Package: kernelPhil (via r-universe)

May 11, 2026

Title Kernel Smoothing Tools for Philology and Historical Dialectology

Version 0.2

URL http://www.icge.co.uk/

Description Contains kernel smoothing tools designed for use by historical dialectologists and philologists for exploring spatial and temporal patterns in noisy historical language data, such as that obtained from historical texts. The main way in which these might differ from other implementations of kernel smoothing is that they assume that the function (linguistic variable) being explored has the form of the relative frequency of a series of discrete possibilities (linguistic variants). This package also offers a way of exploring distributions in 2-dimensional space and in time with separate kernels, and tools for identifying appropriate bandwidths for these.

License GPL (>= 3)

Encoding UTF-8

Imports benchmarkme, directlabels, dplyr, ggplot2, grDevices, gridExtra, Hmisc, pbapply, reshape2, rlang, stats, terra, utils, wordspace

RoxygenNote 7.3.3

NeedsCompilation no

Author Tamsin Blaxter [aut, cre] (ORCID: <https://orcid.org/0000-0002-1466-8306>)

Maintainer Tamsin Blaxter <tamsinblaxter@gmail.com>

Repository https://cran.r-universe.dev

Date/Publication 2025-11-12 21:10:02 UTC

RemoteUrl https://github.com/cran/kernelPhil

RemoteRef HEAD

RemoteSha 22b834d0de4fa382833dd3df655ec7f1b7595f97

Contents

`calculate.bandwidths.by.resolution`
Calculate temporal bandwidths by spatial resolution

Description

This function calculates relationships between temporal bandwidth and possible spatial resolution for a given power and suggests minimum possible temporal bandwidth for a given resolution

Usage

```
calculate.bandwidths.by.resolution(
  dataset,
  dependent.variable = "dependent.variable",
  time = "year",
  weight = "weight",
  alpha = 0.05,
  margin = 0.1,
  measure.times,
  temporal.bandwidth.limits,
  temporal.bandwidth.n.levels = 200,
  minimum.spatial.resolution = 5,
  summary.plots = FALSE,
  kernel.function = gaussian.kernel
)
```

Arguments

`dataset`	The dataset to be smoothed as a data.frame.
`dependent.variable`	String name of the column in dataset with the dependent variable (defaults to "dependent.variable"); this column should be numeric or factor.
`time`	String name of the column in dataset with the time variable (defaults to "year").

weight	String name of column in the dataset with numeric weights (defaults to "weight").
alpha	Numeric alpha for calculating error margins (defaults to 0.05).
margin	Numeric desired error margin for calculating spatial bandwidths (defaults to 0.1).
measure.times	A numeric vector of specific times at which to make estimates; if not given, will default to seq(from=min(time),to=max(time),length.out=5).
temporal.bandwidth.limits	Numeric vector of length 2 specifying minimum and maximum temporal bandwidth to be tested (defaults to the range of time*0.01 to the range of time*2).
temporal.bandwidth.n.levels	Number of distinct levels of temporal bandwidth to be tested (defaults to 200).
minimum.spatial.resolution	Numeric minimum spatial resolution.
summary.plots	If TRUE, plots of smoothed sample density, the dependent variable, and variance are returned along with the plot of resolution by bandwidth.
kernel.function	The kernel function, one of gaussian.kernel, gaussian.square.kernel, triangular.kernel, square.kernel, or a custom function (defaults to gaussian.kernel).

Value

A list containing a plot of spatial resolution by temporal bandwidth, along with other summary plots of the data if summary.plots==TRUE, and the calculated minimum temporal bandwidth.

`calculate.bandwidths.by.separated.points`
Calculate temporal bandwidths by separated points

Description

This function calculates relationships between temporal bandwidth and spatial bandwidths at a series of specified points for a given power and suggests minimum possible temporal bandwidth such that bandwidths at those points are never greater than 2.2365*the distance to the nearest point (for gaussian kernels) or 2*that distance (for other kernels)

Usage

```
calculate.bandwidths.by.separated.points(
  dataset,
  dependent.variable = "dependent.variable",
  x = "x",
  y = "y",
  time = "year",
  weight = "weight",
  alpha = 0.05,
```

```
  margin = 0.1,
  separated.points,
  measure.times,
  temporal.bandwidth.limits,
  temporal.bandwidth.n.levels = 200,
  kernel.function = gaussian.kernel,
  projection = NA,
  include.visualisation = FALSE,
  separated.points.labels,
  round.up.low.variance = FALSE
)
```

Arguments

dataset	The dataset to be smoothed as a data.frame.
dependent.variable	String name of the column in dataset with the dependent variable (defaults to "dependent.variable"); this column should be numeric or factor.
x	String name of column containing numeric x co-ordinate (defaults to "x").
y	String name of column containing numeric y co-ordinate (defaults to "y").
time	String name of the column in dataset with the time variable (defaults to "year").
weight	String name of column in the dataset with numeric weights (defaults to "weight").
alpha	Numeric alpha for calculating error margins (defaults to 0.05).
margin	Numeric desired error margin for calculating spatial bandwidths (defaults to 0.1).
separated.points	Data.frame containing two columns same names as x,y in dataset with x and y coordinates of points to be kept separate.
measure.times	A numeric vector of specific times at which to make estimates; if not given, will default to seq(from=min(time),to=max(time),length.out=5).
temporal.bandwidth.limits	A numeric vector of length 2 specifying minimum and maximum temporal bandwidth to be tested (defaults to the range of time*0.01 to the range of time*2).
temporal.bandwidth.n.levels	Number of distinct levels of temporal bandwidth to be tested (defaults to 200).
kernel.function	The kernel function, one of gaussian.kernel, gaussian.square.kernel, triangular.kernel, square.kernel, or a custom function (defaults to gaussian.kernel).
projection	A spatial projection as a proj4 string - if given, data will be projected before smoothing and results will be deprojected before returning.
include.visualisation	If TRUE, will return a ggplot visualisation.
separated.points.labels	String vector of the names of the separated points (used in the visualisation).
round.up.low.variance	Set to TRUE if there are periods of time with extremely low variance.

Value

A list with suggested bandwidth and the choke point and time, plus a visualisation of bandwidths and resolutions if include.visualisation==TRUE.

`kernel.smooth.in.space`

Kernel smooth data in space alone

Description

This function performs kernel smoothing on a dataset in space alone.

Usage

```
kernel.smooth.in.space(
  dataset,
  dependent.variable = "dependent.variable",
  x = "x",
  y = "y",
  weight = "weight",
  normalise.by,
  data.type = "factor",
  alpha = 0.05,
  margin = 0.1,
  kernel.function = gaussian.kernel,
  adaptive.spatial.bw = TRUE,
  measure.points,
  projection = NA,
  round.up.low.variance = TRUE,
  explicit = TRUE
)
```

Arguments

`dataset`	Dataset to be smoothed as a data.frame.
`dependent.variable`	String name of the single column in dataset with the factor dependent variable (if data.type=="factor") or a vector of column names with numeric counts (if data.type=="count") (defaults to "dependent.variable").
`x`	String name of column containing numeric x co-ordinate (defaults to "x").
`y`	String name of column containing numeric y co-ordinate (defaults to "y").
`weight`	String name of column in the dataset with numeric weights (defaults to "weight").
`normalise.by`	String name of column by which data should be normalised (typically factor with document, speaker or writer ids).

`data.type` The type of the dependent variable: either "factor", if each row is a token, or "count", if each row is a document, speaker or writer with token counts in separate columns (defaults to "factor").

`alpha` Numeric alpha for calculating error margins (defaults to 0.05).

`margin` Numeric desired error margin for calculating spatial bandwidths (defaults to 0.1).

`kernel.function`
The kernel function, one of gaussian.kernel, gaussian.square.kernel, triangular.kernel, square.kernel, or a custom function (defaults to gaussian.kernel).

`adaptive.spatial.bw`
A boolean indicating whether the spatial bandwidth is adaptive (set to achieve margin at every point) or static (set to the average of bandwidths needed to achieve margin at every point).

`measure.points` A data.frame of spatial points at which estimates are to be made, with two columns with the same names as x,y in dataset; if not supplied, estimates are at the same locations as dataset.

`projection` The spatial projection as a proj4 string - if given, data will be projected before smoothing and results will be deprojected before returning.

`round.up.low.variance`
Set to TRUE if there are periods of time with extremely low variance (defaults to TRUE).

`explicit` If TRUE, progress will be reported with a progress bar (defaults to TRUE).

Value

A data.frame with the smoothed estimates.

Examples

```
n=400;
synthesised.data<-data.frame(x=stats::runif(n),y=stats::runif(n),
    year=stats::runif(n,0,sqrt(2)));
synthesised.data$dependent.variable<-unlist(lapply(1:nrow(synthesised.data),
    function(X){
    stats::dist(as.matrix(synthesised.data[c(1,X),1:2]),method =
        "euclidean")<synthesised.data$year[X];
}))
result<-kernelPhil::kernel.smooth.in.space(dataset = synthesised.data);
ggplot2::ggplot(result,ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
    ggplot2::geom_point();
```

`kernel.smooth.in.space.and.time` *Kernel smooth data in space and time*

Description

This function performs kernel smoothing on a dataset in time and space. A static temporal kernel is applied first, and then an (optionally) adaptive spatial kernel on this weighted data.

Usage

```
kernel.smooth.in.space.and.time(
  dataset,
  dependent.variable = "dependent.variable",
  x = "x",
  y = "y",
  time = "year",
  weight = "weight",
  normalise.by,
  data.type = "factor",
  alpha = 0.05,
  margin = 0.1,
  kernel.function = gaussian.kernel,
  adaptive.spatial.bw = TRUE,
  temporal.bandwidth,
  measure.points,
  measure.times,
  projection = NA,
  explicit = TRUE
)
```

Arguments

`dataset`	The dataset to be smoothed as a data.frame.
`dependent.variable`	String name of the single column in dataset with the factor dependent variable (if data.type=="factor") or a vector of column names with numeric counts (if data.type=="count") (defaults to "dependent.variable").
`x`	String name of column containing numeric x co-ordinate (defaults to "x").
`y`	String name of column containing numeric y co-ordinate (defaults to "y").
`time`	String name of the column in dataset with the time variable (defaults to "year").
`weight`	String name of column in the dataset with numeric weights (defaults to "weight").
`normalise.by`	String name of column by which data should be normalised (typically factor with document, speaker or writer ids).

`data.type` The type of the dependent variable as a string: either "factor", if each row is a token, or "count", if each row is a document, speaker or writer with token counts in separate columns (defaults to "factor").

`alpha` Numeric alpha for calculating error margins (defaults to 0.05).

`margin` Numeric desired error margin for calculating spatial bandwidths (defaults to 0.1).

`kernel.function`
The kernel function, one of gaussian.kernel, gaussian.square.kernel, triangular.kernel, square.kernel, or a custom function (defaults to gaussian.kernel).

`adaptive.spatial.bw`
Boolean indicating whether the spatial bandwidth is adaptive (set to achieve margin at every point) or static (set to the average of bandwidths needed to achieve margin at every point).

`temporal.bandwidth`
Numeric bandwidth of the (gaussian) temporal kernel.

`measure.points` A data.frame of spatial points at which estimates are to be made, with two columns with the same names as x,y in dataset; if not supplied, estimates are at the same locations as dataset.

`measure.times` A numeric vector of specific times at which to make estimates; if not given, will default to seq(from=min(time),to=max(time),length.out=5).

`projection` Spatial projection as a proj4 string - if given, data will be projected before smoothing and results will be deprojected before returning.

`explicit` If TRUE, progress will be reported with a progress bar (defaults to TRUE).

Value

A list containing the parameters and a data.frame with the smoothed estimates.

Examples

```
n=200;
synthesised.data<-data.frame(x=stats::runif(n),y=stats::runif(n),
    year=stats::runif(n,0,sqrt(2)));
synthesised.data$dependent.variable<-unlist(lapply(1:nrow(synthesised.data),
    function(X){
    stats::dist(as.matrix(synthesised.data[c(1,X),1:2]),method =
        "euclidean")<synthesised.data$year[X];
}));
result<-kernelPhil::kernel.smooth.in.space.and.time(dataset =
    synthesised.data,temporal.bandwidth = 0.25,measure.times =
    seq(from=-0.05,to=1.15,length.out=4),alpha = 0.15,margin = 0.2);
gridExtra::grid.arrange(ggplot2::ggplot(result$results[[1]],
    ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
    ggplot2::geom_point(),ggplot2::ggplot(result$results[[2]],
    ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
    ggplot2::geom_point(),ggplot2::ggplot(result$results[[3]],
    ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
    ggplot2::geom_point(),ggplot2::ggplot(result$results[[4]],
```

```
  ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
  ggplot2::geom_point());
```

`kernel.smooth.in.space.and.time.with.margins`
Kernel smooth data in space and time, returning specific error margins at each point

---

Description

This function performs kernel smoothing on a dataset in time and space. A static temporal kernel is applied first, and then an (optionally) adaptive spatial kernel on this weighted data. Note that this is the same as kernel.smooth.in.space.and.time() except that it returns specific error margins with every estimate and is *much* slower.

Usage

```
kernel.smooth.in.space.and.time.with.margins(
  dataset,
  dependent.variable = "dependent.variable",
  x = "x",
  y = "y",
  time = "year",
  weight = "weight",
  normalise.by,
  data.type = "factor",
  alpha = 0.05,
  margin = 0.1,
  kernel.function = gaussian.kernel,
  adaptive.spatial.bw = TRUE,
  temporal.bandwidth,
  measure.points,
  measure.times,
  projection = NA,
  explicit = TRUE
)
```

Arguments

`dataset`	The dataset to be smoothed as a data.frame.
`dependent.variable`	String name of the single column in dataset with the factor dependent variable (if data.type=="factor") or a vector of column names with numeric counts (if data.type=="count") (defaults to "dependent.variable").
`x`	String name of column containing numeric x co-ordinate (defaults to "x").
`y`	String name of column containing numeric y co-ordinate (defaults to "y").

time	String name of the column in dataset with the time variable (defaults to "year").
weight	String name of column in the dataset with numeric weights (defaults to "weight").
normalise.by	String name of column by which data should be normalised (typically factor with document, speaker or writer ids).
data.type	The type of the dependent variable as a string: either "factor", if each row is a token, or "count", if each row is a document, speaker or writer with token counts in separate columns (defaults to "factor").
alpha	Numeric alpha for calculating error margins (defaults to 0.05).
margin	Numeric desired error margin for calculating spatial bandwidths.
kernel.function	The kernel function, one of gaussian.kernel, gaussian.square.kernel, triangular.kernel, square.kernel, or a custom function (defaults to gaussian.kernel).
adaptive.spatial.bw	Boolean indicating whether the spatial bandwidth is adaptive (set to achieve margin at every point) or static (set to the average of bandwidths needed to achieve margin at every point).
temporal.bandwidth	Numeric bandwidth of the (gaussian) temporal kernel.
measure.points	A data.frame of spatial points at which estimates are to be made, with two columns with the same names as x,y in dataset; if not supplied, estimates are at the same locations as dataset.
measure.times	A numeric vector of specific times at which to make estimates; if not given, will default to seq(from=min(time),to=max(time),length.out=5).
projection	The spatial projection as a proj4 string - if given, data will be projected before smoothing and results will be deprojected before returning.
explicit	If TRUE, progress will be reported with a progress bar (defaults to TRUE).

Value

A list containing the parameters and a data.frame with the smoothed estimates.

Examples

```
n=200;
synthesised.data<-data.frame(x=stats::runif(n),y=stats::runif(n),
    year=stats::runif(n,0,sqrt(2)));
synthesised.data$dependent.variable<-unlist(lapply(1:nrow(synthesised.data),
    function(X){
    stats::dist(as.matrix(synthesised.data[c(1,X),1:2]),method =
        "euclidean")<synthesised.data$year[X];
}));
result<-kernelPhil::kernel.smooth.in.space.and.time.with.margins(dataset =
    synthesised.data,temporal.bandwidth = 0.2,measure.times =
    seq(from=0.15,to=0.85,length.out=2),alpha=0.4,margin=0.2);
gridExtra::grid.arrange(ggplot2::ggplot(result$results[[1]],
    ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
    ggplot2::geom_point(),ggplot2::ggplot(result$results[[2]],
```

```
ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
ggplot2::geom_point())
```

`kernel.smooth.in.space.with.margins`
Kernel smooth data in space alone, returning specific error margins at each point

---

Description

This function performs kernel smoothing on a dataset in space alone. It is the same as kernel.smooth.in.space(), except that the results include the error margins for the estimates at every point. Note that it is *much* slower than kernel.smooth.in.space().

Usage

```
kernel.smooth.in.space.with.margins(
  dataset,
  dependent.variable = "dependent.variable",
  x = "x",
  y = "y",
  weight = "weight",
  normalise.by,
  data.type = "factor",
  alpha = 0.05,
  margin = 0.1,
  kernel.function = gaussian.kernel,
  adaptive.spatial.bw = TRUE,
  measure.points,
  projection = NA,
  round.up.low.variance = TRUE,
  explicit = TRUE
)
```

Arguments

`dataset`	The dataset to be smoothed as a data.frame.
`dependent.variable`	String name of the single column in dataset with the factor dependent variable (if data.type=="factor") or a vector of column names with numeric counts (if data.type=="count") (defaults to "dependent.variable").
`x`	String name of column containing numeric x co-ordinate (defaults to "x").
`y`	String name of column containing numeric y co-ordinate (defaults to "y").
`weight`	String name of column in the dataset with numeric weights (defaults to "weight").
`normalise.by`	String name of column by which data should be normalised (typically factor with document, speaker or writer ids).

`data.type` The type of the dependent variable as a string: either "factor", if each row is a token, or "count", if each row is a document, speaker or writer with token counts in separate columns (defaults to "factor").

`alpha` Numeric alpha for calculating error margins (defaults to 0.05).

`margin` Numeric desired error margin for calculating spatial bandwidths (defaults to 0.1).

`kernel.function`
The kernel function, one of gaussian.kernel, gaussian.square.kernel, triangular.kernel, square.kernel, or a custom function (defaults to gaussian.kernel).

`adaptive.spatial.bw`
A boolean indicating whether the spatial bandwidth is adaptive (set to achieve margin at every point) or static (set to the average of bandwidths needed to achieve margin at every point).

`measure.points` A data.frame of spatial points at which estimates are to be made, with two columns with the same names as x,y in dataset; if not supplied, estimates are at the same locations as dataset.

`projection` The spatial projection as a proj4 string - if given, data will be projected before smoothing and results will be deprojected before returning.

`round.up.low.variance`
Set to TRUE if there are periods of time with extremely low variance (defaults to TRUE).

`explicit` If TRUE, progress will be reported with a progress bar (defaults to TRUE).

Value

A data.frame with the smoothed estimates.

Examples

```
n=400;
synthesised.data<-data.frame(x=stats::runif(n),y=stats::runif(n),
    year=stats::runif(n,0,sqrt(2)));
synthesised.data$dependent.variable<-unlist(lapply(1:nrow(synthesised.data),
    function(X){
    stats::dist(as.matrix(synthesised.data[c(1,X),1:2]),method =
        "euclidean")<synthesised.data$year[X];
}))
result<-kernelPhil::kernel.smooth.in.space.with.margins(dataset = synthesised.data);
ggplot2::ggplot(result,ggplot2::aes(x=x,y=y,colour=relative_density_TRUE))+
    ggplot2::geom_point();
```

`kernel.smooth.in.time` *Kernel smooth data in time alone*

Description

This function performs kernel smoothing on a dataset in time alone.

Usage

```
kernel.smooth.in.time(
  dataset,
  dependent.variable = "dependent.variable",
  time = "year",
  weight = "weight",
  bandwidth = 10,
  sample.density.threshold = 3,
  length.out = 1000,
  alpha = 0.05,
  xlabel = "year",
  ylabel,
  greyscale = "compatible",
  save.path = "",
  measure.times,
  kernel.function = gaussian.kernel
)
```

Arguments

`dataset`	The dataset to be smoothed as a data.frame.
`dependent.variable`	String name of the column in dataset with the dependent variable (defaults to "dependent.variable"); this column should be numeric or factor.
`time`	String name of the column in dataset with the time variable (defaults to "year").
`weight`	String name of column in the dataset with numeric weights (defaults to "weight").
`bandwidth`	Numeric bandwidth of the kernel function.
`sample.density.threshold`	Numeric local density of samples below which no estimates will be returned.
`length.out`	The number of measure points along the time axis (defaults to 1000).
`alpha`	Numeric alpha for calculating error margins (defaults to 0.05).
`xlabel`	String label for the x-axis in returned plot (defaults to "year").
`ylabel`	String label for the y-axis in returned plot.
`greyscale`	If TRUE, plot will be in greyscale; if "compatible", plot will use a colour spectrum which also goes light>dark; otherwise, will use a non-greyscale-compatible colour scale.

save.path	String path to save plot to (if not given, plot will not be saved).
measure.times	A numeric vector of specific times at which to make estimates; if given, sample.density.threshold and length.out will be ignored.
kernel.function	The kernel function, one of gaussian.kernel, gaussian.square.kernel, triangular.kernel, square.kernel, or a custom function.

Value

A list containing a data.frame with the smoothed estimates, and a ggplot grob visualising them.

Examples

```
n=1000;
synthesised.data<-data.frame(x=stats::runif(n),y=stats::runif(n),
    year=stats::runif(n,0,sqrt(2)));
synthesised.data$dependent.variable<-unlist(lapply(1:nrow(synthesised.data),
    function(X){
    stats::dist(as.matrix(synthesised.data[c(1,X),1:2]),method =
    "euclidean")<synthesised.data$year[X];
}))
result<-kernelPhil::kernel.smooth.in.time(dataset = synthesised.data,
    bandwidth = 0.05,sample.density.threshold = 100);
result$plot;
```

---

load.kernel.smooths *Load kernel smooths*

Description

Loads the output of kernel.smooth.in.space.and.time() or kernel.smooth.in.space.and.time.with.margins() previously saved with save.kernel.smooths()

Usage

```
load.kernel.smooths(location)
```

Arguments

location	String location to which results were saved.

Value

A list containing the parameters and a data.frame with the smoothed estimates (same structure as returned by kernel.smooth.in.space.and.time() and kernel.smooth.in.space.and.time.with.margins()).

nearest.point *Identify nearest point in time to a given estimate value*

Description

This function takes the output of kernel.smooth.in.time and identifies the point in time when the smoothed estimate comes closest to some specific value. This is useful for tasks like identifying the likely midpoint of a change.

Usage

```
nearest.point(kernel.smooths, density, variant, n = 1, timerange)
```

Arguments

kernel.smooths	A list output by kernel.smooth.in.time().
density	The value of the dependent variable for which a time is to be identified.
variant	If the dependent variable was a factor, which level is being examined (do not give a value if dependent variable was numeric).
n	The number of nearest points to be returned (useful if the estimates cross the relevant threshold multiple times, defauls to 1).
timerange	Numeric vector of length two - used to restrict search to a specific time range within the kernel.smooth.in.time(), in the form c(min,max).

Value

One or more numeric values.

save.kernel.smooths *Save kernel smooths in space and time*

Description

Saves the output of kernel.smooth.in.space.and.time() or kernel.smooth.in.space.and.time.with.margins() to a directory

Usage

```
save.kernel.smooths(kernel.smooth, location, variable.name)
```

Arguments

kernel.smooth	A list output of kernel.smooth.in.space.and.time() or kernel.smooth.in.space.and.time.with.margins().
location	String location on the disk to save output to.
variable.name	String name of the variable (used in filenames).

Value

No returned value, called to save data to disk.

Index