Package: PPTS (via r-universe)

May 29, 2026

Type Package

Title Point Process Time Series

Version 1.0

Date 2026-05-09

Author Daniel Gervini [aut, cre], Simon Kopischke [aut]

Maintainer Daniel Gervini <gervini@uwm.edu>

Description Provides functions for point process time series.
Autocorrelation functions for spatial and temporal time series,
and estimation of trend-plus-seasonality models for temporal
and spatial time series. See Gervini (2025)
<doi:10.1111/jtsa.70018> and Gervini and Kopischke (2026)
<doi:10.48550/arXiv.2605.21884>.

License MIT + file LICENSE

LazyData true

NeedsCompilation no

Depends R (>= 3.5.0)

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

Date/Publication 2026-05-29 12:30:17 UTC

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

RemoteRef HEAD

RemoteSha ff19d2e5c3f6d7f8e3436399dfd4cdd135d8f68f

Contents

`PPTS-package` *Provides functions for point process time series.*

Description

ACF for spatial and temporal time series, estimation of trend-only and trend-plus-seasonality for temporal and spatial time series, and respective confidence intervals.

Author(s)

Daniel Gervini

Maintainer: Daniel Gervini <gervini@uwm.edu>

`acf_sPP` *Autocorrelogram for spatial point-process time series*

Description

Spatial binned autocorrelograms.

Usage

```
acf_sPP(x, rng, kmax = 10*log10(length(x)), nbin = 9,
        alpha = 0.05, MC = 10000, no.plot = FALSE)
```

Arguments

`x`	(n-list) Observations (x[[i]] is a mi x 2 matrix, possibly mi=0, first column is longitude, second column is latitude)
`rng`	(4-vector) Spatial range of the process (first two numbers are longitude range, the other two are latitude range)
`kmax`	(int>=0) Maximum lag (default 10*log10(n))
`nbin`	(int>=4) Number of bins for binned estimator (default 9) (if not a perfect square integer, the closest perfect square is used)
`alpha`	(scalar) Significance test level
`MC`	(integer) Number of Monte Carlo replications for significance threshold computation (Default 10,000. Use MC=0 if no threshold computation is desired)
`no.plot`	(logical) Plot is shown if FALSE (default)

Value

lag	(kmax-vector) Lags at which autocorrelations are computed (1:kmax)
rho	(kmax-vector) Autocorrelations at lags 1:kmax
thr	(scalar) Significance threshold

Author(s)

Daniel Gervini

References

Gervini, D. (2025), "Autocorrelation functions for point-process time series", Journal of Time Series Analysis, DOI: 10.1111/jtsa.70018

Examples

```
data(crime)
acf_sPP(x=crime$x,rng=crime$range)
```

---

acf_tPP *Autocorrelogram for temporal point-process time series*

Description

Temporal binned autocorrelograms.

Usage

```
acf_tPP(x, rng, kmax = 10*log10(length(x)), nbin = 5,
        alpha = 0.05, MC = 10000, no.plot = FALSE)
```

Arguments

x	(n-list) Observations (x[[i]] is a vector of length mi, possibly mi=0)
rng	(2-vector) Time range of the process
kmax	(int>=0) Maximum lag (default 10*log10(n))
nbin	(int>=0) Number of bins for binned estimator (default 5)
alpha	(scalar) Significance test level
MC	(integer) Number of Monte Carlo replications for significance threshold computation (Default 10,000. Use MC=0 if no threshold computation is desired)
no.plot	(logical) Plot is shown if FALSE (default)

Value

lag	(kmax-vector) Lags at which autocorrelations are computed (1:kmax)
rho	(kmax-vector) Autocorrelations at lags 1:kmax
thr	(scalar) Significance threshold

Author(s)

Daniel Gervini

References

Gervini, D. (2025), "Autocorrelation functions for point-process time series", Journal of Time Series Analysis, DOI: 10.1111/jtsa.70018

Examples

```
data(bikes)
acf_tPP(x=bikes$x,rng=bikes$range)
```

---

bikes *Divvy Bike Rides*

Description

Daily bicycle check-out times (Jan 1, 2016 to Dec 31, 2016) at Ashland & Wrightwood station of the Divvy bike-sharing system in the city of Chicago.

Usage

```
data("bikes")
```

Format

A list with two elements: range (vector of length 2) and x (list of length 366).

Details

Vector bikes$range indicates time range used for some (but not all) analyses in the paper. List bikes$x contains the daily bike check-out times as vectors, with time given in hours in the [0,24) range.

Source

Chicago Data Portal, https://data.cityofchicago.org/

References

Gervini, D. (2025). Autocorrelation functions for point-process time series. Journal of Time Series Analysis. DOI: 10.1111/jtsa.70018.

Gervini, D. and Kopischke, S.A. (2026). Trend and seasonality estimation for point-process time series. DOI: 10.48550/arXiv.2605.21884

NSF DMS 2412015: "Statistical methods for point-process time series".

Examples

```
require(PPTS)
data(bikes)
oldpar <- par(no.readonly = TRUE)
fit <- trend_seas_t(x=bikes$x,rng=c(4,24),d=7,c.deg=3,sp.nk=5,maxit=10000)
par(mfrow=c(2,1),mar=c(4,4,1,1),mgp=c(2,1,0))
plot(fit$tt,exp(fit$ct),type="l",xlab="t (day)",ylab="exp(c(t))")
title(main="Multiplicative trend")
matplot(fit$u,exp(fit$mus),type="l",lty=1,xlab="u (hour)",ylab=expression(nu[j](u)))
title(main="Baseline seasonal intensities")
par(oldpar)
```

---

crime *Chicago Street Theft*

Description

Daily street theft locations in the North side of Chicago for the year 2014.

Usage

```
data("crime")
```

Format

A list with two elements: range (vector of length 4) and x (list of length 365).

Details

Vector crime$range (W longitude, E longitude, S latitude, N latitude) indicates spatial range used for the analyses in the paper.

List crime$x contains locations of daily incidents in matrix form (each row corresponds to a different incident, columns are longitude-latitude coordinates)

Source

Chicago Data Portal, https://data.cityofchicago.org/

References

Gervini, D. (2025). Autocorrelation functions for point-process time series. Journal of Time Series Analysis. DOI: 10.1111/jtsa.70018.

Gervini, D. and Kopischke, S.A. (2026). Trend and seasonality estimation for point-process time series. DOI: 10.48550/arXiv.2605.21884

NSF DMS 2412015: "Statistical methods for point-process time series".

Examples

```
require(PPTS)
data(crime)
fit <- trend_seas_s(x=crime$x,rng=crime$range,d=1,c.deg=3,sp.nk=3,maxit=10000)
plot(fit$t,exp(fit$c),type="l",,xlab="t (day)",ylab="exp(c(t))")
title(main="Multiplicative trend")
filled.contour(fit$u1,fit$u2,exp(fit$mu[,,1]))
title(main=expression(paste("Baseline intensity ",nu(u))),xlab="Longitude",ylab="Latitude")
```

`trend_seas_s` *Trend and seasonality estimation for spatial Point Process Time Series*

Description

Fits seasonal additive model $\mu(u,t) = c(t) + \mu_{j(t)}(u)$ to log-intensity process $\log(\Lambda_t(u))$, where $c(t)$ is a polynomial and $\mu_1(u), \ldots, \mu_d(u)$ are tensor-product splines.

Usage

```
trend_seas_s(x, rng, d = 1, r = length(x) - length(x)%%d, c.deg = 1,
             sp.nk = 1, sp.deg = 3, maxit = 100, conf = 0.95)
```

Arguments

`x`	(n-list) Observations (x[[i]] is a mi x 2 matrix, possibly mi=0, first column is longitude, second column is latitude)
`rng`	(4-vector) Spatial range of the process (first two numbers are longitude range, the other two are latitude range)
`d`	(integer) Seasonal period (use default 1 for non-seasonal model)
`r`	(integer) Period of trend c(t) (must be a multiple of d; default is n-n%%d)
`c.deg`	(integer) Degree of polynomial c(t)
`sp.nk`	(integer) Number of spline knots per dimension for mu_j(u)
`sp.deg`	(integer) Spline degree for mu_j(u)
`maxit`	(integer) Maximum number of iterations
`conf`	(scalar) Confidence interval level (default is 0.95)

Value

u1	(ng-vector) Grid for evaluation of mu_j(u) (longitude)
u2	(ng-vector) Grid for evaluation of mu_j(u) (latitude)
mus	(ng x ng x d -array) Seasonal means mu_j(u) evaluated at u1 x u2 (mus[i,k,j] = mu_j(u1[i],u2[k]))
t	(vector) Grid for evaluation of c(t) (t = 1:n)
c	(vector) Trend c(t) evaluated at t
tt	(vector) Grid for evaluation of one cycle of c(t) (tt = 1:r)
ct	(vector) One cycle of trend c(t) evaluated at tt
eta	(vector) Basis coefficients for c(t)
thetas	(matrix) Basis coefficients for mu_j(u) (in rows)
ub_ct	(vector) Upper conf bound for c(t)
lb_ct	(vector) Lower conf bound for c(t)
ub_ect	(vector) Upper conf bound for exp(c(t))
lb_ect	(vector) Lower conf bound for exp(c(t))
ub_mus	(ng x ng x d -array) Upper conf bound for mu_j(u)'s evaluated at u1 x u2
lb_mus	(ng x ng x d -array) Lower conf bound for mu_j(u)'s evaluated at u1 x u2
ub_lmb	(ng x ng x d -array) Upper conf bound for exp(mu_j(u))'s evaluated at u1 x u2
lb_lmb	(ng x ng x d -array) Lower conf bound for exp(mu_j(u))'s evaluated at u1 x u2
Omega	(matrix) Covariance matrix of joint parameters

Author(s)

Daniel Gervini

References

Gervini, D., and Kopischke, S.A. (2006), "Trend and seasonality estimation for point-process time series." DOI: 10.48550/arXiv.2605.21884

Examples

```
data(crime)
fit <- trend_seas_s(x=crime$x,rng=crime$range,d=7,c.deg=3,sp.nk=3)
plot(fit$t,fit$c,type="l")
filled.contour(fit$u1,fit$u2,exp(fit$mus[,,1]))
```

trend_seas_t — *Trend and seasonality estimation for temporal Point Process Time Series*

Description

Fits seasonal additive model $\mu(u,t) = c(t) + \mu_{j(t)}(u)$ to log-intensity process $\log(\Lambda_t(u))$, where $c(t)$ is a polynomial and $\mu_1(u),\dots,\mu_d(u)$ are splines.

Usage

```
trend_seas_t(x, rng, d = 1, r = length(x) - length(x)%%d, c.deg = 1,
             sp.nk = 1, sp.deg = 3, maxit = 100, conf = 0.95)
```

Arguments

`x`	(n-list) Observations (x[[i]] is a vector of length mi, possibly mi=0)
`rng`	(vector, length 2) Time range of the process
`d`	(integer) Seasonal period (use default 1 for non-seasonal model)
`r`	(integer) Period of trend c(t) (must be a multiple of d; default is n-n%%d)
`c.deg`	(integer) Degree of polynomial c(t)
`sp.nk`	(integer) Number of spline knots for mu_j(u)
`sp.deg`	(integer) Spline degree for mu_j(u)
`maxit`	(integer) Maximum number of iterations
`conf`	(scalar) Confidence interval level (default is 0.95)

Value

`u`	(vector) Grid for evaluation of mu_j(u)
`mus`	(matrix) Seasonal means mu_j(u) (in columns) evaluated at u
`t`	(vector) Grid for evaluation of c(t) (t = 1:n)
`c`	(vector) Trend c(t) evaluated at t
`tt`	(vector) Grid for evaluation of one cycle of c(t) (tt = 1:r)
`ct`	(vector) One cycle of trend c(t) evaluated at tt
`eta`	(vector) Basis coefficients for c(t)
`thetas`	(matrix) Basis coefficients for mu_j(u) (in rows)
`ub_ct`	(vector) Upper conf bound for c(t)
`lb_ct`	(vector) Lower conf bound for c(t)
`ub_ect`	(vector) Upper conf bound for exp(c(t))
`lb_ect`	(vector) Lower conf bound for exp(c(t))
`ub_mus`	(matrix) Upper conf bound for mu_j(u) (in columns)

lb_mus	(matrix) Lower conf bound for mu_j(u) (in columns)
ub_lmb	(matrix) Upper conf bound for exp(mu_j(u)) (in columns)
lb_lmb	(matrix) Lower conf bound for exp(mu_j(u)) (in columns)
Omega	(matrix) Covariance matrix of joint parameters

Author(s)

Daniel Gervini

References

Gervini, D., and Kopischke, S.A. (2006), "Trend and seasonality estimation for point-process time series." DOI: 10.48550/arXiv.2605.21884

Examples

```
data(bikes)
fit <- trend_seas_t(x=bikes$x,rng=bikes$range,d=7,c.deg=3,sp.nk=5)
plot(fit$tt,fit$ct,type="l")
matplot(fit$u,exp(fit$mus),type="l",lty=1)
```

Index