

**useTimeVector** Parameter used for plotting. A vector of length `NROW(df)`, containing date/time information (default = NA)

**timeStamp** If `useTimeVector` is not NA, a character string that can be passed to `lubridate::stamp()` to format the the dates/times passed in `useTimeVector` (default = "01-01-1999")

**Value**

a dataframe

**See Also**Use `dc_win()` to get the Dynamic Complexity measure.Other Dynamic Complexity functions: `dc_ccp`, `dc_f`, `dc_win`, `plotDC_ccp`, `plotDC_lvl`, `plotDC_res`


---

<code>dc_f</code>	<i>Fluctuation Intensity</i>
-------------------	------------------------------

---

**Description**

Fluctuation intensity is one of two components of which the product is the Dynamic Complexity measure.

**Usage**

```
dc_f(df, win = NROW(df), scale_min, scale_max, doPlot = FALSE,
     useVarNames = TRUE, colOrder = TRUE, useTimeVector = NA,
     timeStamp = "01-01-1999")
```

**Arguments**

**df** A dataframe containing multivariate time series data from 1 person. Rows should indicate time, columns should indicate the time series variables. All time series in `df` should be on the same scale, an error will be thrown if the range of the time series in `df` is not `[scale_min, scale_max]`.

**win** Size of window in which to calculate Dynamic Complexity. If `win < NROW(df)` the window will move along the time series with a stepsize of 1 (default = `NROW(df)`)

**scale\_min** The theoretical minimum value of the scale. Used to calculate expected values, so it is important to set this to the correct value.

**scale\_max** The theoretical maximum value of the scale. Used to calculate expected values, so it is important to set this to the correct value.

**doPlot** If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible `ggplot2::ggplot()` object. (default = FALSE)

**useVarNames** Use the column names of `df` as variable names in the Complexity Resonance Diagram (default = TRUE)

**colOrder** If TRUE, the order of the columns in `df` determines the of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by mean value of Dynamic Complexity (default = TRUE)

*dc\_f2*

19

**useTimeVector**    Parameter used for plotting. A vector of length `NROW(df)`, containing date/time information (default = NA)

**timeStamp**        If `useTimeVector` is not NA, a character string that can be passed to `lubridate::stamp()` to format the the dates/times passed in `useTimeVector` (default = "01-01-1999")

**Value**

dataframe

**See Also**

Use `dc_win()` to get the dynamic complexity measure.

Other Dynamic Complexity functions: `dc_ccp`, `dc_d`, `dc_win`, `plotDC_ccp`, `plotDC_lvl`, `plotDC_res`

---

<i>dc_f2</i>	<i>Title</i>
--------------	--------------

---

**Description**

Title

**Usage**

```
dc_f2(df, win = NROW(df), scale_min, scale_max, doPlot = FALSE,
      useVarNames = TRUE, colOrder = TRUE, useTimeVector = NA,
      timeStamp = "01-01-1999")
```

---

<i>dc_win</i>	<i>Dynamic Complexity</i>
---------------	---------------------------

---

**Description**

Calculates Dynamic Complexity, a complexity index for short and coarse-grained time series (Schiepek & Strunk, 2010; Schiepek, 2003; Haken & Schiepek 2006).

**Usage**

```
dc_win(df, win = NROW(df), scale_min, scale_max, doPlot = FALSE,
       doPlotF = FALSE, doPlotD = FALSE, returnFandD = FALSE,
       useVarNames = TRUE, colOrder = TRUE, useTimeVector = NA,
       timeStamp = "01-01-1999")
```

**Arguments**

df	A dataframe containing multivariate time series data from 1 person. Rows should indicate time, columns should indicate the time series variables. All time series in df should be on the same scale, an error will be thrown if the range of the time series in df is not [scale_min, scale_max].
win	Size of window in which to calculate Dynamic Complexity. If win < NROW(df) the window will move along the time series with a stepsize of 1 (default = NROW(df))
scale_min	The theoretical minimum value of the scale. Used to calculate expected values, so it is important to set this to the correct value.
scale_max	The theoretical maximum value of the scale. Used to calculate expected values, so it is important to set this to the correct value.
doPlot	If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE)
doPlotF	If TRUE shows a Complexity Resonance Diagram of the Fluctuation Intensity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE) #' @param doPlotD If TRUE shows a Complexity Resonance Diagram of the Distribution Uniformity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE)
returnFandD	Returns a list object containing the dynamic complexity series as well as the F and D series. (default = FALSE)
useVarNames	Use the column names of df as variable names in the Complexity Resonance Diagram (default = TRUE)
colOrder	If TRUE, the order of the columns in df determines the order of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by mean value of Dynamic Complexity (default = TRUE)
useTimeVector	Parameter used for plotting. A vector of length NROW(df), containing date/time information (default = NA)
timeStamp	If useTimeVector is not NA, a character string that can be passed to <code>lubridate::stamp()</code> to format the dates/times passed in useTimeVector (default = "01-01-1999")

**Value**

If doPlot = TRUE, a list object containing a data frame of Dynamic Complexity values and a ggplot2 object of the dynamic complexity resonance diagram (e.g. Schiepek et al., 2016). If doPlot = FALSE the data frame with Dynamic Complexity series is returned.

**Author(s)**

Merlijn Olthof  
Fred Hasselman

**References**

Schiepek, G., & Strunk, G. (2010). The identification of critical fluctuations and phase transitions in short term and coarse-grained time series-a method for the real-time monitoring of human change processes. *Biological cybernetics*, 102(3), 197-207.

Schiepek, G. (2003). A Dynamic Systems Approach to Clinical Case Formulation. *European Journal of Psychological Assessment*, 19, 175-184.

Haken, H. & Schiepek, G. (2006, 2. Aufl. 2010). Synergetik in der Psychologie. Selbstorganisation verstehen und gestalten. Göttingen: Hogrefe.

Schiepek, G. K., Stöcker-Schmidinger, B., Aichhorn, W., Schüller, H., & Aas, B. (2016). Systemic case formulation, individualized process monitoring, and state dynamics in a case of dissociative identity disorder. *Frontiers in psychology*, 7, 1545.

## See Also

Other Dynamic Complexity functions: [dc\\_ccp](#), [dc\\_d](#), [dc\\_f](#), [plotDC\\_ccp](#), [plotDC\\_lv1](#), [plotDC\\_res](#)

---

di2bi	<i>Distance to binary matrix</i>
-------	----------------------------------

---

## Description

Distance matrix to binary matrix based on threshold value

## Usage

```
di2bi(distmat, emRad, theiler = 0, convMat = FALSE)
```

## Arguments

distmat	Distance matrix
emRad	The radius or threshold value
theiler	= Use a theiler window around the line of identity / synchronisation to remove high auto-correlation at short time-lags (default = 0)
convMat	Should the matrix be converted from a distmat object of class <code>Matrix::Matrix()</code> to <code>base::matrix()</code> (or vice versa)

## Value

A (sparse) matrix with only 0s and 1s

## See Also

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2we](#), [dist\\_hamming](#), [rp\\_lineDist](#), [rp\\_nzdiags](#), [rp\\_plot](#), [rp\\_size](#), [rp](#)

Other Distance matrix operations (recurrence network): [di2we](#), [rn\\_plot](#), [rn\\_recSpec](#), [rn\\_scaleoGram](#), [rn](#)

---

di2we	<i>Distance 2 weighted matrix</i>
-------	-----------------------------------

---

**Description**

Distance matrix to weighted matrix based on threshold value

**Usage**

```
di2we(distmat, emRad, convMat = FALSE)
```

**Arguments**

distmat	Distance matrix
emRad	The radius or threshold value
convMat	convMat Should the matrix be converted from a distmat object of class <code>Matrix::Matrix()</code> to <code>base::matrix()</code> (or vice versa)

**Value**

A matrix with 0s and leaves the values < threshold distance value

**See Also**

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2bi](#), [dist\\_hamming](#), [rp\\_lineDist](#), [rp\\_nzdiags](#), [rp\\_plot](#), [rp\\_size](#), [rp](#)

Other Distance matrix operations (recurrence network): [di2bi](#), [rn\\_plot](#), [rn\\_recSpec](#), [rn\\_scaleoGram](#), [rn](#)

---

dist_hamming	<i>Calculate Hamming distance</i>
--------------	-----------------------------------

---

**Description**

Calculate Hamming distance

**Usage**

```
dist_hamming(X, Y = NULL, embedded = TRUE)
```

**Arguments**

X	A matrix (of coordinates)
Y	A matrix (of coordinates)
embedded	Do X and/or Y represent surrogate dimensions of an embedded time series?

**Value**

A hamming-distance matrix of X, or X and Y. Useful for ordered and unordered categorical data.

**Author(s)**

Fred Hasselman

**See Also**

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2bi](#), [di2we](#), [rp\\_lineDist](#), [rp\\_nzdiags](#), [rp\\_plot](#), [rp\\_size](#), [rp](#)

elascer

*Elastic Scaler - A Flexible Rescale Function***Description**

The 'elastic scaler' will rescale numeric vectors (1D, or columns in a matrix or data.frame) to a user defined minimum and maximum, either based on the extrema in the data, or, a minimum and maximum defined by the user.

**Usage**

```
elascer(x, mn = NA, mx = NA, lo = 0, hi = 1, groupwise = FALSE,
        keepNA = TRUE, boundaryPrecision = NA,
        tol = .Machine$double.eps^0.5)
```

**Arguments**

x	Input vector or data frame.
mn	Minimum value of original, defaults to <code>min(x, na.rm = TRUE)</code> if set to NA.
mx	Maximum value of original, defaults to <code>max(x, na.rm = TRUE)</code> if set to NA.
lo	Minimum value to rescale to, defaults to 0.
hi	Maximum value to rescale to, defaults to 1.
groupwise	If x is a data frame with 2+ columns, mn = NA and/or mx = NA and groupwise = TRUE, scaling will occur
keepNA	Keep NA values?
boundaryPrecision	If set to NA the precision of the input will be the same as the precision of the output. This can cause problems when detecting values that lie just outside of, or, exactly on boundaries given by lo and hi, e.g. after saving the data using a default precision. Setting boundaryPrecision to an integer value will ensure that the boundaries of the new scale are given by <code>round(..., digits = boundaryPrecision)</code> . Alternatively one could just round all the output after rescaling to a desired precision (default = NA)
tol	The tolerance for deciding whether a value is on the boundary lo or hi (default = <code>.Machine\$double.eps^0.5</code> )

**Details**

Three uses:

1. `elascer(x)` - Scale x to data range: `min(x.out)==0; max(x.out)==1`
2. `elascer(x,mn,mx)` - Scale x to arg. range: `min(x.out)==mn==0; max(x.out)==mx==1`
3. `elascer(x,mn,mx,lo,hi)` - Scale x to arg. range: `min(x.out)==mn==lo; max(x.out)==mx==hi`

**Value**

scaled inout

**Examples**

```
# Works on numeric objects
somenumbers <- cbind(c(-5,100,sqrt(2)),c(exp(1),0,-pi))

# Using the defaults:
# 1. mn and mx are derived globally (groupWise = FALSE)
# 2. values rescaled to hi and lo are integers, 0 and 1 respectively
elascer(somenumbers)

# If the data contain values < mn they will return as < lo
elascer(somenumbers,mn=-100)
# If the data contain values > mx they will return > hi
elascer(somenumbers,mx=99)

# Effect of setting groupWise
elascer(somenumbers,lo=-1,hi=1)
elascer(somenumbers,lo=-1,hi=1, groupwise = TRUE)

elascer(somenumbers,mn=-10,mx=100,lo=-1,hi=4)
elascer(somenumbers,mn= NA,mx=100,lo=-1,hi=4, groupwise = TRUE)

# Effect of setting boundaryPrecision
x <- rbind(1/3, 1/7)

re1 <- elascr(x, lo = 0, hi = 1/13, boundaryPrecision = NA)
max(re1)==0.07692308 # FALSE
max(re1)==1/13      # TRUE

re2 <- elascr(x, lo = 0, hi = 1/13, boundaryPrecision = 8)
max(re2)==0.07692308 # TRUE
max(re2)==1/13      # FALSE
```

est\_emDim

*Estimate number of embedding dimensions***Description**

A wrapper for nonlinearTseries::estimateEmbeddingDim

**Usage**

```
est_emDim(y, delay = est_emLag(y), maxDim = 15, threshold = 0.95,
  max.relative.change = 0.1, doPlot = FALSE, ...)
```

**Arguments**

y	Time series or numeric vector
delay	Embedding lag



`est_emLag`

25

<code>maxDim</code>	Maximum number of embedding dimensions
<code>threshold</code>	See <code>nonlinearTseries::estimateEmbeddingDim()</code>
<code>max.relative.change</code>	See <code>nonlinearTseries::estimateEmbeddingDim()</code>
<code>doPlot</code>	Plot
<code>...</code>	Other arguments (not in use)

**Value**

Embedding dimensions

---

<code>est_emLag</code>	<i>Estimate embedding lag (tau)</i>
------------------------	-------------------------------------

---

**Description**A wrapper for `nonlinearTseries::timemLag`**Usage**

```
est_emLag(y, selection.methods = "first.minimum", maxLag = length(y)/4,
...)
```

**Arguments**

<code>y</code>	Time series or numeric vector
<code>selection.methods</code>	Selecting an optimal embedding lag (default: Return "first.e.decay", "first.zero", "first.minimum", "first.value", where value is $1/\exp(1)$ )
<code>maxLag</code>	Maximal lag to consider (default: 1/4 of timeseries length)
<code>...</code>	Additional parameters

**Value**

The ami function with requested minima

---

factor_obs_exp	<i>Add expected factor labels to observed values</i>
----------------	--

---

**Description**

Add expected factor labels to observed values

**Usage**

```
factor_obs_exp(observed_Ncat, observed_labels, expected_Ncat = 0,
               expected_labels = "", varname = "")
```

**Arguments**

observed_Ncat	obsN
observed_labels	
	obsL
expected_Ncat	expN
expected_labels	
	expL
varname	varname

**Value**

character vector

**See Also**

Other State Space Grid functions: [ssg\\_gwf2long](#), [ssg\\_winning](#)

---

fd_allan	<i>Allan Variance Analysis</i>
----------	--------------------------------

---

**Description**

Allan Variance Analysis

**Usage**

```
fd_allan(y, fs = stats::tsp(stats::hasTsp(y))[3], useSD = FALSE,
         doPlot = FALSE, returnPlot = FALSE, returnPLAW = FALSE,
         returnInfo = FALSE, silent = FALSE, noTitle = FALSE,
         tsName = "y")
```

**Arguments**

y	A numeric vector or time series object
fs	Sample frequency in Hz
useSD	Use the standard deviation instead of variance?
doPlot	Return the log-log scale versus fluctuation plot with linear fit (default = TRUE).
returnPlot	Return ggplot2 object (default = FALSE)
returnPLAW	Return the power law data (default = FALSE)
returnInfo	Return all the data used in DFA (default = FALSE)
silent	Silent-ish mode
noTitle	Do not generate a title (only the subtitle)
tsName	Name of y added as a subtitle to the plot

**Value**

A dataframe with the Allan Factor (variance), Alan standard deviation and error due to bin size

**See Also**

Other Fluctuation Analyses: [fd\\_RR](#), [fd\\_dfa](#), [fd\\_mfdfa](#), [fd\\_psd](#), [fd\\_sda](#), [fd\\_sev](#)

---

fd_boxcount2D	<i>2D Boxcount for 1D signal</i>
---------------	----------------------------------

---

**Description**

2D Boxcount for 1D signal

**Usage**

```
fd_boxcount2D(y = NA, unitSquare = TRUE, image2D = NA,
  resolution = 1, removeTrend = FALSE, polyOrder = 1,
  standardise = c("none", "mean.sd", "median.mad")[1],
  adjustSumOrder = FALSE, scaleMin = 0, scaleMax = floor(log2(NROW(y)
    * resolution)), scaleS = NA, minData = 2^(scaleMin + 1),
  maxData = 2^(scaleMax - 1), doPlot = FALSE, returnPlot = FALSE,
  returnPLAW = FALSE, returnInfo = FALSE, returnLocalScaling = FALSE,
  silent = FALSE, noTitle = FALSE, tsName = "y")
```

**Arguments**

y	A numeric vector or time series object.
unitSquare	Create unit square image of y? This is required for estimating FD of time series (default = TRUE)
image2D	A matrix representing a 2D image, argument y and unitSquare will be ignored (default = NA)
resolution	The resolution used to embed the timeseries in 2D, a factor by which the dimensions the matrix will be multiplied (default = 1)

removeTrend	If TRUE, will call <a href="#">ts_detrend</a> on y (default = FALSE)
polyOrder	Order of polynomial trend to remove if removeTrend = TRUE
standardise	Standardise y using <a href="#">casnet::ts_standardise()</a> with adjustN = FALSE (default = none)
adjustSumOrder	Adjust the order of the time series (by summation or differencing), based on the global scaling exponent, see e.g. <a href="https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg">https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg</a> Ihlen (2012) (default = 'FALSE')
scaleMin	Minimum scale value (as $2^{\text{scale}}$ ) to use (default = 0)
scaleMax	Maximum scale value (as $2^{\text{scale}}$ ) to use (default = max of $\log_2(\text{nrows})$ and $\log_2(\text{ncols})$ )
scales	If not NA, pass a numeric vector listing the scales (as a power of 2) on which to evaluate the boxcount. Arguments scaleMax, scaleMin, and scaleResolution will be ignored (default = NA)
minData	Minimum number of time/data points inside a box for it to be included in the slope estimation (default = $2^{\text{scaleMin}}$ )
maxData	Maximum number of time/data points inside a box for it to be included in the slope estimation (default = $2^{\text{scaleMax}}$ )
doPlot	Return the log-log scale versus bulk plot with linear fit (default = TRUE).
returnPlot	Return ggplot2 object (default = FALSE)
returnPLAW	Return the power law data (default = FALSE)
returnInfo	Return all the data used in DFA (default = FALSE)
returnLocalScaling	Return estimates of FD for each scale
silent	Silent-ish mode (default = TRUE)
noTitle	Do not generate a title (only the subtitle)
tsName	Name of y added as a subtitle to the plot (default = y)

**Value**

The boxcount fractal dimension and the 'local' boscount fractal dimension

**Note**

This function was inspired by the Matlab function `boxcount.m` written by F. Moisy. Fred Hasselman adapted the function for R for the purpose of the unit square boxcount analysis for 1D time series. The original Matlab toolbox has more options and contains more functions (e.g. 1D and 3D boxcount).

**Examples**

```
fd_boxcount2D(y = rnorm(100))
```

fd\_dfa

*Detrended Fluctuation Analysis (DFA)***Description**

fd\_dfa

**Usage**

```
fd_dfa(y, fs = NULL, removeTrend = c("no", "poly", "adaptive",
  "bridge")[2], polyOrder = 1, standardise = c("none", "mean.sd",
  "median.mad")[2], adjustSumOrder = FALSE, scaleMin = 2,
  scaleMax = floor(log2(NROW(y)/2)), scaleResolution = (scaleMax -
  scaleMin), scales = NA, overlap = 0, minData = 4, doPlot = FALSE,
  returnPlot = FALSE, returnPLAW = FALSE, returnInfo = FALSE,
  silent = FALSE, noTitle = FALSE, tsName = "y")
```

**Arguments**

y	A numeric vector or time series object.
fs	Sample rate
removeTrend	Method to use for detrending, see <code>fractal::DFA()</code> (default = "poly")
polyOrder	Order of polynomial trend to remove if removeTrend = "poly"
standardise	Standardise by the series using <code>casnet::ts_standardise()</code> with adjustN = FALSE (default = "mean.sd")
adjustSumOrder	Adjust the time series (summation or differencing), based on the global scaling exponent, see e.g. <a href="https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg">https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg</a> Ihlen (2012) (default = FALSE)
scaleMin	Minimum scale (as a power of 2) to use
scaleMax	Maximum scale (as a power of 2) to use
scaleResolution	The scales at which detrended fluctuation will be evaluated are calculated as: $(\text{scaleMax} - \text{scaleMin}) / \text{scaleResolution}$ . The default value yields no resolution of scales: $(\text{scaleMax} - \text{scaleMin})$ . Common values
scales	If not NA, it should be a numeric vector listing the scales on which to evaluate the detrended fluctuations. Arguments scaleMax, scaleMin, scaleResolution will be ignored.
overlap	Turn DFA into a sliding window analysis. A number in $[0 \dots 1]$ representing the amount of 'bin overlap'. If $\text{length}(y) = 1024$ and overlap is .5, a scale of 4 will be considered a sliding window of size 4 with stepsize $\text{floor}(.5 * 4) = 2$ . The detrended fluctuation in For scale 128 this will be (default = 0)
minData	Minimum number of data points in a bin needed to calculate detrended fluctuation
doPlot	Return the log-log scale versus fluctuation plot with linear fit (default = TRUE).
returnPlot	Return ggplot2 object (default = FALSE)
returnPLAW	Return the power law data (default = FALSE)

returnInfo	Return all the data used in DFA (default = FALSE)
silent	Silent-ish mode
noTitle	Do not generate a title (only the subtitle)
tsName	Name of y added as a subtitle to the plot

**Value**

Estimate of Hurst exponent (slope of  $\log(\text{bin})$  vs.  $\log(\text{RMSE})$ ) and an FD estimate based on Hasselman(2013) A list object containing:

- A data matrix PLAW with columns `freq.norm`, `size` and `bulk`.
- Estimate of scaling exponent `sap` based on a fit over the standard range (`fullRange`), or on a user defined range `fitRange`.
- Estimate of the the Fractal Dimension (FD) using conversion formula's reported in Hasselman(2013).
- Information output by various functions.

**Author(s)**

Fred Hasselman

**References**

Hasselman, F. (2013). When the blind curve is finite: dimension estimation and model inference based on empirical waveforms. *Frontiers in Physiology*, 4, 75. <http://doi.org/10.3389/fphys.2013.00075>

**See Also**

Other Fluctuation Analyses: [fd\\_RR](#), [fd\\_allan](#), [fd\\_mfdfa](#), [fd\\_psd](#), [fd\\_sda](#), [fd\\_sev](#)

---

fd\_mfdfa

*Multi-fractal Detrended Fluctuation Analysis*

---

**Description**

Multi-fractal Detrended Fluctuation Analysis

**Usage**

```
fd_mfdfa(signal, qq = c(-10, -5:5, 10), mins = 6, maxs = 12,
  ressc = 30, m = 1)
```

**Arguments**

signal	An input signal.
qq	A vector containing a range of values for the order of fluctuation $q$ .
mins	Minimum scale to consider.
maxs	Maximum scale to consider.
ressc	ressc
m	m

*fd\_psd*

31

**Value**

output

**See Also**Other Fluctuation Analyses: [fd\\_RR](#), [fd\\_allan](#), [fd\\_dfa](#), [fd\\_psd](#), [fd\\_sda](#), [fd\\_sev](#)


---

<i>fd_psd</i>	<i>Power Spectral Density Slope (PSD).</i>
---------------	--

---

**Description**

Estimate Alpha, Hurst Exponent and Fractal Dimension through log-log slope.

**Usage**

```
fd_psd(y, fs = NULL, standardise = TRUE, detrend = TRUE,
       fitMethod = c("lowest25", "Wijnants", "Hurvich-Deo")[3],
       doPlot = FALSE, returnPlot = FALSE, returnPLAW = FALSE,
       returnInfo = FALSE, silent = FALSE, noTitle = FALSE,
       tsName = "y")
```

**Arguments**

<i>y</i>	A numeric vector or time series object.
<i>fs</i>	Sample rate (default = NULL)
<i>standardise</i>	standardise the series (default = TRUE).
<i>detrend</i>	Subtract linear trend from the series (default = TRUE).
<i>fitMethod</i>	Method to decide on a frequency range for log-log fit. Can be one of: "lowest25", "Wijnants", "Hurvich-Deo" (default). See details for more info.
<i>doPlot</i>	Return the log-log spectrum with linear fit (default = TRUE).
<i>returnPlot</i>	Return ggplot2 object (default = FALSE)
<i>returnPLAW</i>	Return the power law data (default = FALSE)
<i>returnInfo</i>	Return all the data used in SDA (default = FALSE)
<i>silent</i>	Run in silent-ish mode (default = TRUE)
<i>noTitle</i>	Do not generate a title (only the subtitle)
<i>tsName</i>	Name of y added as a subtitle to the plot

**Details**

Calls function [sapa::SDF\(\)](#) to estimate the scaling exponent of a timeseries based on the periodogram frequency spectrum. After detrending and normalizing the signal (if requested), SDF is called using a Tukey window (raised cosine [taper](#)).

A line is fitted on the periodogram in log-log coordinates. The full range is fitted as well as one of three fit-ranges:

- lowest25 - The 25% lowest frequencies
- Wijnants - The 50 lowest frequencies (Wijnants et al., 2012)
- HurvichDeo - The Hurvich-Deo estimate, see ([fractal::HDEst\(\)](#))

## Value

A list object containing:

- A data matrix PLAW with columns freq.norm, size and bulk.
- Estimate of scaling exponent alpha based on a fit over the lowest 25% frequencies (low25), or using the HD estimate HD.
- Estimate of the the Fractal Dimension (FD) using conversion formula's reported in Hasselman(2013).
- Information output by various functions.

## Author(s)

Fred Hasselman

## References

Hasselman, F. (2013). When the blind curve is finite: dimension estimation and model inference based on empirical waveforms. *Frontiers in Physiology*, 4, 75. <http://doi.org/10.3389/fphys.2013.00075>

## See Also

Other Fluctuation Analyses: [fd\\_RR](#), [fd\\_allan](#), [fd\\_dfa](#), [fd\\_mfdfa](#), [fd\\_sda](#), [fd\\_sev](#)

---

fd_RR	<i>Relative Roughness</i>
-------	---------------------------

---

## Description

Relative Roughness is a ratio of local variance (autocovariance at lag-1) to global variance (autocovariance at lag-0) that can be used to classify different 'noises'.

## Usage

fd\_RR(y)

## Arguments

y                      A numeric vector.

## Details

$$RR = 2 * \left[ 1 - \frac{\gamma(y)}{Var(y)} \right]$$

## Value

The Relative Roughness of y, the values of local and global variance are returned as attributes



References

- Marmelat, V., Torre, K., & Delignieres, D. (2012). Relative roughness: an index for testing the suitability of the monofractal model. *Frontiers in Physiology*, 3, 208.

See Also

Other Fluctuation Analyses: [fd\\_allan](#), [fd\\_dfa](#), [fd\\_mfdfa](#), [fd\\_psd](#), [fd\\_sda](#), [fd\\_sev](#)

---

fd_sda	<i>Standardised Dispersion Analysis (SDA).</i>
--------	--

---

Description

fd\_sda

Usage

```
fd_sda(y, fs = NULL, standardise = c("mean.sd", "median.mad")[1],
  detrend = FALSE, polyOrder = 1, adjustSumOrder = FALSE,
  scaleMin = 2, scaleMax = floor(log2(NROW(y)/2)),
  scaleResolution = 30, scales = NA, overlap = 0, minData = 4,
  doPlot = FALSE, returnPlot = FALSE, returnPLAW = FALSE,
  returnInfo = FALSE, silent = FALSE, noTitle = FALSE,
  tsName = "y")
```

Arguments

y	A numeric vector or time series object.
fs	Sample rate (default = NULL)
standardise	standardise the series (default = "mean.sd")
detrend	Subtract linear trend from the series (default = FALSE)
polyOrder	Order of detrending polynomial
adjustSumOrder	Adjust the time series (summation or differencing), based on the global scaling exponent, see e.g. <a href="https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg">https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg</a> Ihlen (2012) (default = FALSE)
scaleMin	Minimum scale to use
scaleMax	Maximum scale to use
scaleResolution	The scales at which the standardised fluctuations are calculated as: (scaleMax-scaleMin)/scaleRes
scales	If not NA, it should be a numeric vector listing the scales on which to evaluate the fluctuations. Arguments scaleMax, scaleMin, scaleResolution will be ignored.
overlap	Turn SDA into a sliding window analysis. A number in [0 ... 1] representing the amount of 'bin overlap'. If length(y) = 1024 and overlap is .5, a scale of 4 will be considered a sliding window of size 4 with stepsize floor(.5 * 4) = 2 (default = 0)

minData	Minimum number of data points in a bin needed to calculate standardised dispersion
doPlot	Output the log-log scale versus fluctuation plot with linear fit by calling function <code>plotFD_loglog()</code> (default = TRUE)
returnPlot	Return ggplot2 object (default = FALSE)
returnPLAW	Return the power law data (default = FALSE)
returnInfo	Return all the data used in SDA (default = FALSE)
silent	Silent-ish mode
noTitle	Do not generate a title (only the subtitle)
tsName	Name of y added as a subtitle to the plot

**Value**

A list object containing:

- A data matrix PLAW with columns `freq.norm`, `size` and `bulk`.
- Estimate of scaling exponent `sap` based on a fit over the standard range (`fullRange`), or on a user defined range `fitRange`.
- Estimate of the the Fractal Dimension (FD) using conversion formula's reported in Hasselman(2013).
- Information output by various functions.

**Author(s)**

Fred Hasselman

**References**

Hasselman, F. (2013). When the blind curve is finite: dimension estimation and model inference based on empirical waveforms. *Frontiers in Physiology*, 4, 75. <http://doi.org/10.3389/fphys.2013.00075>

**See Also**

Other Fluctuation Analyses: [fd\\_RR](#), [fd\\_allan](#), [fd\\_dfa](#), [fd\\_mfdfa](#), [fd\\_psd](#), [fd\\_sev](#)

---

 fd\_sev

---

*Calculate FD using Sevcik's method*


---

**Description**

Calculate FD using Sevcik's method

**Usage**

```
fd_sev(y, detrend = FALSE, adjustSumOrder = FALSE,
       smallNapprox = FALSE, doPlot = FALSE, returnPlot = FALSE,
       returnPLAW = FALSE, returnInfo = FALSE, silent = FALSE,
       noTitle = FALSE, tsName = "y")
```

**Arguments**

y	A time series or numeric vector
detrend	Subtract linear trend from the series (default = TRUE).
adjustSumOrder	Adjust the time series (summation or differencing), based on the global scaling exponent, see e.g. <a href="https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg">https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg</a> Ihlen (2012) (default = TRUE)
smallNapprox	Force use of small sample approximation (default for N < 128)
doPlot	Return the log-log scale versus fluctuation plot with linear fit (default = TRUE).
returnPlot	Return ggplot2 object (default = FALSE)
returnPLAW	Return the power law data (default = FALSE)
returnInfo	Return all the data used in DFA (default = FALSE)
silent	Silent-ish mode
noTitle	Do not generate a title (only the subtitle)
tsName	Name of y added as a subtitle to the plot

**Value**

An FD estimate

**Author(s)**

Fred Hasselman

**References**

Sevcik, C. (1998). A procedure to Estimate the Fractal Dimension of Waveforms. Paper available at <http://arxiv.org/pdf/1003.5266.pdf>

**See Also**

Other Fluctuation Analyses: [fd\\_RR](#), [fd\\_allan](#), [fd\\_dfa](#), [fd\\_mfdfa](#), [fd\\_psd](#), [fd\\_sda](#)

---

flight\_Cauchy

---

Create Cauchy Flight

---

**Description**

Creates a Cauchy flight by taking increments from the Cauchy distributions implemented as the stable distribution (`stabledist::rstable()`) with index parameter  $\alpha = 1$  and skewness parameter  $\beta = 0$ .

**Usage**

```
flight_Cauchy(N = 1000, ndims = 2, alpha = 1, beta = 0,
  scale = 1, location = 0)
```

**Arguments**

N	Length of time series (default = 1000)
ndims	Number of dimensions (default = 2)
alpha	Index of stability parameter in $(0, 2]$
beta	Skewness parameter in $[-1, 1]$
scale	Scale parameter in $(0, \text{Inf})$
location	Location (shift) parameter in $[-\text{Inf}, \text{Inf}]$

**Value**

A data frame with ndims columns and N rows.

**Examples**

```
df <- flight_Cauchy()
plot(density(diff(df$dim1)))
plot(df$dim1, df$dim2, type = "l")
```

---

flight_LevyPareto	Create a Levy-Pareto flight
-------------------	-----------------------------

---

**Description**

Creates a Rayleigh flight by taking increments from the Normal distributions implemented as the stable distribution (`stabledist::rstable()`) with index parameter  $\alpha = 1.5$  and skewness parameter  $\beta = 0$ .

**Usage**

```
flight_LevyPareto(N = 1000, ndims = 2, alpha = 1.5, beta = 0,
  scale = 1, location = 0)
```

**Arguments**

N	Length of time series (default = 1000)
ndims	Number of dimensions (default = 2)
alpha	Index of stability parameter in $(0, 2]$
beta	Skewness parameter in $[-1, 1]$
scale	Scale parameter in $(0, \text{Inf})$
location	Location (shift) parameter in $[-\text{Inf}, \text{Inf}]$

**Details**

Note that the increments are not strictly from the distribution called **the** Levy distribution, but rather **a** a Levy-with-Pareto-tail-type distribution (i.e. when  $1 < \alpha < 2$ ). Use  $\alpha = 1/2$  and  $\beta = 1$  if **the** Levy distribution is required.

*flight\_Rayleigh*

37

**Value**

A data frame with ndims columns and N rows.

**Examples**

```
# Levy-Pareto
df <- flight_LevyPareto()
plot(density(diff(df$dim1)))
plot(df$dim1, df$dim2, type = "l")

# "The" Levy distribution
df <- flight_LevyPareto(alpha = 1/2, beta = 1)
plot(density(diff(df$dim1)))
plot(df$dim1, df$dim2, type = "l")
```

---

<code>flight_Rayleigh</code>	<i>Create Rayleigh Flight (Brownian Motion)</i>
------------------------------	---

---

**Description**

Creates a Rayleigh flight by taking increments from the Normal distributions implemented as the stable distribution (`stabledist::rstable()`) with index parameter  $\alpha = 2$  and skewness parameter  $\beta = 0$ .

**Usage**

```
flight_Rayleigh(N = 1000, ndims = 2, alpha = 2, beta = 0,
  scale = 1, location = 0)
```

**Arguments**

<code>N</code>	Length of time series (default = 1000)
<code>ndims</code>	Number of dimensions (default = 2)
<code>alpha</code>	Index of stability parameter in $(0, 2]$
<code>beta</code>	Skewness parameter in $[-1, 1]$
<code>scale</code>	Scale parameter in $(0, \text{Inf})$
<code>location</code>	Location (shift) parameter in $[-\text{Inf}, \text{Inf}]$

**Value**

A data frame with ndims columns and N rows.

**Examples**

```
df <- flight_Rayleigh()
plot(density(diff(df$dim1)))
plot(df$dim1, df$dim2, type = "l")
```

---

getColours	<i>Get some nice colours</i>
------------	------------------------------

---

### Description

Get some nice colours

### Usage

```
getColours(pal = c("pe", "mm", "le", "an")[1], Ncols = 20)
```

### Arguments

pal	The colour palette, one of "pe", "mm", "le" or "an" (default = "pe")
Ncols	Number of colours

### Value

A list of colours

### Examples

```
getColours(Ncol=5)
```

---

get_os	<i>Which OS is running?</i>
--------	-----------------------------

---

### Description

Some systems not tested, but based on the cran page: [check flavors](#)

### Usage

```
get_os()
```

### Value

A string, "osx", "windows", "linux"

---

*gg\_plotHolder**gg\_plotHolder*

---

**Description***gg\_plotHolder***Usage***gg\_plotHolder()***Value**A blank *ggplot2* object that can be used in concordance with *grid.arrange*.**Examples**

```
# Create a plot with marginal distributions.
library(ggplot2)
library(scales)

df <- data.frame(x = rnorm(n = 100),
                 y = rnorm(n = 100),
                 group = factor(sample(x=c(0,1),
                                     size = 100, replace = TRUE)))

scatterP <- ggplot(df, aes(x = x, y = y, colour = group)) +
  geom_point() +
  gg_theme()

xDense <- ggplot(df, aes(x = x, fill = group)) +
  geom_density(aes(y= ..count..), trim=FALSE, alpha=.5) +
  gg_theme("noax") +
  theme(legend.position = "none")

yDense <- ggplot(df, aes(x = y, fill = group)) +
  geom_density(aes(y= ..count..), trim=FALSE, alpha=.5) +
  coord_flip() +
  gg_theme("noax") +
  theme(legend.position = "none")

library(gridExtra)
grid.arrange(xDense,
             gg_plotHolder(),
             scatterP,
             yDense,
             ncol=2, nrow=2,
             widths=c(4, 1.4),
             heights=c(1.4, 4))
```

gg\_theme

*gg\_theme***Description**

gg\_theme

**Usage**

```
gg_theme(type = c("clean", "noax"))
```

**Arguments**

type                      One of "clean", or "noax"

**Details**

Will generate a "clean" ggplot theme, or a theme without any axes ("noax").

Some scientific journals explicitly request the Arial font should be used in figures. This can be achieved by using .afm font format (see, e.g. <http://www.pure-mac.com/font.html>).

**Value**

A theme for ggplot2.

**Examples**

```
library(ggplot2)
g <- ggplot(data.frame(x = rnorm(n = 100), y = rnorm(n = 100)), aes(x = x, y = y)) + geom_point()
g + gg_theme()
g + gg_theme("noax")
```

growth\_ac

*Examples of dynamical growth models (maps)***Description**

Autocatalytic Growth: Iterating differential equations (maps)

**Usage**

```
growth_ac(Y0 = 0.01, r = 1, k = 1, N = 100, type = c("driving",
"damping", "logistic", "vanGeert")[1])
```

**Arguments**

Y0                      Initial value.  
r                        Growth rate parameter.  
k                        Carrying capacity.  
N                        Length of the time series.  
type                    One of: "driving" (default), "damping", "logistic", "vanGeert1991".



*growth\_ac\_cond*

41

## Value

A timeseries object of length N.

## Author(s)

Fred Hasselman

## See Also

Other autocatalytic growth functions: [growth\\_ac\\_cond](#)

## Examples

```
# The logistic map in the chaotic regime
growth_ac(Y0 = 0.01, r = 4, type = "logistic")
```

---

<code>growth_ac_cond</code>	<i>Examples of conditional dynamical growth models (maps)</i>
-----------------------------	---

---

## Description

Conditional Autocatalytic Growth: Iterating differential equations (maps)

## Usage

```
growth_ac_cond(Y0 = 0.01, r = 0.1, k = 2, cond = cbind.data.frame(Y
  = 0.2, par = "r", val = 2), N = 100)
```

## Arguments

<code>Y0</code>	Initial value
<code>r</code>	Growth rate parameter
<code>k</code>	Carrying capacity
<code>cond</code>	Conditional rules passed as a data.frame of the form: <code>cbind.data.frame(Y = ..., par = ..., val = ...)</code>
<code>N</code>	Length of the time series

## Author(s)

Fred Hasselman

## See Also

Other autocatalytic growth functions: [growth\\_ac](#)

**Examples**

```
# Plot with the default settings
library(lattice)
xyplot(growth_ac_cond())

# The function can take a set of conditional rules
# and apply them sequentially during the iterations.
# The conditional rules are passed as a `data.frame`

(cond <- cbind.data.frame(Y = c(0.2, 0.6), par = c("r", "r"), val = c(0.5, 0.1)))
xyplot(growth_ac_cond(cond=cond))

# Combine a change of `r` and a change of `k`

(cond <- cbind.data.frame(Y = c(0.2, 1.99), par = c("r", "k"), val = c(0.5, 3)))
xyplot(growth_ac_cond(cond=cond))

# A fantasy growth process

cond <- cbind.data.frame(Y = c(0.1, 1.99, 1.999, 2.5, 2.9),
  par = c("r", "k", "r", "r", "k"),
  val = c(0.3, 3, 0.9, 0.1, 1.3))

xyplot(growth_ac_cond(cond=cond))
```

---

layout_as_spiral	<i>Layout a graph on a spiral</i>
------------------	-----------------------------------

---

**Description**

Layout a graph on a spiral

**Usage**

```
layout_as_spiral(g, type = c("Archimedean", "Bernoulli", "Fermat",
  "Euler"), arcs = 6, a = 1, b = NULL, rev = FALSE)
```

**Arguments**

<b>g</b>	An igraph object. If (rev = FALSE) the vertex with the lowest index will be placed in the centre of the spiral, the highest index will be most outer vertex,
<b>type</b>	Spiral type, one of "Archimedean", "Bernoulli", "Fermat", or "Euler" (default = "Archimedean")
<b>arcs</b>	The number of arcs (half circles/ovals) that make up the spiral (default = 10)
<b>a</b>	Parameter controlling the distance between spiral arms, however, the effect will vary for different spiral types (default = 0.5)
<b>b</b>	Parameter controlling where the spiral originates. A value of 1 will generally place the origin in the center. The default NULL will choose a value based on the different spiral types (default = NULL)
<b>rev</b>	If TRUE the vertex with the highest index will be placed in the centre of the spiral (default = FALSE)

*make\_spiral\_focus*

43

**Value**

An igraph layout

**Examples**

```
library(igraph)

g <- igraph::sample_gnp(100, 1/100)

# Equiangular spiral: Any line from the origin cuts at the same angle.
plot(g, layout = layout_as_spiral(g, type = "Bernoulli", arcs = 5))

# The arms of Fermat's spiral diverge quadratically.
plot(g, layout = layout_as_spiral(g, type = "Fermat", arcs = 5))

# Equidistance of intersection points along a line through the origin.
plot(g, layout = layout_as_spiral(g, type = "Archimedean", arcs = 5))
```

---

<code>make_spiral_focus</code>	<i>Spiral Graph with Epoch Focus</i>
--------------------------------	--------------------------------------

---

**Description**

Turn an [igraph](#) object into a spiral graph returning a [ggplot2](#) object.

**Usage**

```
make_spiral_focus(g, arcs = 6, a = 1, b = NULL, rev = FALSE,
  curvature = -0.6, angle = 90, markTimeBy = NULL, alphaV = 1,
  alphaE = 0.6, title = "", subtitle = "", showEpochLegend = TRUE,
  markEpochsBy = NULL, epochColours = NULL, epochLabel = "Epoch",
  showSizeLegend = FALSE, sizeLabel = "Size", scaleVertexSize = c(1,
  6), vertexBorderColour = "black", scaleEdgeSize = 1/5,
  defaultEdgeColour = "grey70", doPlot = TRUE)
```

**Arguments**

<code>g</code>	An igraph object. If ( <code>rev = FALSE</code> ) the vertex with the lowest index will be placed in the centre of the spiral, the highest index will be most outer vertex,
<code>arcs</code>	The number of arcs (half circles/ovals) that make up the spiral (default = 10)
<code>a</code>	Parameter controlling the distance between spiral arms, however, the effect will vary for different spiral types (default = 0.5)
<code>b</code>	Parameter controlling where the spiral originates. A value of 1 will generally place the origin in the center. The default NULL will choose a value based on the different spiral types (default = NULL)
<code>rev</code>	If TRUE the vertex with the highest index will be placed in the centre of the spiral (default = FALSE)
<code>curvature</code>	The curvature parameter for edges see <a href="#">geom_curve()</a> (default = -0.7)

angle	The angle parameter for edges see <a href="#">geom_curve()</a> (default = 90)
markTimeBy	Include a vector that indicates time. The time will be displayed on the plot. Pass TRUE to generate auto labels (experimental)
title	A title for the plot
subtitle	A subtitle for the plot
showEpochLegend	Should a legend be shown for the epoch colours? (default = TRUE)
markEpochsBy	A vector of length <code>vcount(g)</code> indicating epochs or groups (default = NULL)
epochColours	A vector of length <code>vcount(g)</code> with colour codes (default = NULL)
epochLabel	A title for the epoch legend (default = "Epoch")
showSizeLegend	Should a legend be shown for the size of the nodes? (default = FALSE)
sizeLabel	Use to indicate if <code>V(g)\$size</code> represents some measure, e.g. <a href="#">igraph::degree()</a> , or, <a href="#">igraph::hubscore()</a> (default = "Size")
scaleVertexSize	Scale the size of the vertices by setting a range for <a href="#">ggplot2::scale_size()</a> . This will not affect the numbers on the size legend (default = <code>c(1, 6)</code> )
vertexBorderColour	Draw a border around the vertices. Pass NULL to use the same colour as the fill colour (default = "black")
scaleEdgeSize	Scale the size of the edges by a constant: <code>E(g)\$width * scaleEdgeSize</code> (default = 1/5)
defaultEdgeColour	Colour of edges that do not connect to the same epoch (default = "grey70")
doPlot	Produce a plot? (default = TRUE)

**Value**

A ggplot object.

**Note**

To keep the igraph object, use the layout function [layout\\_as\\_spiral\(g\)](#) when plotting the graph.

**Examples**

```
library(igraph)
g <- sample_gnp(200, 1/20)
V(g)$size <- degree(g)
make_spiral_graph(g, markTimeBy = TRUE, showSizeLegend = TRUE, sizeLabel = "Node degree")
```

---

make_spiral_graph	<i>Make Spiral Graph</i>
-------------------	--------------------------

---

## Description

Turn an [igraph](#) object into a spiral graph returning a [ggplot2](#) object.

## Usage

```
make_spiral_graph(g, type = "Archimedean", arcs = 6, a = 1,
  b = NULL, rev = FALSE, curvature = -0.6, angle = 90,
  markTimeBy = NULL, alphaV = 1, alphaE = 0.6, title = "",
  subtitle = "", showEpochLegend = TRUE, markEpochsBy = NULL,
  epochColours = NULL, epochLabel = "Epoch", showSizeLegend = FALSE,
  sizeLabel = "Size", scaleVertexSize = c(1, 6),
  vertexBorderColour = "black", scaleEdgeSize = 1/5,
  defaultEdgeColour = "grey70", doPlot = TRUE)
```

## Arguments

<code>g</code>	An <a href="#">igraph</a> object. If ( <code>rev = FALSE</code> ) the vertex with the lowest index will be placed in the centre of the spiral, the highest index will be most outer vertex,
<code>type</code>	Spiral type, one of "Archimedean", "Bernoulli", "Fermat", or, "Euler" (default = "Archimedean")
<code>arcs</code>	The number of arcs (half circles/ovals) that make up the spiral (default = 10)
<code>a</code>	Parameter controlling the distance between spiral arms, however, the effect will vary for different spiral types (default = 0.5)
<code>b</code>	Parameter controlling where the spiral originates. A value of 1 will generally place the origin in the center. The default NULL will choose a value based on the different spiral types (default = NULL)
<code>rev</code>	If TRUE the vertex with the highest index will be placed in the centre of the spiral (default = FALSE)
<code>curvature</code>	The curvature parameter for edges see <a href="#">geom_curve()</a> (default = -0.7)
<code>angle</code>	The angle parameter for edges see <a href="#">geom_curve()</a> (default = 90)
<code>markTimeBy</code>	Include a vector that indicates time. The time will be displayed on the plot. Pass TRUE to generate auto labels (experimental)
<code>title</code>	A title for the plot
<code>subtitle</code>	A subtitle for the plot
<code>showEpochLegend</code>	Should a legend be shown for the epoch colours? (default = TRUE)
<code>markEpochsBy</code>	A vector of length <code>vcount(g)</code> indicating epochs or groups (default = NULL)
<code>epochColours</code>	A vector of length <code>vcount(g)</code> with colour codes (default = NULL)
<code>epochLabel</code>	A title for the epoch legend (default = "Epoch")
<code>showSizeLegend</code>	Should a legend be shown for the size of the nodes? (default = FALSE)
<code>sizeLabel</code>	Use to indicate if <code>V(g)\$size</code> represents some measure, e.g. <a href="#">igraph::degree()</a> , or, <a href="#">igraph::hubscore()</a> (default = "Size")

scaleVertexSize	Scale the size of the vertices by setting a range for <code>ggplot2::scale_size()</code> . This will not affect the numbers on the size legend (default = <code>c(1, 6)</code> )
vertexBorderColour	Draw a border around the vertices. Pass NULL to use the same colour as the fill colour (default = "black")
scaleEdgeSize	Scale the size of the edges by a constant: <code>E(g)\$width * scaleEdgeSize</code> (default = 1/5)
defaultEdgeColour	Colour of edges that do not connect to the same epoch (default = "grey70")
doPlot	Produce a plot? (default = TRUE)

**Value**

A ggplot object.

**Note**

To keep the igraph object, use the layout function `layout_as_spiral(g)` when plotting the graph.

**Examples**

```
library(igraph)

g <- igraph::sample_gnp(200, 1/20)
V(g)$size <- degree(g)
make_spiral_graph(g, markTimeBy = TRUE, showSizeLegend = TRUE, sizeLabel = "Node degree")
```

---

mif

---

*Mutual Information Function*


---

**Description**

Calculate the lagged mutual information function within (auto-mif) or between (cross-mif) time series, or, conditional on another time series (conditional-cross-mif). Alternatively, calculate the total information of a multivariate dataset for different lags.

**Usage**

```
mif(y, lags = -10:10, nbins = ceiling(2 * NROW(y)^(1/3)),
    doPlot = FALSE, surTest = FALSE, alpha = 0.05)
```

**Arguments**

**y** A Nx1 matrix for auto-mif, a Nx2 matrix or data frame for cross-mif, a Nx3 matrix or data frame for mif between col 1 and 2 conditional on col 3; or a NxM matrix or data frame for the multi-information function. Mutual information for each lag will be calculated using functions in package `infotheo::infotheo()` for lags lagged versions of the time series.

lags	The lags to evaluate mutual information.
nbins	The number of bins passed to <code>infotheo::discretize()</code> if <code>y</code> is a matrix or <code>casnet::ts_discrete()</code>
doPlot	Produce a plot of the symbolic time series by calling <code>plotRED_mif()</code> (default = FALSE)
surTest	If TRUE, a surrogate will be conducted using simple surrogates. The surrogates will be created from the transition probabilities of the discretised time series, i.e. the probability of observing bin <code>j</code> when the current value is in bin <code>j</code> . The number of surrogates needed will be computed based on the value of the alpha parameter, conceived as a one-sided test: $\mu_i > 0$ .
alpha	The alpha level for the surrogate test (default = 0.05)

**Value**

The auto- or cross-mi function

**See Also**

Other Redundancy measures (mutual information): `mi_mat`, `mif_interlayer`

**Examples**

```
# Lags to evaluate mutual information
lags <- -10:30

# Auto-mutual information
y1 <- sin(seq(0, 100, by = 1/8)*pi)

(mif(data.frame(y1), lags = lags))

# Cross-mutual information, y2 is a lagged version y1
y2 <- y1[10:801]

y <- data.frame(ts_trimfill(y1, y2, action = "trim.cut"))
(mif(y, lags = lags))

# Conditional mutual information, add some noise to y2 and add it as a 3rd column
y$s <- y2+rnorm(NROW(y2))
(mif(y, lags = lags))

# Multi-information, the information of the entire multivariate series at each lag
y$y3 <- cumsum(rnorm(NROW(y)))
(mif(y, lags = lags))
```

---

mif\_interlayer

---

*Inter-layer mutual information*


---

**Description**

Inter-layer mutual information

**Usage**

```
mif_interlayer(g0, g1, probTable = FALSE)
```

**Arguments**

- `g0` An igraph object representing a layer in a multiplex graph
- `g1` An igraph object representing a layer in a multiplex graph
- `probTable` Option to return the table with marginal and joint degree distribution probabilities (default = TRUE)

**Value**

The inter-layer mutual information between `g1` and `g2`. If `probTable=TRUE`, a list object with two fields, the inter-layer mutual information and the table with marginal and joint degree distributions

**See Also**

Other Redundancy measures (mutual information): [mi\\_mat](#), [mif](#)

---

<code>mi_mat</code>	<i>Mutual Information variations</i>
---------------------	--------------------------------------

---

**Description**

Mutual Information variations

**Usage**

```
mi_mat(y, ID1, ID2, discreteBins = ceiling(2 * NROW(ID1)^(1/3)))
```

**Arguments**

- `y` A matrix with time series in columns
- `ID1` ids
- `ID2` ids
- `discreteBins` Number of bins to use when discretizing the time series

**Value**

mi in nats

**See Also**

Other Redundancy measures (mutual information): [mif\\_interlayer](#), [mif](#)



*noise\_fBm*

49

---

*noise\_fBm*

*Generate fractional Brownian motion*

---

### Description

Generate fractional Brownian motion

### Usage

```
noise_fBm(H = 1.5, N = 100, mu = NULL, sigma = NULL)
```

### Arguments

H	Hurst exponent
N	Length of noise series
mu	Mean
sigma	SD

### Value

fBm

---

*noise\_fGn*

*Generate fractional Gaussian noise*

---

### Description

Generate fractional Gaussian noise

### Usage

```
noise_fGn(H = 0.5, N = 100, mu = NULL, sigma = NULL)
```

### Arguments

H	Hurst exponent
N	Length of noise series
mu	Mean
sigma	SD

### Value

fGn

---

noise_powerlaw	<i>Generate noise series with power law scaling exponent</i>
----------------	--

---

**Description**

Generate noise series with power law scaling exponent

**Usage**

```
noise_powerlaw(y = NULL, alpha = -1, N = 100, standardise = FALSE,
               randomPower = FALSE, seed = NA)
```

**Arguments**

y	Time series to use as a 'model'. If specified, N will be $N = \text{length}(y)$ , and the series will be constructed based on <code>stats::fft(y)</code> .
alpha	The log-log spectral slope, the scaling exponent. Use 0 for white noise, negative numbers for anti-persistent noises: -1 for $\frac{1}{f}$ noise, positive numbers for persistent noises, e.g. 1 for blue noise.
N	Length of the time series
standardise	Forces scaling of the output to the range $[-1, 1]$ , consequently the power law will not necessarily extend right down to 0Hz.
randomPower	If TRUE phases will be deterministic, uniformly distributed in $[-\pi, \pi]$ . If FALSE, the spectrum will be stochastic with a Chi-square distribution. If y is not NULL this argument will be ignored.
seed	Provide an integer number to set the seed for the random number generator in order to get reproducible results. If NA (default) no user defined seed will be set,

**Value**

Time series with a power law of alpha.

**Note**

Adapted from a Matlab script called `powernoise.m` by Max Little. The script contained the following commented text:

With no option strings specified, the power spectrum is

---

plotDC_ccp	<i>Plot Cumulative Complexity Peaks</i>
------------	---

---

**Description**

Plot Cumulative Complexity Peaks

**Usage**

```
plotDC_ccp(df_ccp, win, useVarNames = TRUE, colOrder = TRUE,
  useTimeVector = NA, timeStamp = "31-01-1999", doPlot = TRUE,
  title = "Critical Instability Plot", subtitle = "",
  xlabel = "Time", ylabel = "")
```

**Arguments**

df_ccp	A dataframe generated by dc_ccp()
useVarNames	Use the column names of df as variable names in the Complexity Resonance Diagram (default = TRUE)
colOrder	If TRUE, the order of the columns in df determines the of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by by mean value of Dynamic Complexity (default = TRUE)
useTimeVector	Parameter used for plotting. A vector of length NROW(df), containing date/time information (default = NA)
timeStamp	If useTimeVector is not NA, a character string that can be passed to <code>lubridate::stamp()</code> to format the the dates/times passed in useTimeVector (default = "01-01-1999")
doPlot	If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE)
title	A title for the plot.
subtitle	A subtitle for the plot.
xlabel	A label for the x-axis.
ylabel	A label for the y-axis.

**Value**

An invisible ggplot2 object.

**See Also**

Other Dynamic Complexity functions: `dc_ccp`, `dc_d`, `dc_f`, `dc_win`, `plotDC_lvl`, `plotDC_res`

---

plotDC_lvl	<i>Plot Peaks versus Levels</i>
------------	---------------------------------

---

**Description**

Produce a plot in which the output of `dc_win()` and `dc_ccp()` on the same multivariate timeseries data is combined with the output of `ts_level()` on a state variable of the same length as the multivariate data.

**Usage**

```
plotDC_lvl(df_win, df_ccp, df_lvl, win, useVarNames = TRUE,
  colOrder = TRUE, useTimeVector = NA, timeStamp = "31-01-1999",
  doPlot = TRUE, title = "Peaks versus Levels Plot", subtitle = "",
  xlabel = "Time", ylabel = "", levelName = "State variable")
```

**Arguments**

df_win	A data frame containing series of Dynamic Complexity values obtained by running function <code>dc_win()</code>
df_ccp	A dataframe generated by <code>dc_ccp()</code>
df_lv1	A dataframe generated by <code>ts_level()</code> of a variable that is considered a state variable.
useVarNames	Use the column names of df as variable names in the Complexity Resonance Diagram (default = TRUE)
colOrder	If TRUE, the order of the columns in df determines the of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by by mean value of Dynamic Complexity (default = TRUE)
useTimeVector	Parameter used for plotting. A vector of length <code>NROW(df)</code> , containing date/time information (default = NA)
timeStamp	If useTimeVector is not NA, a character string that can be passed to <code>lubridate::stamp()</code> to format the the dates/times passed in useTimeVector (default = "01-01-1999")
doPlot	If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE)
title	A title for the plot.
subtitle	A subtitle for the plot.
xlabel	A label for the x-axis.
ylabel	A label for the y-axis.
levelName	A name for the state variable.

**Value**

An invisible ggplot2 object.

**See Also**

Other Dynamic Complexity functions: `dc_ccp`, `dc_d`, `dc_f`, `dc_win`, `plotDC_ccp`, `plotDC_res`

---

plotDC\_res

---

*Plot Complexity Resonance Diagram*


---

**Description**

Plot Complexity Resonance Diagram

**Usage**

```
plotDC_res(df_win, win, useVarNames = TRUE, colOrder = TRUE,
  useTimeVector = NA, timeStamp = "01-01-1999", doPlot = TRUE,
  title = "Complexity Resonance Diagram", subtitle = "",
  xlabel = "Time", ylabel = "")
```

**Arguments**

df_win	A data frame containing series of Dynamic Complexity values obtained by running function <code>dc_win()</code>
useVarNames	Use the column names of df as variable names in the Complexity Resonance Diagram (default = TRUE)
colOrder	If TRUE, the order of the columns in df determines the of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by mean value of Dynamic Complexity (default = TRUE)
useTimeVector	Parameter used for plotting. A vector of length <code>NROW(df)</code> , containing date/time information (default = NA)
timeStamp	If useTimeVector is not NA, a character string that can be passed to <code>lubridate::stamp()</code> to format the the dates/times passed in useTimeVector (default = "01-01-1999")
doPlot	If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE)
title	A title for the plot.
subtitle	A subtitle for the plot.
xlabel	A label for the x-axis.
ylabel	A label for the y-axis.

**Value**

An invisible ggplot2 object.

**See Also**

Other Dynamic Complexity functions: `dc_ccp`, `dc_d`, `dc_f`, `dc_win`, `plotDC_ccp`, `plotDC_lvl`

---

plotFD_loglog	<i>Plot output from fluctuation analyses based on log-log regression</i>
---------------	--

---

**Description**

Plot output from fluctuation analyses based on log-log regression

**Usage**

```
plotFD_loglog(fd.OUT, title = "", subtitle = "", xlabel = "Bin size",
  ylabel = "Fluctuation", logBase = NA)
```

**Arguments**

fd.OUT	Output from one of the fd_ functions that use log-log regression to get scaling exponents.
title	Plot title
subtitle	Plot subtitle
xlabel	x label
ylabel	y label
logBase	base of the log used

54

*plotNET\_groupColour*

### Value

A ggplot object

---

plotNET_BA	<i>Example of Barabasi scale-free network</i>
------------	---

---

### Description

A wrapper around [igraph::sample\\_pa\(\)](#)

### Usage

```
plotNET_BA(n = 100, pwr = 1, out.dist = NULL, doPlot = TRUE)
```

### Arguments

n	Number of vertices
pwr	Power of preferential attachment
out.dist	Degree distribution
doPlot	Plot the igraph object

### Value

A Barabasi scale-free igraph object

### See Also

[igraph::sample\\_pa\(\)](#)

Other tools for plotting networks: [plotNET\\_SW](#), [plotNET\\_groupColour](#), [plotNET\\_groupWeight](#), [plotNET\\_prep](#)

---

plotNET_groupColour	<i>Vertex Group Colours</i>
---------------------	-----------------------------

---

### Description

Identify Vertex and/or Edge groups by colour.

### Usage

```
plotNET_groupColour(g, groups, colourV = TRUE, alphaV = 1,
  colourE = FALSE, alphaE = 0.8, groupColours = NULL,
  defaultEdgeColour = "grey70", doPlot = TRUE)
```

**Arguments**

<code>g</code>	An igraph object
<code>groups</code>	A named numeric vector with <code>length(V(g))</code> integers representing each group, or, a named character vector describing each group. If <code>names(groups)==NULL</code> then the names of the vector will be set as <code>names(groups) == V(g)\$name</code> . If <code>V(g)\$name==NULL</code> , the names of the vector will be set by the Vertex index
<code>colourV</code>	Colour Vertices based on groups (default = TRUE)
<code>alphaV</code>	Set transparency for Vertices (default = 1)
<code>colourE</code>	Colour Edges based on groups. Edges connecting to vertices of the same group will be coloured as the group (default = FALSE)
<code>alphaE</code>	Set transparency for Edges. A single numeric, or a vector of length <code>ecount(g)</code> (default = 0.8)
<code>groupColours</code>	A list of length groups with valid colour codes
<code>defaultEdgeColour</code>	Default edge colour
<code>doPlot</code>	Plot the igraph object

**Value**

An igraph object with vertices and/or edges coloured by groups listed in groups

**See Also**

Other tools for plotting networks: [plotNET\\_BA](#), [plotNET\\_SW](#), [plotNET\\_groupWeight](#), [plotNET\\_prep](#)

---

`plotNET_groupWeight`     *Set Edge weights by group*

---

**Description**

Use a layout which takes a weights

**Usage**

```
plotNET_groupWeight(g, groups, weighth.within = 100, weight.between = 1,
  preserve.weight.within = FALSE, preserve.weight.between = FALSE,
  doPlot = FALSE, returnOnlyWeights = TRUE)
```

**Arguments**

<code>g</code>	An igraph object whose edges ( <code>get.edgelist(g)</code> ) will be re-weighted according to the membership argument.
<code>groups</code>	A named numeric vector with <code>length(V(g))</code> integers representing each group, or, a named character vector describing each group. If <code>names(groups)==NULL</code> then the names of the vector will be set as <code>names(groups) == V(g)\$name</code> . If <code>V(g)\$name==NULL</code> , the names of the vector will be set by the Vertex index
<code>weighth.within</code>	The weight within a group (default = 100)
<code>weight.between</code>	The weight within a group (default = 1)

```

preserve.weight.within      If E(g)$weights is not NULL, try to preserve edge weights within a group
preserve.weight.between    If E(g)$weights is not NULL, try to preserve edge weights between a groups
doPlot                      Plot the igraph object
returnOnlyWeights           Do not return the graph, just the weights. If FALSE this will return the graph
                           object, otherwise it returns E(g)$weights

```

**Value**

A numeric vector with `length(get.edgelist(g))` edge weights that will cluster groups defined in membership if a layout is used that can handle edge weights as a parameter (see examples).

**See Also**

Other tools for plotting networks: [plotNET\\_BA](#), [plotNET\\_SW](#), [plotNET\\_groupColour](#), [plotNET\\_prep](#)

**Examples**

```

# Make a star graph and let the odd numbers cluster together
library(igraph)
g <- make_full_graph(10, directed=FALSE)
E(g)$width <- 3
V(g)$name <- paste(1:10)
membership <- rep(c(1,2),5)
names(membership) <- V(g)$name
E(g)$weight <- plotNET_groupWeight(g,membership,1000,10)
g$layout=layout.fruchterman.reingold(g,weights=E(g)$weight)
plot(g)

# Make 3 groups by changing the 'membership' vector
membership[3:6] <- 3
names(membership) <- V(g)$name
E(g)$weight <- plotNET_groupWeight(g,membership,1000,10)
g$layout=layout.fruchterman.reingold(g,weights=E(g)$weight)
plot(g)

# Use plotNET_groupColour for Vertex and Edge group colours
g <- plotNET_groupColour(g, membership, colourE=TRUE)
plot(g)

```

---

plotNET\_prep

---

*Plot Network Based on RQA*


---

**Description**

Plot Network Based on RQA



**Usage**

```
plotNET_prep(g, labels = NA, nodesize = c("degree", "hubscore",
  "strength", "eccentricity", "coreness")[1], labelsizesize = "asnodesize",
  edgeweight = "weight", removeZeroDegree = TRUE,
  removeSelfLoops = TRUE, doPlot = TRUE)
```

**Arguments**

<code>g</code>	An igraph object
<code>labels</code>	Vertex labels
<code>nodesize</code>	Set nodesizes by degree( <code>g</code> , <code>normalised = TRUE</code> ) (default), hubscore( <code>g</code> )\$vector, or, strength( <code>g</code> ), eccentricity( <code>g</code> ), coreness( <code>g</code> ). If a numeric value is passed all vertex sizes will be set to that value.
<code>labelsizesize</code>	Set labelsizesize: "asnodesize" sets the cex for the labels to coincide with nodesize (with min of .4 and max of 1.1). A single numeric value sets the cex of all labels to that value. A numeric vector of length two, c(min,max) wil scale the node sizes to min and max which
<code>edgeweight</code>	Set size of edges to "E( <code>g</code> )\$weight" by passing "weight". If a single numeric value is provided all edges will be set to that value.
<code>removeZeroDegree</code>	Remove vertices with degree( <code>g</code> ) == 0 (default = TRUE)
<code>removeSelfLoops</code>	Calls simplify( <code>g</code> ) (default = TRUE)
<code>doPlot</code>	Plot the igraph object.

**Value**

an igraph object

**See Also**

Other tools for plotting networks: [plotNET\\_BA](#), [plotNET\\_SW](#), [plotNET\\_groupColour](#), [plotNET\\_groupWeight](#)

---

plotNET\_SW

*Example of Strogatz-Watts small-world network*

---

**Description**

A wrapper around `igraph::sample_smallworld()` with `dim=1`

**Usage**

```
plotNET_SW(n = 100, k = 5, p = 0.05, doPlot = TRUE)
```

**Arguments**

<code>n</code>	Size of the lattice (integer)
<code>k</code>	Neighbourhood size (integer)
<code>p</code>	Rewiring probability (between 0 and 1)
<code>doPlot</code>	PLot the igraph object

**Value**

A Strogatz-Watts small-world igraph object

**See Also**

[igraph::sample\\_smallworld\(\)](#)

Other tools for plotting networks: [plotNET\\_BA](#), [plotNET\\_groupColour](#), [plotNET\\_groupWeight](#), [plotNET\\_prep](#)

---

plotRED_acf	<i>Plot ACF and PACF</i>
-------------	--------------------------

---

**Description**

Plot ACF and PACF

**Usage**

```
plotRED_acf(y, Lmax = max(round(NROW(y)/4), 10), alpha = 0.05,
  doPlot = TRUE, returnCorFun = FALSE)
```

**Arguments**

y	A time series or numeric vector
Lmax	Maximum number of lags
alpha	Significance level
doPlot	Plot output
returnCorFun	Return the data

**Value**

Either an invisible ggplot2 object or a list containing the plot and the data

**See Also**

Other Plot redundancy functions: [plotRED\\_mif](#)

---

plotRED_mif	<i>Plot various MI functions</i>
-------------	----------------------------------

---

**Description**

Plot various MI functions

**Usage**

```
plotRED_mif(mif.OUT = NULL, lags = 0:max(round(NROW(y)/4), 10),
  nbins = ceiling(2 * NROW(y)^(1/3)), surTest = FALSE, alpha = 0.05,
  doPlot = TRUE, returnMIFun = TRUE)
```

**Arguments**

mif.OUT	Output from function <code>mif()</code>
lags	The lags to evaluate mutual information.
nbins	The number of bins passed to <code>infotheo::discretize()</code> if y is a matrix or <code>casnet::ts_discrete()</code>
surTest	If TRUE, a surrogate will be conducted using simple surrogates. The surrogates will be created from the transition probabilities of the discretised time series, i.e. the probability of observing bin j when the current value is in bin j. The number of surrogates needed will be computed based on the value of the alpha parameter, conceived as a one-sided test: $mi > 0$ .
alpha	The alpha level for the surrogate test (default = 0.05)
doPlot	Produce a plot of the symbolic time series by calling <code>plotRED_mif()</code> (default = FALSE)
returnMIFun	Return the data

**Value**

Either an invisible ggplot2 object or a list containing the plot and the data

**See Also**

Other Plot redundancy functions: `plotRED_acf`

---

plotSUR_hist	<i>Surrogate Test</i>
--------------	-----------------------

---

**Description**

Surrogate Test

**Usage**

```
plotSUR_hist(surrogateValues, observedValue, sides = c("two.sided",
  "greater", "less")[1], binWidth = NULL, measureName = "",
  title = "", doPlot = TRUE, returnOnlyPvalue = FALSE)
```

**Arguments**

surrogateValues	Vector of measures based on surrogate time series
observedValue	The measure obtained from the observed value
sides	Is this a 1 or 2-sided test (default = 1)
binWidth	The size of the histogram bins. The default is to look for the max. number of digits and set the width to $1/10^{(Ndigits-1)}$ . If integers are detected width will be set to 1.
measureName	Label for x-axis
title	A title for the plot
doPlot	Plot a histogram of the distribution (default = TRUE)
returnOnlyPvalue	Do not return the graph, just the point p-value (default = FALSE)
	alpha Significance threshold for the test. This value is currently calculated from the data as $\frac{1}{rank} * Nsides$ , setting it will not have an effect.

**Value**

A point p-value for the observed value, and/or a histogram of the distribution (ggplot2 object).

---

plotTS_multi	<i>Plot Multivariate Time Series Data</i>
--------------	---

---

**Description**

Plot Multivariate Time Series Data

**Usage**

```
plotTS_multi(df, timeVec = NA, groupVec = NA, useVarNames = TRUE,
  colOrder = TRUE, doPlot = TRUE, title = "", subtitle = "",
  xlabel = "Time", ylabel = "", returnPlotData = FALSE,
  useRibbon = FALSE, overlap = 1)
```

**Arguments**

df	A data frame with time series in columns.
timeVec	If numeric, the number of the column in df which contains a time=keeping variable. If NA, the time vector will be 1:NROW(df) (default = NA)
groupVec	A vector indicating the names of the time series in the columns of df. If NA, the column names of df will be used, excluding the timeVec, if present. (default = NA)
useVarNames	Use the column names of df as variable names in the Complexity Resonance Diagram (default = TRUE)
colOrder	If TRUE, the order of the columns in df determines the of variables on the y-axis. Use FALSE for alphabetic/numeric order. Use NA to sort by by mean value of Dynamic Complexity (default = TRUE)

*repmat*

61

<code>doPlot</code>	If TRUE shows a Complexity Resonance Diagram of the Dynamic Complexity and returns an invisible <code>ggplot2::ggplot()</code> object. (default = FALSE)
<code>title</code>	A title for the plot.
<code>subtitle</code>	A subtitle for the plot.
<code>xlabel</code>	A label for the x-axis.
<code>ylabel</code>	A label for the y-axis.
<code>returnPlotData</code>	Return the restructured data frame used to create the plot (default = FALSE)
<code>useRibbon</code>	Neat for distributions
<code>overlap</code>	Multiplier for scaling the series around the y-offset. Default is <code>offset + elascerc(y, lo = -.45*overlap, hi = .45*overlap)</code> and if <code>useRibbon = TRUE</code> it is <code>offset + elascerc(y, lo = 0*overlap, hi = .95*overlap)</code> . (default = 1)

**Value**A `ggplot` object.**Examples**

```
# Generate some coloured noise
N <- 512
noises <- seq(-3,3,by=.5)
y <- data.frame(matrix(rep(NA,length(noises)*N), ncol=length(noises)))

for(c in seq_along(noises)){
  y[,c] <- noise_powerlaw(N=N, alpha = noises[c])
}
colnames(y) <- paste0(noises)

plotTS_multi(y)
```

---

<code>repmat</code>	<i>Repeat Copies of a Matrix</i>
---------------------	----------------------------------

---

**Description**

Repeat Copies of a Matrix

**Usage**

```
repmat(X, m, n)
```

**Arguments**

<code>X</code>	A matrix
<code>m</code>	Multiply <code>dim(X)[1]</code> m times
<code>n</code>	Multiply <code>dim(X)[2]</code> n times

**Value**

A repeated matrix

rn

*Create a Recurrence Network Matrix***Description**

This function serves as a wrapper for function `rp()`, it will add some attributes to the matrix related to network representation. These attributes will be used to decide which network type to generate (e.g. undirected, directed, weighted, etc.)

**Usage**

```
rn(y1, y2 = NULL, emDim = 1, emLag = 1, emRad = NULL,
   directed = FALSE, weighted = FALSE, includeDiagonal = FALSE,
   to.ts = NULL, order.by = NULL, to.sparse = FALSE,
   method = "Euclidean", targetValue = 0.05, returnGraph = TRUE,
   doPlot = FALSE, silent = TRUE, ...)
```

**Arguments**

<code>y1</code>	A numeric vector or time series
<code>y2</code>	A numeric vector or time series for cross recurrence
<code>emDim</code>	The embedding dimensions
<code>emLag</code>	The embedding lag
<code>emRad</code>	The threshold ( <code>emRad</code> ) to apply to the distance matrix to create a binary or weighted matrix. If <code>NULL</code> , an unthresholded matrix will be created (default = <code>NULL</code> )
<code>directed</code>	Should the matrix be considered to represent a directed network? (default = <code>FALSE</code> )
<code>weighted</code>	Should the matrix be considered to represent a weighted network? (default = <code>FALSE</code> )
<code>includeDiagonal</code>	Should the diagonal of the matrix be included when creating the network (default = <code>FALSE</code> )
<code>to.ts</code>	Should <code>y1</code> and <code>y2</code> be converted to time series objects?
<code>order.by</code>	If <code>to.ts = TRUE</code> , pass a vector of the same length as <code>y1</code> and <code>y2</code> . It will be used as the time index, if <code>NA</code> the vector indices will be used to represent time.
<code>to.sparse</code>	Should sparse matrices be used?
<code>method</code>	Distance measure to use. Any option that is valid for argument <code>method</code> of <code>proxy::dist()</code> . Type <code>proxy::pr_DB\$get_entries()</code> to see a list of all the options. Common methods are: "Euclidean", "Manhattan", "Minkowski", "Chebyshev" (or the same but shorter: "L2", "L1", "Lp" and "max" distance) (default = "Euclidean")
<code>targetValue</code>	A value passed to <code>crqa_radius(..., type="fixed", targetMeasure="RR")</code> if <code>is.na(emRad)==TRUE</code> .
<code>returnGraph</code>	Return an <code>igraph::igraph()</code> object (default = <code>TRUE</code> )
<code>doPlot</code>	Plot the matrix by calling <code>rp_plot()</code> with default settings
<code>silent</code>	Silent-ish mode
<code>...</code>	Any paramters to pass to <code>rn_plot()</code> if <code>doPlot = TRUE</code>

RNG

63

### Value

A (Coss-) Recurrence matrix that can be interpreted as an adjacency (or incidence) matrix.

### See Also

Other Distance matrix operations (recurrence network): [di2bi](#), [di2we](#), [rn\\_plot](#), [rn\\_recSpec](#), [rn\\_scaleoGram](#)

---

RNG

*Random Number Sequences*

---

### Description

A dataset containing sequences of 100 numbers generated by 242 participants who were instructed to generate random sequences.

### Usage

RNG

### Format

A data frame with 24200 rows and 3 variables:

**ID** Participant ID

**time** Temporal order

**number** A number between 1 and 9

### Source

<https://www.frontiersin.org/articles/10.3389/fnhum.2015.00319/full>

### References

Oomens, W., Maes, J. H., Hasselmann, F., & Egger, J. I. (2015). A time series approach to random number generation: using recurrence quantification analysis to capture executive behavior. *Frontiers in human neuroscience*, 9

rn\_plot

*Plot (thresholded) distance matrix as a network***Description**

Plot (thresholded) distance matrix as a network

**Usage**

```
rn_plot(RN, plotDimensions = FALSE, plotMeasures = FALSE,
        drawGrid = FALSE, markEpochsLOI = NULL, Chromatic = NULL,
        radiusValue = NA, title = "", xlab = "", ylab = "",
        plotSurrogate = NA, returnOnlyObject = FALSE)
```

**Arguments**

RN	A distance matrix or recurrence matrix
plotDimensions	Should the state vectors be plotted if they are available as attributes of RM (default = TRUE)
plotMeasures	Print common (C)RQA measures in the plot if the matrix is binary
drawGrid	Draw a grid on the recurrence plot (default = FALSE)
markEpochsLOI	Pass a factor whose levels indicate different epochs or phases in the time series and use the line of identity to represent the levels by different colours (default = NULL)
Chromatic	If TRUE and there are more than two discrete values in RM, give recurrent points a distinct colour. If RM was returned by <code>crqa_rp(..., chromatic = TRUE)</code> , the recurrence plot will colour-code recurrent points according to the category values in <code>attributes(RM)\$chromaticRP</code> (default = FALSE)
radiusValue	If <code>plotMeasures = TRUE</code> and RM is an unthresholded matrix, this value will be used to calculate recurrence measures. If <code>plotMeasures = TRUE</code> and RM is already a binary recurrence matrix, pass the radius that was used as a threshold to create the matrix for display purposes. If <code>plotMeasures = TRUE</code> and <code>radiusValue = NA</code> , function <code>crqa_radius()</code> will be called with default settings (find a radius that yields .05 recurrence rate). If <code>plotMeasures = FALSE</code> this setting will be ignored.
title	A title for the plot
plotSurrogate	Should a 2-panel comparison plot based on surrogate time series be added? If RM has attributes <code>y1</code> and <code>y2</code> containing the time series data (i.e. it was created by a call to <code>rp()</code> ), the following options are available: "RS" (random shuffle), "RP" (randomised phases), "AAFT" (amplitude adjusted fourier transform). If no timeseries data is included, the columns will be shuffled. NOTE: This is not a surrogate test, just 1 surrogate is created from <code>y1</code> .
returnOnlyObject	Return the ggplot object only, do not draw the plot (default = TRUE)

**Value**

A nice plot of the recurrence network



*rn\_recSpec*

65

**See Also**

Other Distance matrix operations (recurrence network): [di2bi](#), [di2we](#), [rn\\_recSpec](#), [rn\\_scaleoGram](#), [rn](#)

*rn\_recSpec**Recurrence Time Spectrum***Description**

Get the recurrence time distribution from a recurrence network.

**Usage**

```
rn_recSpec(RN, fitRange = NULL, fs = 1, doPlot = TRUE,
  returnPlot = FALSE, returnPLAW = FALSE, returnInfo = FALSE,
  silent = TRUE, noTitle = FALSE, tsName = "y")
```

**Arguments**

<code>RN</code>	A thresholded recurrence matrix generated by function <code>rn()</code>
<code>fitRange</code>	If <code>NULL</code> the entire range will be used for log-log slope. If a 2-element vector of integers, this will represent the range of recurrence times to use for fitting the log=log slope (e.g. <code>c(1, 50)</code> would fit the first 50 recurrence times).
<code>fs</code>	Sample rate (default = 1)
<code>doPlot</code>	Should a plot of the recurrence time spectrum be produced?
<code>returnPlot</code>	Return <code>ggplot2</code> object (default = <code>FALSE</code> )
<code>returnPLAW</code>	Return the power law data (default = <code>FALSE</code> )
<code>returnInfo</code>	Return all the data used in SDA (default = <code>FALSE</code> )
<code>silent</code>	Silent-ish mode
<code>noTitle</code>	Do not generate a title (only the subtitle)
<code>tsName</code>	Name of y added as a subtitle to the plot

**Value**

A vector of frequencies of recurrence times and a plot (if requested)

**See Also**

Other Distance matrix operations (recurrence network): [di2bi](#), [di2we](#), [rn\\_plot](#), [rn\\_scaleoGram](#), [rn](#)

---

rn_scaleoGram	<i>Recurrence Time Scaleogram</i>
---------------	-----------------------------------

---

**Description**

Display a recurrence network in a space representing Time (x-axis) x Scale (y-axis). The scale axis will be determined by the latency between the occurrence of a value in the (embedded) time series vector and its recurrences in the future (i.e. only the upper triangle of the recurrence matrix will be displayed, excluding the diagonal).

**Usage**

```
rn_scaleoGram(RN, returnOnlyObject = FALSE)
```

**Arguments**

RN	A thresholded recurrence matrix generated by function <code>rn()</code>
returnOnlyObject	Return the ggplot / ggraph object only, do not draw the plot (default = FALSE)

**Value**

A ggraph graph object

**See Also**

Other Distance matrix operations (recurrence network): [di2bi](#), [di2we](#), [rn\\_plot](#), [rn\\_recSpec](#), [rn](#)

---

rp	<i>Create a Distance Matrix</i>
----	---------------------------------

---

**Description**

Create a Distance Matrix

**Usage**

```
rp(y1, y2 = NULL, emDim = 1, emLag = 1, emRad = NULL,
  to.ts = NULL, order.by = NULL, to.sparse = FALSE,
  weighted = FALSE, method = "Euclidean", targetValue = 0.05,
  doPlot = FALSE, silent = TRUE, ...)
```

**Arguments**

y1	A numeric vector or time series
y2	A numeric vector or time series for cross recurrence
emDim	The embedding dimensions
emLag	The embedding lag
emRad	The threshold (emRad) to apply to the distance matrix to create a binary or weighted matrix. If NULL, an unthresholded matrix will be created (default = NULL)
to.ts	Should y1 and y2 be converted to time series objects?
order.by	If to.ts = TRUE, pass a vector of the same length as y1 and y2. It will be used as the time index, if NA the vector indices will be used to represent time.
to.sparse	Should sparse matrices be used?
weighted	If FALSE a binary matrix will be returned. If TRUE every value larger than emRad will be 0, but values smaller than emRad will be retained (default = FALSE)
method	Distance measure to use. Any option that is valid for argument method of <a href="#">proxy::dist()</a> . Type <code>proxy::pr_DB\$get_entries()</code> to see a list of all the options. Common methods are: "Euclidean", "Manhattan", "Minkowski", "Chebyshev" (or the same but shorter: "L2", "L1", "Lp" and "max" distance) (default = "Euclidean")
targetValue	A value passed to <code>crqa_radius(..., type="fixed", targetMeasure="RR")</code> if <code>is.na(emRad)==TRUE</code> .
doPlot	Plot the matrix by calling <a href="#">rp_plot()</a> with default settings
silent	Silent-ish mode
...	Any parameters to pass to <a href="#">rp_plot()</a> if doPlot = TRUE

**Value**

A (Coss-) Recurrence matrix with attributes:

1. `emdims1` and `emdims2` - A matrix of surrogate dimensions
2. `emdims1.name` and `emdims2.name` - Names of surrogate dimensions
3. `method` and `call` - The distance method used by [proxy::dist\(\)](#)
4. `weightd` - Whether a weighted matrix is returned
5. `emDim`, `emLag` and `emRad` - The embedding parameters
6. `AUTO` - Whether the matrix represents AUTO recurrence

**See Also**

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2bi](#), [di2we](#), [dist\\_hamming](#), [rp\\_lineDist](#), [rp\\_nzdiags](#), [rp\\_plot](#), [rp\\_size](#)

---

rp_copy_attributes	<i>Copy Matrix Attributes</i>
--------------------	-------------------------------

---

**Description**

Simple attribute copy used in casnet to convert between matrix and Matrix classes and back.

**Usage**

```
rp_copy_attributes(source, target, source_remove = c("names",
  "row.names", "class", "dim", "dimnames", "x"))
```

**Arguments**

source	Source matrix
target	Target matrix
source_remove	Remove these attribute fields from the source before copying.

**Value**

The target matrix with attributes copied deom the source matrix.

---

rp_lineDist	<i>Line length distributions</i>
-------------	----------------------------------

---

**Description**

Extract lengths of diagonal, vertical and horizontal line segments from a recurrence matrix.

**Usage**

```
rp_lineDist(RM, DLmin = 2, VLmin = 2, HLmin = 2,
  DLmax = length(Matrix::diag(RM)) - 1,
  VLmax = length(Matrix::diag(RM)) - 1,
  HLmax = length(Matrix::diag(RM)) - 1, d = NULL, theiler = NULL,
  invert = FALSE, AUTO = NULL, chromatic = FALSE, matrices = FALSE)
```

**Arguments**

RM	A thresholded recurrence matrix (binary: 0 - 1)
DLmin	Minimal diagonal line length (default = 2)
VLmin	Minimal vertical line length (default = 2)
HLmin	Minimal horizontal line length (default = 2)
DLmax	Maximal diagonal line length (default = length of diagonal -1)
VLmax	Maximal vertical line length (default = length of diagonal -1)
HLmax	Maximal horizontal line length (default = length of diagonal -1)

d	Vector of diagonals to be extracted from matrix RP before line length distributions are calculated. A one element vector will be interpreted as a window size, e.g., d = 50 will extract the diagonal band -50:50. A two element vector will be interpreted as a band, e.g. d = c(-50,100) will extract diagonals -50:100. If length(d) > 2, the numbers will be interpreted to refer to individual diagonals, d = c(-50,50,100) will extract diagonals -50,50,100.
theiler	Size of the theiler window, e.g. theiler = 1 removes diagonal bands -1,0,1 from the matrix. If length(d) is NULL, 1 or 2, the theiler window is applied before diagonals are extracted. The theiler window is ignored if length(d)>2, or if it is larger than the matrix or band indicated by parameter d.
invert	Relevant for Recurrence Time analysis: Return the distribution of 0 valued segments in nonzero diagonals/verticals/horizontals. This indicates the time between subsequent line structures.
AUTO	Is this an AUTO RQA?
chromatic	Chromatic RQA?
matrices	Return the matrices ?

### Details

Based on the Matlab function linedists by Stefan Schinkel, Copyright (C) 2009 Stefan Schinkel, University of Potsdam, <http://www.agnld.uni-potsdam.de>

References: S. Schinkel, N. Marwan, O. Dimigen & J. Kurths (2009): "Confidence Bounds of recurrence-based complexity measures Physics Letters A, 373(26), pp. 2245-2250

Copyright (C) 2009 Stefan Schinkel, University of Potsdam <http://www.agnld.uni-potsdam.de>

### Value

A list object with distributions of line lengths. If matrices = TRUE dataframes are returned whose columns represent the nonzero diagonals, verticals, or, horizontals.

### Author(s)

Fred Hasselmann

### See Also

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2bi](#), [di2we](#), [dist\\_hamming](#), [rp\\_nzdiags](#), [rp\\_plot](#), [rp\\_size](#), [rp](#)

---

rp\_nzdiags

---

rp\_nzdiags

---

### Description

Get all nonzero diagonals of a binary matrix, or, diagonals specified as a vector by argument d.

### Usage

```
rp_nzdiags(RM = NULL, d = NULL, returnVectorList = TRUE,
  returnNZtriplets = FALSE, removeNZ = TRUE, silent = TRUE)
```

**Arguments**

RM	A binary (0,1) matrix.
d	An optional vector of diagonals to extract.
returnVectorList	Return list
returnNZtriplets	Return a dataframe with coordinates of only nonzero elements in diagonals (default = FALSE)
removeNZ	Remove nonzero diagonals if TRUE. If FALSE returns the full diagonals matrix. Use e.g. to plot diagonal recurrence profiles (default = TRUE)
silent	Silent-ish mode

**Value**

A matrix object with nonzero diagonals as columns and/or a dataframe with coordinates of nonzero diagonal elements

**Author(s)**

Fred Hasselman

**See Also**

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2bi](#), [di2we](#), [dist\\_hamming](#), [rp\\_lineDist](#), [rp\\_plot](#), [rp\\_size](#), [rp](#)

---

rp\_plot

---

*Plot (thresholded) distance matrix as a recurrence plot*


---

**Description**

Plot (thresholded) distance matrix as a recurrence plot

**Usage**

```
rp_plot(RM, plotDimensions = FALSE, plotMeasures = FALSE,
        plotRadiusRRbar = TRUE, drawGrid = FALSE, markEpochsLOI = NULL,
        Chromatic = NULL, radiusValue = NA, title = "", xlabel = "",
        ylabel = "", plotSurrogate = NA, returnOnlyObject = FALSE)
```

**Arguments**

RM	A distance matrix or recurrence matrix
plotDimensions	Should the state vectors be plotted if they are available as attributes of RM (default = TRUE)
plotMeasures	Print common (C)RQA measures in the plot if the matrix is binary
plotRadiusRRbar	The Radius-RR-bar is a colour-bar guide plotted with an unthresholded distance matrix indicating a number of RR values one would get if a certain distance threshold were chosen (default = TRUE)

*rp\_size*

71

drawGrid	Draw a grid on the recurrence plot (default = FALSE)
markEpochsLOI	Pass a factor whose levels indicate different epochs or phases in the time series and use the line of identity to represent the levels by different colours (default = NULL)
Chromatic	If TRUE and there are more than two discrete values in RM, give recurrent points a distinct colour. If RM was returned by <code>crqa_rp(..., chromatic = TRUE)</code> , the recurrence plot will colour-code recurrent points according to the category values in <code>attributes(RM)\$chromaticRP</code> (default = FALSE)
radiusValue	If <code>plotMeasures = TRUE</code> and RM is an unthresholded matrix, this value will be used to calculate recurrence measures. If <code>plotMeasures = TRUE</code> and RM is already a binary recurrence matrix, pass the radius that was used as a threshold to create the matrix for display purposes. If <code>plotMeasures = TRUE</code> and <code>radiusValue = NA</code> , function <code>crqa_radius()</code> will be called with default settings (find a radius that yields .05 recurrence rate). If <code>plotMeasures = FALSE</code> this setting will be ignored.
title	A title for the plot
xlabel	An x-axis label
ylabel	An y-axis label
plotSurrogate	Should a 2-panel comparison plot based on surrogate time series be added? If RM has attributes <code>y1</code> and <code>y2</code> containing the time series data (i.e. it was created by a call to <code>rp()</code> ), the following options are available: "RS" (random shuffle), "RP" (randomised phases), "AAFT" (amplitude adjusted fourier transform). If no timeseries data is included, the columns will be shuffled. NOTE: This is not a surrogate test, just 1 surrogate is created from <code>y1</code> .
returnOnlyObject	Return the ggplot object only, do not draw the plot (default = TRUE)

**Value**

A nice plot of the recurrence matrix.

**See Also**

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2bi](#), [di2we](#), [dist\\_hamming](#), [rp\\_lineDist](#), [rp\\_nzdiags](#), [rp\\_size](#), [rp](#)

---

<i>rp_size</i>	<i>rp_size</i>
----------------	----------------

---

**Description**

*rp\_size*

**Usage**

`rp_size(mat, AUTO = NULL, theiler = NULL)`

**Arguments**

<code>mat</code>	A Matrix object
<code>AUTO</code>	Is the Matrix an Auto Recurrence Matrix? If so, the length of the diagonal will be subtracted from the matrix size, pass FALSE to prevent this behaviour. If NULL (default) AUTO will take on the value of <code>isSymmetric(mat)</code> .
<code>theiler</code>	Should a Theiler window be applied?

**Value**

Matrix size for computation of recurrence measures.

**See Also**

Other Distance matrix operations (recurrence plot): [bandReplace](#), [di2bi](#), [di2we](#), [dist\\_hamming](#), [rp\\_lineDist](#), [rp\\_nzdiags](#), [rp\\_plot](#), [rp](#)

**Examples**

```
# Create a 10 by 10 matrix
library(Matrix)
m <- Matrix(rnorm(10),10,10)

rp_size(m,TRUE,0) # Subtract diagonal
rp_size(m,FALSE,0) # Do not subtract diagonal
rp_size(m,NULL,0) # Matrix is symmetrical, AUTO is set to TRUE
rp_size(m,NULL,1) # Subtract a Theiler window of 1 around and including the diagonal
```

sa2fd\_dfa

*Informed Dimension estimate from DFA slope (H)***Description**

Conversion formula: Detrended Fluctuation Analysis (DFA) estimate of the Hurst exponent (a self-affinity parameter *sa*) to an informed estimate of the (fractal) dimension (FD).

**Usage**

```
sa2fd_dfa(sa, ...)
```

**Arguments**

<code>sa</code>	Self-Afinity parameter estimate based on DFA slope (e.g., <a href="#">fd_sda()</a> ).
<code>...</code>	Other arguments

**Details**

The DFA slope (*H*) will be converted to a dimension estimate using:

$$D_{DFA} \approx 2 - (\tanh(\log(3) * sa))$$



*sa2fd\_psd*

73

## Value

An informed estimate of the Fractal Dimension, see Hasselmann(2013) for details.

## Author(s)

Fred Hasselmann

## References

Hasselmann, F. (2013). When the blind curve is finite: dimension estimation and model inference based on empirical waveforms. *Frontiers in Physiology*, 4, 75. <http://doi.org/10.3389/fphys.2013.00075>

---

<i>sa2fd_psd</i>	<i>Informed Dimension estimate from Spectral Slope (alpha)</i>
------------------	--

---

## Description

Conversion formula: From periodogram based self-affinity parameter estimate (sa) to an informed estimate of the (fractal) dimension (FD).

## Usage

`sa2fd_psd(sa, ...)`

## Arguments

<code>sa</code>	Self-Affinity parameter estimate based on PSD slope (e.g., <code>fd_psd()</code> )
<code>...</code>	Other arguments

## Details

The spectral slope will be converted to a dimension estimate using:

$$D_{PSD} \approx \frac{3}{2} + \frac{14}{33} * \tanh \left( Slope * \ln(1 + \sqrt{2}) \right)$$

## Value

An informed estimate of the Fractal Dimension, see Hasselmann(2013) for details.

## Author(s)

Fred Hasselmann

## References

Hasselmann, F. (2013). When the blind curve is finite: dimension estimation and model inference based on empirical waveforms. *Frontiers in Physiology*, 4, 75. <http://doi.org/10.3389/fphys.2013.00075>

---

sa2fd\_sda

*Informed Dimension estimate from SDA slope.*


---

### Description

Conversion formula: Standardised Dispersion Analysis (SDA) estimate of self-affinity parameter (SA) to an informed estimate of the fractal dimension (FD).

### Usage

```
sa2fd_sda(sa, ...)
```

### Arguments

sa	Self-affinity parameter estimate based on SDA slope (e.g., <code>fd_sda()</code> ).
...	Other arguments

### Details

Note that for some signals different PSD slope values project to a single SDA slope. That is, SDA cannot distinguish dplyr::between all varieties of power-law scaling in the frequency domain.

### Value

An informed estimate of the Fractal Dimension, see Hasselman(2013) for details.

### Author(s)

Fred Hasselman

### References

Hasselman, F. (2013). When the blind curve is finite: dimension estimation and model inference based on empirical waveforms. *Frontiers in Physiology*, 4, 75. <http://doi.org/10.3389/fphys.2013.00075>

---

set\_command\_line\_rp

*Set command line RQA executable*


---

### Description

Set command line RQA executable

### Usage

```
set_command_line_rp()
```

### Value

Message informing whether the procedure was succesful.

---

ssg_gwf2long	<i>Import GridWare files</i>
--------------	------------------------------

---

**Description**

Import GridWare files

**Usage**

```
ssg_gwf2long(gwf_name, delta_t = 0.01, returnOnlyData = TRUE,
             saveLongFormat = FALSE)
```

**Arguments**

gwf_name	Name of the GridWare project file. A directory named <code>../gwf_name_trjs</code> must be present at the location of the project file.
delta_t	Time between two samples or sampling frequency
returnOnlyData	Just return the data, do not return a list object with data, variable info and preferences.
saveLongFormat	Save the long format trajectory data as a <code>.csv</code> file in the same location as <code>gwf_name</code>

**Value**

A data frame containing State Space Grid trajectories, or a list object with additional info.

**See Also**

Other State Space Grid functions: [factor\\_obs\\_exp](#), [ssg\\_winning](#)

---

ssg_winning	<i>Winning procedure for SSG</i>
-------------	----------------------------------

---

**Description**

Find attractor states in a State Space Grid using a winning procedure.

**Usage**

```
ssg_winning(durations, screeCut)
```

**Arguments**

durations	A data frame obtained by function <a href="#">ts_duration()</a>
screeCut	Cutoff based on a scree plot.

**Value**

Attractor frame

See Also

Other State Space Grid functions: [factor\\_obs\\_exp](#), [ssg\\_gwf2long](#)

---

SWtestE	<i>Small World test</i>
---------	-------------------------

---

Description

Small World test

Usage

SWtestE(g, p = 1, N = 20)

Arguments

g	An igraph object
p	p
N	N

---

ts_center	<i>Center a vector</i>
-----------	------------------------

---

Description

Center a vector

Usage

ts\_center(numvec, na.rm = TRUE, type = c("mean", "median")[1])

Arguments

numvec	A numeric vector
na.rm	Set the na.rm field
type	Center on the "mean" (default) or the "median" of the vector.

Value

A mean or median centered vector

Author(s)

Fred Hasselman

See Also

Other Time series operations: [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrnd](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

ts_changeindex	<i>Find change indices</i>
----------------	----------------------------

---

**Description**

Find change indices

**Usage**

```
ts_changeindex(y, returnRectdata = TRUE, groupVar = NULL,
  labelVar = NULL, discretize = FALSE, nbins = 5)
```

**Arguments**

y	An indicator variable representing different levels of a variable or factor
returnRectdata	Return a dataframe suitable for shading a ggplot2 graph with <code>ggplot2::geom_rect()</code>
groupVar	Pass a value (length 1) or variable (length of y) that can be used as a variable to join the indices by if returnRectdata = TRUE
labelVar	If y is not a character vector, provide a vector of labels equal to length(y)
discretize	If y is a continuous variable, setting discretize = TRUE will partition the values of y into nbins number of bins, each value of y will be replaced by its bin number.
nbins	Number of bins to use to change a continuous y (if discretize = TRUE) into a variable with nbins levels

**Value**

Either a vector with the indices of change in y, or, a data frame with variables xmin, xmax, ymin, ymax, label

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

ts_checkfix	<i>Check and/or Fix a vector</i>
-------------	----------------------------------

---

**Description**

Check and/or Fix a vector

**Usage**

```
ts_checkfix(y, checkNumericVector = TRUE, checkWholeNumbers = FALSE,
  checkTimeVector = FALSE, checkPow2 = FALSE, checkScale = FALSE,
  checkSummationOrder = FALSE, checkNonStationarity = FALSE,
  checkNonHomogeneity = FALSE, fixNumericVector = FALSE,
  fixWholeNumbers = FALSE, fixTimeVector = FALSE, fixPow2 = FALSE,
  fixNA = TRUE, fixScale = FALSE, fixSummationOrder = FALSE,
  fixNonStationarity = FALSE, fixNonHomogeneity = FALSE)
```

**Arguments**

y	A time series object or numeric vector
checkNumericVector	is 1D numeric vector?
checkWholeNumbers	contains only wholenumbers?
checkTimeVector	has time vector?
checkPow2	length is power of 2?
checkScale	checkScale
checkSummationOrder	checkSummationOrder
checkNonStationarity	checkNonStationarity
checkNonHomogeneity	checkNonHomogeneity
fixNumericVector	return a 1D numeric vector (WARNING: Data frames and Matrices with NCOL > 1 wil be converted to long form)
fixWholeNumbers	fixWholeNumber
fixTimeVector	fixTimeVector
fixPow2	foxPow2
fixNA	fixNA
fixScale	fixScale
fixSummationOrder	fixSummationOrder
fixNonStationarity	fixNonStationarity
fixNonHomogeneity	fixNonHomogeneity

**Value**

A 'check' report and/or a 'fixed' vector y.

*ts\_dettrend*

79

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_dettrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

*ts\_dettrend**Detrend a time series***Description**

Detrend a time series

**Usage**

```
ts_dettrend(y, polyOrder = 1)
```

**Arguments**

<i>y</i>	A time series of numeric vector
<i>polyOrder</i>	order Order of polynomial trend to remove

**Value**

Residuals after detrending polynomial of order *order*

**Author(s)**

Fred Hasselman

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

*ts\_diff**Derivative of time series***Description**

Iteratively differenced series up to order. The same length as the original series is recovered by calculating the mean of two vectors for each iteration: One with a duplicated first value and one with a duplicated last value.

**Usage**

```
ts_diff(y, order = 1, addColumns = TRUE, keepDerivatives = FALSE,
        maskEdges = NULL, silent = TRUE)
```

**Arguments**

y	A timeseries object or numeric vector or a matrix in which columns are variables and rows are numeric values observed over time.
order	How many times should the difference iteration be applied? (default = 1)
addColumnns	Should the derivative(s) be added to the input vector/matrix as columns? (default = TRUE)
keepDerivatives	If TRUE and order > 1, all derivatives from 1:order will be returned as a matrix (default = FALSE)
maskEdges	Mask the values at the edges of the derivatives by any numeric type that is not NULL (default = NULL)
silent	Silent-ish mode

**Value**

Depending on the setting of addColumns and the object type passed as y, a vector of equal length as y iteratively differenced by order times; a matrix with derivatives, or a matrix with original(s) and derivative(s).

**Note**

The values at the edges of the derivatives represent endpoint averages and should be excluded from any subsequent analyses. Set argument maskEdges to a value of your choice.

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

**Examples**

```
# Get an interesting numeric vector from package DescTools
y <- DescTools::Fibonacci(1:26)

# Return the first order derivative as a vector
ts_diff(y=y,addColumnns=FALSE)

# Return original and derivative as a matrix
plot(stats::ts(ts_diff(y=y, addColumns=TRUE)))

# Works on multivariate data objects with mixed variable types
df <- data.frame(x=letters, y=1:26, z=sin(y))

# Returns only derivatives of the numeric columnns
ts_diff(y=df,addColumnns=FALSE)

# Returns original data with derivatives of the numeric columns
ts_diff(y=df, order=4, addColumns=TRUE)

# Plot logistic S-curve and derivatives 1 to 3
```



```

S <- stats::plogis(seq(-5,5,.1))
plot(stats::ts(ts_diff(S, order=3, keepDerivatives = TRUE)))
abline(v=which(seq(-5,5,.1)==0), col= "red3", lwd=3)

# Plot again, but with masked edges
(maskEdge <- ts_diff(S, order=3, keepDerivatives = TRUE, maskEdges = NA))
plot(stats::ts(maskEdge))
abline(v=which(seq(-5,5,.1)==0), col= "red3", lwd=3)

```

---

ts_discrete	<i>Discrete representation</i>
-------------	--------------------------------

---

## Description

Return a discrete representation of  $y$  by binning the observed values and returning the transfer probabilities.

## Usage

```
ts_discrete(y, nbins = ceiling(2 * NROW(y)^(1/3)), keepNA = TRUE)
```

## Arguments

<code>y</code>	Numeric vector or time series to be discretised.
<code>nbins</code>	Number of bins to use for calculating transfer probabilities (default = $\text{ceiling}(2 \times \text{length}(y)^{(1/3)})$ )
<code>keepNA</code>	If TRUE, any NA values will first be removed and later re-inserted into the discretised time series.

## Value

A discretised version of  $y$

## See Also

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

ts_duration	<i>Time series to Duration series</i>
-------------	---------------------------------------

---

**Description**

Time series to Duration series

**Usage**

```
ts_duration(y, timeVec = stats::time(y), fs = stats::frequency(y),
  tolerance = 0)
```

**Arguments**

y	A time series, numeric vector, or categorical variable.
timeVec	A vector, same length as y containing timestamps, or, sample indices.
fs	Optional sampling frequency if timeVec represents sample indices. An extra column duration.fs will be added which represents $1/fs * \text{duration}$ in samples
tolerance	A number tol indicating a range $[y-tol, y+tol]$ to consider the same value. Useful when y is continuous (default = 0)

**Value**

A data frame

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

**Examples**

```
library(invctr)
# Create data with events and their timecodes
coder <- data.frame(beh=c("stare","stare","coffee","type","type","stare"), t=c(0,5,10,15,20,25))

ts_duration(y = coder$beh, timeVec = coder$t)
```

---

ts_embed	<i>Delay embedding of a time series</i>
----------	---

---

**Description**

Create a state vector based on an embedding lag and a number of embedding dimanesions.

**Usage**

```
ts_embed(y, emDim, emLag, returnOnlyIndices = FALSE, silent = TRUE)
```

**Arguments**

y	Time series
emDim	Embedding dimension
emLag	Embedding lag
returnOnlyIndices	Return only the index of y for each surrogate dimension, not the values (default = FALSE)
silent	Silent-ish mode

**Value**

The lag embedded time series

**Author(s)**

Fred Hasselman

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

ts_integrate	Create a timeseries profile
--------------	-----------------------------

---

**Description**

Create a timeseries profile

**Usage**

```
ts_integrate(y)
```

**Arguments**

y	A 1D timeseries
---	-----------------

**Value**

The profile

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

**Examples**

```
y <- runif(1000,-3,3)
plot(ts(y))
y_i <- ts_integrate(y)
plot(ts(y_i))
```

---

ts_levels	Detect levels in time series
-----------	------------------------------

---

**Description**

Use recursive partitioning function ([rpart](#) to perform a 'classification' of relatively stable levels in a timeseries.

**Usage**

```
ts_levels(y, minDataSplit = 12,
  minLevelDuration = round(minDataSplit/3), changeSensitivity = 0.01,
  maxLevels = 30, method = c("anova", "poisson", "class", "exp")[1])
```

**Arguments**

y	A time series of numeric vector
minDataSplit	An integer indicating how many datapoints should be in a segment before it will be analysed for presence of a level change (default = 12)
minLevelDuration	Minimum duration (number of datapoint) of a level (default = round(minDataSplit/3))
changeSensitivity	A number indicating a criterion of change that must occur before declaring a new level. Higher numbers indicate higher levels of change must occur before a new level is considered. For example, if method = "anova", the overall $R^2$ after a level is introduced must increase by the value of changeSensitivity, see the cp parameter in <a href="#">rpart::rpart.control</a> . (default = 0.01)
maxLevels	Maximum number of levels in one series (default = 30)
method	The partitioning method to use, see the manual pages of <a href="#">rpart</a> for details.

**Value**

A list object with fields tree and pred. The latter is a data frame with columns x (time), y (the variable of interest) and p the predicted levels in y.

**Author(s)**

Fred Hasselman

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

**Examples**

```
# Levels in white noise?

set.seed(4321)
wn <- ts_levels(rnorm(100))
plot(wn$pred$x, wn$pred$y, type = "l")
lines(wn$pred$p, col = "red3", lwd = 2)

# This is due to the default changeSensitivity of 0.01

lines(ts_levels(rnorm(100), changeSensitivity = .1)$pred$p, col = "steelblue", lwd = 2)
```

---

ts_peaks	<i>Find Peaks or Wells</i>
----------	----------------------------

---

**Description**

Find Peaks or Wells

**Usage**

```
ts_peaks(y, window = 3, includeWells = FALSE,
         minPeakDist = round(window/2), minPeakHeight = 0.2 * diff(range(y,
         na.rm = TRUE)))
```

**Arguments**

y	A time series or numeric vector
window	Window in which to look for peaks or wells
includeWells	Find wells?
minPeakDist	Minimum distance between peaks or wells
minPeakHeight	Minimum height / depth for a peak / well

**Value**

Index with peak or well coordinates

**Author(s)**

Fred Hasselman

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

ts_permtest_block	<i>Permutation Test: Block Randomisation</i>
-------------------	--

---

**Description**

Use block randomisation to get a permutation test evaluation of the deviation of an observed value at each time point from a target value. To do block permutation without any tests, pass NULL for argument targetValue.

**Usage**

```
ts_permtest_block(y1, y2 = NULL, targetValue = 0, Nperms = 19,
                 sim = "geom", l = 3, alpha = 0.05, returnBootObject = FALSE)
```

**Arguments**

y1	Time series 1. The goal of the permutation test will be to decide whether the difference $y1 - \text{targetValue} \neq 0$ for each time point, given alpha.
y2	An optional second time series. If this timeseries is provided then the goal of the permutation test will be the to decide wether the difference $y2 - y1 \neq \text{targetValue}$ for each time point, given alpha.
targetValue	The target value for the permutation test. If NULL, the function will return a data frame with the block randomised surrogates columns (default = 0)
Nperms	Number of permutations (default = 19)
sim	Value passed to the sim argument of <code>boot::tsboot()</code> valid options are: "model", "fixed", "geom", (default = "geom")
l	Block sizes to use, see <code>boot::tsboot()</code> for details (default = 3)
alpha	Alpha level for deciding significance (default = 0.05)
returnBootObject	Return the boot object (default = FALSE)
...	Other arguments passed to function <code>boot::tsboot()</code>

**Value**

A data frame with the difference time series and variables indicating N and significance.

**See Also**

Other Time series operations: `ts_center`, `ts_changeindex`, `ts_checkfix`, `ts_detrend`, `ts_diff`, `ts_discrete`, `ts_duration`, `ts_embed`, `ts_integrate`, `ts_levels`, `ts_peaks`, `ts_permtest_transmat`, `ts_rasterize`, `ts_sd`, `ts_slice`, `ts_standardise`, `ts_sumorder`, `ts_symbolic`, `ts_trimfill`, `ts_windower`

**Examples**

```
set.seed(4321)
y1 <- rnorm(5000)
y2 <- y1 - (mean(y1) + rnorm(1))

ts_permtest_block(y1 = y1, y2 = y2)
```

---

ts\_permtest\_transmat    *Permutation Test: Transition Matrix*

---

**Description**

Monte Carlo resampling of a time series using a discretised version of y, a sequence of bin numbers with unique values equal to nbins:

1. The discrete version of y will be used to generate a transition matrix of size nbins X nbins.
2. This transition matrix will be used to resample values

**Usage**

```
ts_permtest_transmat(y1, y2 = NULL, targetValue = 0,
  nbins = ceiling(2 * length(y1)^(1/3)), Nperms = 19, alpha = 0.05,
  keepNA = TRUE)
```

**Arguments**

y1	Time series 1. The goal of the permutation test will be to decide whether the difference $y1 - \text{targetValue} \neq 0$ for each time point, given alpha.
y2	An optional second time series. If this timeseries is provided then the goal of the permutation test will be the to decide wether the difference $y2 - y1 \neq \text{targetValue}$ for each time point, given alpha.
targetValue	The target value for the permutation test. If NULL, the function will return a data frame with the block randomised surrogates columns (default = 0)
nbins	Number of bins to use (default = $\text{ceiling}(2 * \text{length}(y1)^{(1/3)})$ )
Nperms	Number of permutations (default = 19)
alpha	Alpha level for deciding significance (default = 0.05)
keepNA	keepNA

**Value**

Resampled series

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

**Examples**

```
set.seed(4321)
y <- rnorm(5000)
ts_permtest_transmat(y)
```

---

ts_rasterize	<i>Turn a 1D time series vector into a 2D curve</i>
--------------	---

---

**Description**

Turn a 1D time series vector into a 2D curve

**Usage**

```
ts_rasterize(y, unitSquare = FALSE, toSparse = TRUE, resolution = 2)
```



ts\_sd

89

Arguments

y	A 1D time series object or numeric vector.
unitSquare	Convert the series to a unit square? (default = FALSE)
toSparse	Convert to sparse Matrix (default = FALSE)
resolution	Factor by which dimensions will be multiplied (default = 2)

Value

A (sparse) matrix representing the time series as a curve in 2D space

See Also

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

Examples

```
y <- rnorm(100)
plot(ts(y))

y_img <- ts_rasterize(y)
image(y_img,col=c("white","black"))
```

---

ts_sd	<i>Standard Deviation estimates</i>
-------	-------------------------------------

---

Description

Calculates the population estimate of the standard deviation, or the unadjusted standard deviation.

Usage

```
ts_sd(y, na.rm = TRUE, type = c("Bessel", "unadjusted")[1],
      silent = TRUE)
```

Arguments

y	Time series or numeric vector
na.rm	Remove missing values before calculations
type	Apply Bessel's correction (divide by N-1) or return unadjusted SD (divide by N)
silent	Silent-ish mode (default = TRUE)

Value

Standard deviation of y

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

<code>ts_slice</code>	<i>Slice columns of a matrix in epochs</i>
-----------------------	--

---

**Description**

Slice columns of a matrix in epochs

**Usage**

```
ts_slice(y, epochSz = 4)
```

**Arguments**

<code>y</code>	A matrix with timeseries as columns
<code>epochSz</code>	Epoch size

**Value**

A list with epochs

**Author(s)**

Fred Hasselman

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

<code>ts_standardise</code>	<i>Standardise a vector</i>
-----------------------------	-----------------------------

---

**Description**

Standardise a vector

**Usage**

```
ts_standardise(y, na.rm = TRUE, keepNAvalues = TRUE,  
  type = c("mean.sd", "median.mad")[1], adjustN = TRUE)
```

*ts\_sumorder*

91

**Arguments**

<i>y</i>	A time series or numeric vector
<i>na.rm</i>	Set the <i>na.rm</i> field
<i>keepNAvalues</i>	If <i>na.rm</i> = TRUE and <i>keepNAvalues</i> = TRUE, any NA values in <i>y</i> will be re-inserted after transformation.
<i>type</i>	Center on the "mean" and divide by <i>sd</i> (default), or center on "median" and divide by <i>mad</i>
<i>adjustN</i>	If TRUE, apply Bessel's correction (divide by <i>N</i> -1) or return the unadjusted SD (divide by <i>N</i> ) (default = TRUE)

**Value**

A standardised vector

**Author(s)**

Fred Hasselman

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

*ts\_sumorder**Adjust time series by summation order***Description**

Many fluctuation analyses assume a time series' Hurst exponent is within the range of 0.2 - 1.2. If this is not the case it is sensible to make adjustments to the time series, as well as the resulting Hurst exponent.

**Usage**

```
ts_sumorder(y, scaleS = NULL, polyOrder = 1, minData = 4)
```

**Arguments**

<i>y</i>	A time series of numeric vector
<i>scaleS</i>	The scales to consider for DFA1
<i>polyOrder</i>	Order of polynomial for detrending in DFA (default = 1)
<i>minData</i>	Minimum number of data points in a bin needed to calculate detrended fluctuation

Details

Following recommendations by [https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image\\_m/fphys-03-00141-t001.jpg](https://www.frontiersin.org/files/Articles/23948/fphys-03-00141-r2/image_m/fphys-03-00141-t001.jpg)Ihlen (2012), a global Hurst exponent is estimated using DFA and y is adjusted accordingly:

- $1.2 < H < 1.8$  first derivative of y, attribute Hadj = 1
- $H > 1.8$  second derivative of y, attribute Hadj = 2
- $H < 0.2$  y is centered and integrated, attribute Hadj = -1
- $0.2 \leq H \leq 1.2$  y is unaltered, attribute Hadj = 0

Value

The input vector, possibly adjusted based on H with an attribute "Hadj" containing an integer by which a Hurst exponent calculated from the series should be adjusted.

References

Ihlen, E. A. F. E. (2012). Introduction to multifractal detrended fluctuation analysis in Matlab. Frontiers in physiology, 3, 141.

See Also

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_symbolic](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

ts_symbolic	<i>Symbolic representation</i>
-------------	--------------------------------

---

Description

Return a discrete representation of y by binning the observed values and returning the transfer probabilities.

Usage

```
ts_symbolic(y, keepNA = TRUE, usePlateaus = FALSE, doPlot = FALSE)
```

Arguments

y	Numeric vector or time series to be discretised.
keepNA	If TRUE, any NA values will first be removed and later re-inserted into the discretised time series.
usePlateaus	Treat consecutive "same" values after "peak" or "trough" as a "peak"/"trough".
doPlot	Create a plot of the symbolized series.

Value

A discretised version of y

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_trimfill](#), [ts\\_windower](#)

---

ts_transmat	<i>Transition matrix</i>
-------------	--------------------------

---

**Description**

Create a transition matrix from a discrete time series, e.g. to generate Monte Carlo simulations.

**Usage**

```
ts_transmat(yd, nbins = unique(yd))
```

**Arguments**

yd	A discrete numeric vector or time series, e.g. transformed using <a href="#">ts_discrete()</a> , or, <a href="#">ts_symbolic()</a> .
nbins	The number of bins used to transform a continuous time series, or, the number of expected (given nbins, or, theoretically possible) values for a discrete series (default = unique(yd))

**Value**

A transition probability matrix

**Examples**

```
set.seed(4321)

# Random uniform numbers
y <- runif(10,0,20)

# Discrete version
yd <- ts_discrete(y, nbins = 10)

# Transition probabilities
ts_transmat(yd = yd, nbins = 10)

# Note: The number of 'observed' bins differs from 'expected' bins
table(yd)

# Not specifying the expected bins will generate a different matrix!
ts_transmat(yd = yd, nbins = length(unique(yd)))
```

ts\_trimfill

*Trim or Fill Vectors***Description**

Trim the largest vector by cutting it, or filling it with NA. Fill the shortest vector with padding.

**Usage**

```
ts_trimfill(x, y, action = c("fill", "trim.cut", "trim.NA")[1],
  type = c("end", "center", "front")[1], padding = 0, silent = TRUE)
```

**Arguments**

x	A numeric vector
y	A numeric vector
action	Use "fill" to fill the shortest vector with padding (default); "trim.cut" to trim the longest vector to the length of the shortest; "trim.NA" to fill the longest vector with NA. This is a shortcut for running action = "trim.cut" with padding=NA, which can be useful if one wants to match the shortest series, but preserve the original length of largest vector.
type	Should trimming or filling take place at the "end" (default), or "front" of the vector? The option "center" will try to distribute trimming by NA or filling by padding evenly across the front and end of the vector.
padding	A value to use for padding (default = 0)
silent	Run silent-ish

**Value**

A list with two vectors of equal length.

**Author(s)**

Fred Hasselman

**See Also**

il\_mi

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_windower](#)

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_windower](#)

---

ts_windower	<i>Get sliding window indices</i>
-------------	-----------------------------------

---

**Description**

Get sliding window indices

**Usage**

```
ts_windower(y, win = length(y), step = round(win/2), overlap = NA,
  adjustY = NA)
```

**Arguments**

y	A time series or numeric vector
win	Size of the window to slide across y
step	Size of steps between windows. Can be larger than win, but is ignored if overlap is not NA.
overlap	A value between [0 .. 1]. If overlap is not NA (default), the value of step is ignored and set to floor(overlap*win). This produces indices in which the size of step is always smaller than win, e.g. for fluctuation analyses that use binning procedures to represent time scales.
adjustY	If not NA, or, FALSE a list object with fields that match one or more arguments of <a href="#">ts_trimfill</a> (except for x, y), e.g. list(action="trim.NA", type="end", padding=NA, silent=TRUE). See Return value below for details.

**Value**

If adjustY = FALSE, or, a list object with fields that represent arguments of [ts\\_trimfill](#), then the (adjusted) vector y is returned with an attribute "windower". This is a list object with fields that contain the indices for each window that fits on y, given win, step or overlap and the settings of adjustY. If adjustY = NA, only the list object is returned.

**See Also**

Other Time series operations: [ts\\_center](#), [ts\\_changeindex](#), [ts\\_checkfix](#), [ts\\_detrrend](#), [ts\\_diff](#), [ts\\_discrete](#), [ts\\_duration](#), [ts\\_embed](#), [ts\\_integrate](#), [ts\\_levels](#), [ts\\_peaks](#), [ts\\_permtest\\_block](#), [ts\\_permtest\\_transmat](#), [ts\\_rasterize](#), [ts\\_sd](#), [ts\\_slice](#), [ts\\_standardise](#), [ts\\_sumorder](#), [ts\\_symbolic](#), [ts\\_trimfill](#)

# Index

## \*Topic **datasets**

- RNG, 63
- add\_alpha, 5
- as.numeric\_character, 5, 6
- as.numeric\_discrete, 6
- as.numeric\_factor, 6, 6
- bandReplace, 7, 21–23, 67, 69–72
- base::matrix(), 21, 22
- boot::tsboot(), 87
- casnet::est\_emDim(), 13
- casnet::est\_emLag(), 13
- casnet::ts\_discrete(), 47, 59
- casnet::ts\_standardise(), 28, 29
- crqa\_cl, 8, 13, 14, 16
- crqa\_diagProfile, 11
- crqa\_parameters, 11, 12, 14, 16
- crqa\_radius, 11, 13, 13, 16
- crqa\_rp, 11, 13, 14, 15
- crqa\_rp\_measures, 11, 13, 14, 16
- dc\_ccp, 16, 18, 19, 21, 51–53
- dc\_d, 17, 17, 19, 21, 51–53
- dc\_f, 17, 18, 18, 21, 51–53
- dc\_f2, 19
- dc\_win, 17–19, 19, 51–53
- dc\_win(), 16, 18, 19, 52, 53
- di2bi, 7, 21, 22, 23, 63, 65–67, 69–72
- di2we, 7, 21, 22, 23, 63, 65–67, 69–72
- dist\_hamming, 7, 21, 22, 22, 67, 69–72
- elascer, 23
- est\_emDim, 24
- est\_emLag, 25
- factor\_obs\_exp, 26, 75, 76
- fd\_allan, 26, 30–35
- fd\_boxcount2D, 27
- fd\_dfa, 27, 29, 31–35
- fd\_mfdfa, 27, 30, 30, 32–35
- fd\_psd, 27, 30, 31, 31, 33–35
- fd\_psd(), 73
- fd\_RR, 27, 30–32, 32, 34, 35
- fd\_sda, 27, 30–33, 33, 35
- fd\_sda(), 72, 74
- fd\_sev, 27, 30–34, 34
- flight\_Cauchy, 35
- flight\_LevyPareto, 36
- flight\_Rayleigh, 37
- fractal::DFA(), 29
- fractal::FNN(), 13
- fractal::HDEst(), 31
- geom\_curve(), 43–45
- get\_os, 38
- getColours, 38
- gg\_plotHolder, 39
- gg\_theme, 40
- ggplot, 61
- ggplot2, 43, 45
- ggplot2::geom\_rect(), 77
- ggplot2::ggplot(), 16–18, 20, 51–53, 61
- ggplot2::scale\_size(), 44, 46
- growth\_ac, 40, 41
- growth\_ac\_cond, 41, 41
- igraph, 43, 45
- igraph::degree(), 44, 45
- igraph::hubscore(), 44, 45
- igraph::igraph(), 62
- igraph::sample\_pa(), 54
- igraph::sample\_smallworld(), 57, 58
- infotheo::discretize(), 47, 59
- infotheo::infotheo(), 46
- layout\_as\_spiral, 42
- layout\_as\_spiral(g), 44, 46
- lubridate::stamp(), 16, 18–20, 51–53
- make\_spiral\_focus, 43
- make\_spiral\_graph, 45
- Matrix::Matrix(), 21, 22
- mi\_mat, 47, 48, 48
- mif, 46, 48
- mif(), 59
- mif\_interlayer, 47, 47, 48
- noise\_fBm, 49



noise\_fGn, 49  
 noise\_powerlaw, 50  
 nonlinearTseries::estimateEmbeddingDim(), 25  
 nonlinearTseries::timeLag(), 13  
 normalizePath(), 10  
  
 parallel::detectCores(), 9, 15  
 plotDC\_ccp, 17–19, 21, 50, 52, 53  
 plotDC\_lvl, 17–19, 21, 51, 51, 53  
 plotDC\_res, 17–19, 21, 51, 52, 52  
 plotFD\_loglog, 53  
 plotNET\_BA, 54, 55–58  
 plotNET\_groupColour, 54, 54, 56–58  
 plotNET\_groupWeight, 54, 55, 55, 57, 58  
 plotNET\_prep, 54–56, 56, 58  
 plotNET\_SW, 54–57, 57  
 plotRED\_acf, 58, 59  
 plotRED\_mif, 58, 59  
 plotRED\_mif(), 47, 59  
 plotSUR\_hist, 59  
 plotTS\_multi, 60  
 proxy::dist(), 62, 67  
 purrr::map2(), 9, 15  
  
 repmat, 61  
 rn, 21, 22, 62, 65, 66  
 rn\_plot, 21, 22, 63, 64, 65, 66  
 rn\_plot(), 62  
 rn\_recSpec, 21, 22, 63, 65, 65, 66  
 rn\_scaleoGram, 21, 22, 63, 65, 66  
 RNG, 63  
 rp, 7, 21–23, 66, 69–72  
 rp(), 64, 71  
 rp\_copy\_attributes, 68  
 rp\_lineDist, 7, 21–23, 67, 68, 70–72  
 rp\_nzdiags, 7, 21–23, 67, 69, 69, 71, 72  
 rp\_plot, 7, 21–23, 67, 69, 70, 70, 72  
 rp\_plot(), 9, 10, 62, 67  
 rp\_size, 7, 21–23, 67, 69–71, 71  
 rpart, 84, 85  
 rpart::rpart.control, 85  
  
 sa2fd\_dfa, 72  
 sa2fd\_psd, 73  
 sa2fd\_sda, 74  
 sapa::SDF(), 31  
 set\_command\_line\_rp, 74  
 ssg\_gwf2long, 26, 75, 76  
 ssg\_winnowing, 26, 75, 75  
 stabledist::rstable(), 35–37  
 SWtestE, 76  
 system2(), 10  
  
 taper, 31  
 ts\_center, 76, 77, 79–95  
 ts\_changeindex, 76, 77, 79–95  
 ts\_checkfix, 76, 77, 77, 79–95  
 ts\_detrend, 28, 76, 77, 79, 79, 80–95  
 ts\_diff, 76, 77, 79, 79, 81–95  
 ts\_discrete, 76, 77, 79, 80, 81, 82–95  
 ts\_discrete(), 93  
 ts\_duration, 76, 77, 79–81, 82, 83–95  
 ts\_duration(), 75  
 ts\_embed, 76, 77, 79–82, 83, 84–95  
 ts\_integrate, 76, 77, 79–83, 84, 85–95  
 ts\_levels, 76, 77, 79–84, 84, 86–95  
 ts\_peaks, 76, 77, 79–85, 86, 87–95  
 ts\_permtest\_block, 76, 77, 79–86, 86, 88–95  
 ts\_permtest\_transmat, 76, 77, 79–87, 87, 89–95  
 ts\_rasterize, 76, 77, 79–88, 88, 90–95  
 ts\_sd, 76, 77, 79–89, 89, 90–95  
 ts\_slice, 76, 77, 79–90, 90, 91–95  
 ts\_standardise, 76, 77, 79–90, 90, 92–95  
 ts\_sumorder, 76, 77, 79–91, 91, 93–95  
 ts\_symbolic, 76, 77, 79–92, 92, 94, 95  
 ts\_symbolic(), 93  
 ts\_transmat, 93  
 ts\_trimfill, 76, 77, 79–93, 94, 95  
 ts\_windower, 76, 77, 79–94, 95