

Package: SDALGCP2 (via r-universe)

July 2, 2026

Title Fast Spatially Discrete Approximation to Log-Gaussian Cox Processes for Aggregated Disease Count Data

Version 0.1.0

Description Fits a spatially discrete approximation to a log-Gaussian Cox process model for spatially aggregated disease count data, estimated by Monte Carlo Maximum Likelihood as in Christensen (2004) <[doi:10.1198/106186004X2525](https://doi.org/10.1198/106186004X2525)> and Johnson, Diggle and Giorgi (2019) <[doi:10.1002/sim.8339](https://doi.org/10.1002/sim.8339)>. Performance-critical steps (aggregated correlation assembly, 'MALA' sampling, the Monte Carlo likelihood, and the Kronecker-structured space-time likelihood) are implemented in C++ via 'RcppArmadillo'. Provides a one-line, 'glm'-like interface and statistical extensions including a nugget term, general 'Matern' smoothness, raster and misaligned covariates, restricted spatial regression, importance-sampling diagnostics and re-anchored 'MCML'.

Depends R (>= 4.2.0)

License GPL-2 | GPL-3

Encoding UTF-8

Language en-GB

LazyData true

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, sf, terra, spatstat.geom, spatstat.random, ggplot2, progress, stats, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0), numDeriv, bench

VignetteBuilder knitr

Config/testthat/edition 3

RoxygenNote 7.3.1

URL <https://github.com/olatunjijohnson/SDALGCP2>,
<https://olatunjijohnson.github.io/SDALGCP2/>

BugReports <https://github.com/olatunjijohnson/SDALGCP2/issues>

NeedsCompilation yes

Author Olatunji Johnson [aut, cre], Emanuele Giorgi [aut], Peter Diggle [aut]

Maintainer Olatunji Johnson <olatunjijohnson21111@gmail.com>

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

Date/Publication 2026-07-02 18:40:25 UTC

RemoteUrl <https://github.com/cran/SDALGCP2>

RemoteRef HEAD

RemoteSha cc2e16b6ed9c3351d616d42ce0045f2f9bcca979

Contents

coef_plot	3
confint.SDALGCP2	3
control_mcmc	4
exceedance	5
laplace_sampling	6
liver	7
map_exceedance	8
mc_diagnostics	9
mcml_fit	9
model_check	11
phi_profile	12
plot.sdalgcp	12
plot.SDALGCP2	14
plot.SDALGCP2_pred	14
plot.SDALGCP2_ST_pred	15
precompute_corr	17
predict.sdalgcp	18
predict.SDALGCP2	19
predict.SDALGCP2_ST	20
print.SDALGCP2	21
print.summary.SDALGCP2	22
report	22
sda_points	23
sdalgcp	24
sdalgcp_control	25
sdalgcp_data	27
SDALGCP2	28
SDALGCP2_misaligned	30
SDALGCP2_raster	31
SDALGCP2_ST	33
summary.SDALGCP2	35

Index

37

coef_plot	<i>Coefficient plot of fixed effects (and sigma²) with confidence intervals</i>
-----------	--

Description

Coefficient plot of fixed effects (and sigma²) with confidence intervals

Usage

```
coef_plot(object, level = 0.95, intercept = FALSE)
```

Arguments

object	a fitted "SDALGCP2" object.
level	confidence level.
intercept	logical; include the intercept.

Value

a ggplot object.

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
coef_plot(fit)
```

confint.SDALGCP2	<i>Wald confidence intervals for an SDALGCP2 fit</i>
------------------	--

Description

Wald confidence intervals for an SDALGCP2 fit

Usage

```
## S3 method for class 'SDALGCP2'
confint(object, parm, level = 0.95, ...)
```

Arguments

object	an object of class "SDALGCP2".
parm	parameters to report (names or indices); default all.
level	confidence level.
...	unused.

Value

a matrix of lower/upper confidence limits.

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
confint(fit)
```

control_mcmc

MCMC control settings for the MALA sampler

Description

MCMC control settings for the MALA sampler

Usage

```
control_mcmc(
  n.sim = 10000,
  burnin = 2000,
  thin = 8,
  h = NULL,
  c1.h = 0.01,
  c2.h = 1e-04
)
```

Arguments

n.sim	total number of iterations.
burnin	burn-in iterations to discard.
thin	thinning interval; (n.sim - burnin) must be a multiple.
h	initial Langevin step size; if missing, $1.65 / d^{(1/6)}$ is used.
c1.h, c2.h	step-size adaptation constants.

Value

a named list consumed by [laplace_sampling](#) / the fit.

Examples

```
## 1000 retained draws (5000 iterations, 2000 burn-in, thin every 3)
ctrl <- control_mcmc(n.sim = 5000, burnin = 2000, thin = 3)
str(ctrl)
```

exceedance	<i>Exceedance probabilities $P(\text{risk} > \text{threshold})$</i>
------------	---

Description

Exceedance probabilities $P(\text{risk} > \text{threshold})$

Usage

```
exceedance(object, thresholds, which = c("adjusted_rr", "relative_risk"))
```

Arguments

object	an "SDALGCP2_pred" object from predict.SDALGCP2 .
thresholds	numeric vector of thresholds.
which	which quantity: "adjusted_rr" (the covariate-adjusted relative risk $\exp(S)$, default) or "relative_risk" (the relative risk $\exp(d'\beta + S)$).

Value

a matrix of exceedance probabilities (locations x thresholds).

See Also

[map_exceedance](#) to map them.

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
pr <- predict(fit, type = "discrete")
## P(adjusted relative risk > 1) and > 1.5 for every region
ex <- exceedance(pr, thresholds = c(1, 1.5), which = "adjusted_rr")
head(ex)
```

laplace_sampling	<i>Sample the latent field $[S Y]$ (Poisson, non-nested) via C++ MALA</i>
------------------	--

Description

Draws posterior samples of the latent Gaussian field for the Poisson, non-nested case. The Laplace mode (Newton step) and the adaptive Metropolis- adjusted Langevin (MALA) loop both run in C++ for speed, with a fixed-seed path for reproducibility.

Usage

```
laplace_sampling(mu, Sigma, y, units.m, control.mcmc)
```

Arguments

mu	prior mean vector.
Sigma	prior covariance matrix.
y	count vector.
units.m	offset vector.
control.mcmc	list from control_mcmc .

Value

list with samples (kept x n matrix) and h (step sizes).

Examples

```
## sample [S | Y] for a tiny 10-unit Poisson example
set.seed(1)
n <- 10
D <- as.matrix(dist(cbind(runif(n), runif(n))))
Sigma <- 0.4 * exp(-D / 0.3)
mu <- rep(log(2), n); m <- rep(100, n)
y <- rpois(n, m * exp(mu + as.numeric(t(chol(Sigma)) %*% rnorm(n))))
out <- laplace_sampling(mu, Sigma, y, m, control_mcmc(n.sim = 2000, burnin = 500, thin = 3))
dim(out$samples)      # (retained draws) x n
```

`liver`*Primary biliary cirrhosis incidence in North East England*

Description

A real aggregated disease-count dataset: incident primary biliary cirrhosis (a chronic liver disease) cases by Lower-layer Super Output Area (LSOA) in the Newcastle and Gateshead area of North East England, with population and area deprivation covariates. This is the case study of Johnson et al. (2019) and a realistic test bed for the spatial model: $\text{cases} \sim \text{deprivation} + \text{offset}(\log(\text{pop}))$.

Usage

`liver`

Format

An `sf` object of 545 LSOA polygons (British National Grid, EPSG:27700) with columns:

lsoa LSOA 2004 census code.

cases observed incident case count in the LSOA.

pop population at risk (the offset; use `offset(log(pop))`).

IMD Index of Multiple Deprivation score (higher = more deprived).

Income income-deprivation score.

Employment employment-deprivation score.

geometry the LSOA polygon.

Source

Johnson, O., Diggle, P. and Giorgi, E. (2019), "A spatially discrete approximation to log-Gaussian Cox processes for modelling aggregated disease count data", *Statistics in Medicine*, 38(24), 4871-4884. doi:10.1002/sim.8339. Population and area-deprivation covariates are from the 2004 English indices of deprivation (Lower-layer Super Output Area level). See `data-raw/liver.R` in the package sources.

See Also

[sdalgcp_data](#) for a small simulated example.

Examples

```
data(liver)
summary(liver$cases)
plot(liver["IMD"])
```

map_exceedance	<i>Map exceedance probabilities $P(\text{risk} > \text{threshold})$</i>
----------------	---

Description

Map exceedance probabilities $P(\text{risk} > \text{threshold})$

Usage

```
map_exceedance(  
  x,  
  threshold = 1,  
  which = c("adjusted_rr", "relative_risk"),  
  bound = NULL,  
  ...  
)
```

Arguments

x	an "SDALGCP2_pred" object.
threshold	a single relative-risk threshold.
which	"adjusted_rr" (covariate-adjusted, default) or "relative_risk".
bound	optional sf boundary (continuous only).
...	unused.

Value

a ggplot object.

See Also

[exceedance](#) for the underlying probabilities.

Examples

```
data(sdalgcp_data)  
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,  
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,  
                                       reanchor = 0))  
pr <- predict(fit, type = "discrete")  
map_exceedance(pr, threshold = 1.5)           # P(adjusted RR > 1.5)
```

mc_diagnostics

Importance-sampling diagnostics for an MCML fit

Description

The MCML estimate reweights latent samples drawn at the anchor towards the optimum. When the optimum is far from the anchor the weights become uneven and the estimate unreliable. This reports the effective sample size of the importance weights at the maximiser and a Monte Carlo standard error for the maximised log-likelihood, $SE \approx \sqrt{1/ESS - 1/B}$.

Usage

```
mc_diagnostics(object, warn_frac = 0.1)
```

Arguments

`object` a fitted "SDALGCP2" object.
`warn_frac` warn if the ESS falls below this fraction of B .

Value

invisibly, a list with `B`, `ESS`, `ESS_frac` and `se_loglik`.

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
d <- mc_diagnostics(fit)
d$ESS_frac # importance-sampling ESS as a fraction of the draws
```

mcml_fit

Monte Carlo maximum likelihood estimation for the spatial SDA-LGCP

Description

Vectorised, Cholesky-based MCML estimation. Simulates the latent field at an anchor, then profiles the importance-sampling MCML objective over the supplied phi grid.

Usage

```
mcml_fit(
  formula,
  data,
  corr,
  par0 = NULL,
  control.mcmc = NULL,
  phi_method = c("grid", "direct"),
  nugget = FALSE,
  reanchor = 0L,
  reanchor_tol = 0.01,
  messages = FALSE
)
```

Arguments

formula	model formula, optionally with an <code>offset()</code> term.
data	data frame holding the model variables.
corr	list with R ($N \times N \times n_phi$ correlation array) and ϕ , e.g. from precompute_corr .
par0	optional starting values <code>c(beta, sigma2, phi)</code> ; if NULL they are derived from a Poisson GLM.
control.mcmc	list from control_mcmc (defaults if NULL).
phi_method	"grid" (profile over the corr phi grid, default) or "direct" (optimise phi continuously; exponential/Matern kernel).
nugget	logical; if TRUE (requires <code>phi_method = "direct"</code>) add a relative nugget, fitting covariance $\sigma^2(R(\phi) + \nu I)$.
reanchor	number of re-anchoring passes (re-simulate the latent field at the current optimum and refit) to raise the importance-sampling ESS.
reanchor_tol	relative-change tolerance for stopping the re-anchoring loop.
messages	logical; print optimiser progress.

Value

an object of class "SDALGCP2" (estimates, covariance, profile, latent samples and metadata).

See Also

[SDALGCP2](#) (the end-to-end wrapper), [precompute_corr](#)

Examples

```
data(sdalgcp_data)
df <- sf::st_drop_geometry(sdalgcp_data)
pts <- sda_points(sdalgcp_data, delta = 1.2, method = 3)
cc <- precompute_corr(pts, phi = seq(2, 8, length.out = 6))
fit <- mcml_fit(cases ~ x1 + offset(log(pop)), df, cc,
```

```
summary(fit) control.mcmc = control_mcmc(n.sim = 2000, burnin = 500, thin = 5))
```

 model_check

Posterior-predictive model checking for an SDALGCP2 fit

Description

Compares observed counts with fitted Poisson means, returns Pearson residuals, and tests them for residual spatial autocorrelation with Moran's I. A non-significant Moran's I indicates the spatial random effect has absorbed the spatial structure.

Usage

```
model_check(object, pred = NULL, nsim = 999, plot = TRUE)
```

Arguments

object	a fitted "SDALGCP2" object.
pred	a discrete prediction from <code>predict(object, "discrete")</code> ; if NULL one is computed with the fitting MCMC controls.
nsim	permutations for the Moran's I p-value.
plot	logical; draw the observed-vs-fitted scatter.

Value

invisibly, a list with fitted, residuals and moran.

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
chk <- model_check(fit, plot = FALSE)
chk$moran          # residual Moran's I and its permutation p-value
```

phi_profile	<i>Profile likelihood and confidence interval for the spatial scale phi</i>
-------------	---

Description

Spline-smoothed profile deviance for phi, with the coverage-level confidence interval where the deviance crosses the chi-squared cutoff.

Usage

```
phi_profile(object, coverage = 0.95, plot = TRUE)
```

Arguments

object	a fitted "SDALGCP2" object.
coverage	confidence level.
plot	logical; draw the deviance curve.

Value

invisibly, a list with the interval and the smoothed profile; a ggplot is drawn when plot = TRUE.

Examples

```
data(sdalgcp_data)
## profile phi on a grid (scale = "grid") so there is a deviance curve to draw
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
               control = sdalgcp_control(scale = "grid", n_sim = 2000,
                                         burnin = 500, thin = 5, reanchor = 0))
phi_profile(fit)
```

plot.sdalgcp	<i>Map an sdalgcp fit</i>
--------------	---------------------------

Description

Predicts and maps a chosen quantity. Works for spatial fits (discrete or continuous) and spatio-temporal fits (select a time).

Usage

```
## S3 method for class 'sdalgcp'
plot(
  x,
  what = c("relative_risk", "adjusted_rr", "relative_risk_se", "adjusted_rr_se",
           "exceedance"),
  type = c("discrete", "continuous"),
  time = NULL,
  threshold = 1,
  which = c("adjusted_rr", "relative_risk"),
  cellsize = NULL,
  sampler = c("mcmc", "laplace"),
  ...
)
```

Arguments

x	an "sdalgcp" fit.
what	one of "relative_risk" (relative risk, default), "adjusted_rr" (covariate-adjusted relative risk), "relative_risk_se", "adjusted_rr_se" or "exceedance".
type	"discrete" (default) or "continuous" (spatial fits).
time	for spatio-temporal fits, the time to map (default: first; use NULL to facet all times).
threshold	threshold for what = "exceedance".
which	for exceedance: "adjusted_rr" (default) or "relative_risk".
cellsize	grid spacing for type = "continuous".
sampler	"mcmc" (default) or "laplace".
...	passed to the mapping layer.

Value

a ggplot object.

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
plot(fit) # relative-risk map (predicts internally)
plot(fit, what = "exceedance", threshold = 1.5)
```

plot.SDALGCP2 *Plot an SDALGCP2 fit (the phi profile deviance)*

Description

Plot an SDALGCP2 fit (the phi profile deviance)

Usage

```
## S3 method for class 'SDALGCP2'
plot(x, ...)
```

Arguments

x an "SDALGCP2" object.
 ... passed to [phi_profile](#).

Value

invisibly, the profile (see [phi_profile](#)).

Examples

```
data(sdalgcp_data)
fit <- SDALGCP2(cases ~ x1 + offset(log(pop)),
               sf::st_drop_geometry(sdalgcp_data), sdalgcp_data, delta = 1.2,
               control.mcmc = control_mcmc(n.sim = 2000, burnin = 500, thin = 5))
plot(fit) # profile deviance for the spatial scale phi
```

plot.SDALGCP2_pred *Map a fitted SDALGCP2 prediction*

Description

Maps any of the four predicted quantities from [predict.SDALGCP2](#) – the relative risk "relative_risk", the covariate-adjusted relative risk "adjusted_rr", or their standard errors "relative_risk_se"/"adjusted_rr_se" – for either discrete (choropleth) or continuous (raster) predictions.

Usage

```
## S3 method for class 'SDALGCP2_pred'
plot(
  x,
  variable = c("relative_risk", "adjusted_rr", "relative_risk_se", "adjusted_rr_se"),
  bound = NULL,
  midpoint = NULL,
  title = NULL,
  ...
)
```

Arguments

<code>x</code>	an object of class "SDALGCP2_pred".
<code>variable</code>	one of "relative_risk", "adjusted_rr", "relative_risk_se", "adjusted_rr_se".
<code>bound</code>	optional sf boundary; continuous surfaces are masked to it and its outline overlaid.
<code>midpoint</code>	optional value to centre a diverging colour scale (defaults to 1 for the relative-risk columns, none for the standard errors).
<code>title</code>	optional plot title.
<code>...</code>	unused.

Value

a ggplot object.

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))

pr <- predict(fit, type = "discrete")
plot(pr, variable = "relative_risk")           # choropleth of relative risk
plot(pr, variable = "adjusted_rr_se")        # its uncertainty
```

plot.SDALGCP2_ST_pred *Map a spatio-temporal prediction for one time*

Description

Maps a chosen quantity ("relative_risk", "adjusted_rr", "relative_risk_se", "adjusted_rr_se" or "exceedance") for a selected time slice of a spatio-temporal prediction.

Usage

```
## S3 method for class 'SDALGCP2_ST_pred'
plot(
  x,
  time = attr(x, "times")[1],
  what = c("relative_risk", "adjusted_rr", "relative_risk_se", "adjusted_rr_se",
    "exceedance"),
  threshold = 1,
  which = c("adjusted_rr", "relative_risk"),
  ...
)
```

Arguments

x	an "SDALGCP2_ST_pred" object from predict() on an "SDALGCP2_ST" fit.
time	the time to map (one of the fitted times); defaults to the first. Use NULL to facet all times.
what	one of "relative_risk", "adjusted_rr", "relative_risk_se", "adjusted_rr_se", "exceedance".
threshold	threshold for what = "exceedance".
which	for exceedance: "adjusted_rr" (default) or "relative_risk".
...	unused.

Value

a ggplot object.

Examples

```
data(sdalgcp_data)
times <- 1:3
panel <- do.call(rbind, lapply(times, function(t) {
  d <- sdalgcp_data; d$time <- t
  d$cases <- rpois(nrow(d), d$pop * exp(-6 + 0.6 * d$x1 + 0.1 * (t - 2)))
  d
}))
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = panel, time = "time",
  control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
    reanchor = 0))

pr <- predict(fit)
plot(pr, time = 2)           # one time slice
plot(pr, time = NULL)       # facet all times
plot(pr, what = "exceedance", threshold = 1.2, time = 3)
```

```
precompute_corr
```

Precompute aggregated region-level correlation matrices

Description

Builds the $N \times N \times \text{length}(\text{phi})$ array of region-level correlations used by the SDA-LGCP model, where

$$R(\phi)_{ij} = \sum_{k,l} w_{ik} w_{jl} C(\|x_{ik} - x_{jl}\|; \phi, \kappa)$$

(population-weighted) or the unweighted mean over candidate-point pairs. The heavy reduction runs in C++ (OpenMP-parallel over region pairs).

Usage

```
precompute_corr(points, phi, kappa = 0.5, weighted = NULL, nthreads = 0L)
```

Arguments

points	a list of length N ; each element holds $\$xy$ (an $n_i \times 2$ matrix of candidate-point coordinates) and, when weighted, $\$weight$ (a length- n_i vector summing to 1). The "weighted" and "my_shp" attributes produced by the point-generation step are honoured and carried through.
phi	numeric vector of spatial scale parameters.
kappa	Matern smoothness; 0.5 (exponential, default), 1.5 or 2.5 use closed forms in C++.
weighted	logical; if NULL (default) it is taken from <code>attr(points, "weighted")</code> .
nthreads	number of OpenMP threads; 0 (default) uses the OpenMP runtime default.

Value

a list with R (the correlation array) and phi, carrying weighted, my_shp and S_coord attributes on R.

See Also

[sda_points](#), [mcm1_fit](#)

Examples

```
data(sdalgcp_data)
pts <- sda_points(sdalgcp_data, delta = 1.2, method = 3)
cc <- precompute_corr(pts, phi = c(2, 4, 6))
dim(cc$R)           # N x N x length(phi)
```

predict.sdalgcp *Predict relative risk from an sdalgcp fit*

Description

Returns a prediction object carrying, for every location, the posterior mean and standard error of the relative risk `relative_risk` ($\exp(\eta) = \exp(d'\beta + S)$) and the covariate-adjusted relative risk `adjusted_rr` ($\exp(S)$). Map it with `plot()` and get hotspot probabilities with [exceedance](#).

Usage

```
## S3 method for class 'sdalgcp'
predict(
  object,
  type = c("discrete", "continuous"),
  sampler = c("mcmc", "laplace"),
  cellsize = NULL,
  ...
)
```

Arguments

<code>object</code>	an "sdalgcp" fit.
<code>type</code>	"discrete" (region level, default) or "continuous" (a grid surface). Ignored for spatio-temporal fits.
<code>sampler</code>	"mcmc" (default) or "laplace".
<code>cellsize</code>	grid spacing for <code>type = "continuous"</code> .
<code>...</code>	passed to the underlying predictor.

Value

for a spatial fit, an `sf` of class "SDALGCP2_pred" with `relative_risk`, `relative_risk_se`, `adjusted_rr` and `adjusted_rr_se` columns (polygons for `type = "discrete"`, grid points for "continuous");
 for a spatio-temporal fit, an "SDALGCP2_ST_pred" object (see [predict.SDALGCP2_ST](#)).

Examples

```
data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
pr <- predict(fit)                    # discrete by default; an sf of relative risks
head(pr)
```

predict.SDALGCP2 *Predict relative risk from a fitted SDALGCP2 model*

Description

Predict relative risk from a fitted SDALGCP2 model

Usage

```
## S3 method for class 'SDALGCP2'
predict(
  object,
  type = c("discrete", "continuous"),
  sampler = c("mcmc", "laplace"),
  cellsize = NULL,
  pred.loc = NULL,
  control.mcmc = NULL,
  ...
)
```

Arguments

object	an object of class "SDALGCP2" from SDALGCP2 or mcmc1_fit .
type	"discrete" for region-level inference or "continuous" for a spatially continuous surface.
sampler	"mcmc" (MALA, default) or "laplace" (fast Gaussian approximation, no MCMC).
cellsize	grid spacing for continuous prediction (ignored if pred.loc supplied).
pred.loc	optional data frame of prediction coordinates (x, y) for continuous prediction.
control.mcmc	optional MCMC controls; defaults to those used at fitting.
...	unused.

Value

an sf (class c("SDALGCP2_pred", "sf", "data.frame")) with one row per location – polygons for type = "discrete", grid-cell points for type = "continuous" – carrying the posterior mean and standard error of two relative-risk quantities:

relative_risk, relative_risk_se the relative risk $\exp(d'\beta + S)$ – the fitted risk relative to the offset baseline, combining the covariate effect and the residual spatial variation. This is the headline disease-mapping quantity.

adjusted_rr, adjusted_rr_se the covariate-adjusted relative risk $\exp(S)$ – the purely spatial relative risk that remains after holding the covariates fixed (the spatial signal the covariates do not explain).

The full posterior draws are retained as object attributes so that [exceedance](#) and [map_exceedance](#) can be computed for either quantity. Map a column with [plot.SDALGCP2_pred](#).

Examples

```

data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))

## region-level (discrete) prediction: an sf you can map or st_write()
pr <- predict(fit, type = "discrete")
head(pr) # relative_risk / adjusted_rr (+ standard errors)
plot(pr, variable = "relative_risk")

## continuous surface on a grid
pr_c <- predict(fit, type = "continuous", cellsize = 1)

```

predict.SDALGCP2_ST *Discrete (region x time) prediction for a spatio-temporal fit*

Description

Draws the latent field at the fitted optimum and returns posterior mean and SD of the incidence relative risk $\exp(\mu + S)$ and covariate-adjusted relative risk $\exp(S)$ for every region and time.

Usage

```

## S3 method for class 'SDALGCP2_ST'
predict(object, control.mcmc = NULL, ...)

```

Arguments

object	an "SDALGCP2_ST" fit.
control.mcmc	optional MCMC controls (defaults to the fitting ones).
...	unused.

Value

a long `sf` of class `c("SDALGCP2_ST_pred", "sf", "data.frame")` with one row per region and time (ordered region-fastest within each time block) and columns `region`, `time`, `relative_risk`, `relative_risk_se` ($\exp(\mu + S)$), `adjusted_rr` and `adjusted_rr_se` ($\exp(S)$) – the same column names as the spatial `predict.SDALGCP2`. The posterior draws are kept in object attributes (for exceedance); map a time slice with `plot.SDALGCP2_ST_pred`.

Examples

```

data(sdalgcp_data)
## stack the spatial example into a 3-time panel with a mild temporal trend
times <- 1:3
panel <- do.call(rbind, lapply(times, function(t) {
  d <- sdalgcp_data; d$time <- t
  d$cases <- rpois(nrow(d), d$pop * exp(-6 + 0.6 * d$x1 + 0.1 * (t - 2)))
  d
}))
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = panel, time = "time",
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))
pr <- predict(fit)      # a long sf: region x time
head(pr)
plot(pr, time = 2)     # map the relative risk at time 2

```

```

print.SDALGCP2      Print an SDALGCP2 fit

```

Description

Print an SDALGCP2 fit

Usage

```

## S3 method for class 'SDALGCP2'
print(x, ...)

```

Arguments

x	an "SDALGCP2" object.
...	unused.

Value

x, invisibly.

```
print.summary.SDALGCP2
```

Print a summary of an SDALGCP2 fit

Description

Print a summary of an SDALGCP2 fit

Usage

```
## S3 method for class 'summary.SDALGCP2'
print(x, ...)
```

Arguments

x	a "summary.SDALGCP2" object.
...	unused.

Value

x, invisibly.

```
report
```

One-call panel of post-fit graphics

Description

Returns the maps and summaries an analyst usually wants after fitting: relative-risk and uncertainty maps, an exceedance map, the coefficient plot and the phi profile. The pieces are returned as a named list of ggplot objects so they can be arranged or printed individually.

Usage

```
report(object, pred = NULL, threshold = 1.5, ...)
```

Arguments

object	a fitted "SDALGCP2" object.
pred	optional discrete prediction; computed if NULL.
threshold	relative-risk threshold for the exceedance map.
...	passed to <code>predict.SDALGCP2</code> when pred is computed.

Value

a named list of ggplot objects.

Examples

```

data(sdalgcp_data)
fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = sdalgcp_data,
              control = sdalgcp_control(n_sim = 2000, burnin = 500, thin = 5,
                                       reanchor = 0))

figs <- report(fit, threshold = 1.5)
names(figs) # relative_risk, uncertainty, exceedance, coefficients, ...
figs$relative_risk # print one of the maps

```

sda_points	<i>Generate candidate sampling points inside each region</i>
------------	--

Description

For every polygon feature in `my_shp` it produces candidate points and aggregation weights, in the list format consumed by [precompute_corr](#).

Usage

```

sda_points(
  my_shp,
  delta,
  method = 1L,
  weighted = FALSE,
  pop_shp = NULL,
  rho = 0.55,
  giveup = 1000L
)

```

Arguments

<code>my_shp</code>	an <code>sf</code> object of POLYGON/MULTIPOLYGON features.
<code>delta</code>	point spacing (grid step / SSI inhibition distance).
<code>method</code>	1 = SSI (default), 2 = uniform random, 3 = regular grid.
<code>weighted</code>	logical; if TRUE, weights are population density read from <code>pop_shp</code> , otherwise equal weights.
<code>pop_shp</code>	a <code>terra::SpatRaster</code> of population density (required when <code>weighted = TRUE</code>).
<code>rho</code>	packing density used to choose the number of points.
<code>giveup</code>	SSI rejection limit.

Value

a list of length `nrow(my_shp)`; each element has `xy` and `weight`. Carries "weighted" and "my_shp" attributes.

See Also

[precompute_corr](#), which consumes this output.

Examples

```
data(sdalgcp_data)
pts <- sda_points(sdalgcp_data, delta = 1.2, method = 3) # regular grid points
length(pts)      # one entry per region
str(pts[[1]])    # $xy candidate coordinates and $weight
```

sdalgcp

Fit a spatially discrete LGCP model for aggregated counts

Description

The main user interface, designed to feel like [glm](#): give a formula and an `sf` data object and it does the rest. The same call covers three settings, chosen from the arguments you supply:

- **spatial** (default): `sdalgcp(y ~ x + offset(log(pop)), data)`;
- **raster covariates**: add `rasters = a SpatRaster` whose layers are named in the formula – these enter on the intensity scale (see [SDALGCP2_raster](#));
- **spatio-temporal**: add `time = the name of a time column`.

Usage

```
sdalgcp(
  formula,
  data,
  time = NULL,
  rasters = NULL,
  covariates = NULL,
  popden = NULL,
  control = sdalgcp_control(),
  verbose = FALSE
)
```

Arguments

<code>formula</code>	a model formula, e.g. <code>cases ~ x1 + offset(log(pop))</code>).
<code>data</code>	an <code>sf</code> object of polygons whose columns hold the response, covariates and offset (one row per region, or per region-time for spatio-temporal fits).
<code>time</code>	optional name of a time column in <code>data</code> ; if given, a spatio-temporal model is fitted (data must have one row per region and time).
<code>rasters</code>	optional <code>terra::SpatRaster</code> of spatially continuous covariates (layers named in formula).

covariates	optional named list of sf point layers giving covariates observed on a different support (e.g. monitors); each is kriged to the candidate points and enters with a Berkson correction (see SDALGCP2_misaligned).
popden	optional population-density SpatRaster; if supplied, the region aggregation is population-weighted.
control	a sdalgcp_control list of settings (smart defaults).
verbose	logical; print progress.

Value

a fitted model object of class `c("sdalgcp", ...)` with `print`, `summary`, `confint`, `predict` and `plot` methods.

See Also

[predict.sdalgcp](#), [sdalgcp_control](#), [SDALGCP2](#), [SDALGCP2_raster](#), [SDALGCP2_ST](#)

Examples

```
library(sf)
set.seed(1)
grid <- st_make_grid(st_as_sfc(st_bbox(c(xmin = 0, ymin = 0, xmax = 20, ymax = 20))),
                    n = c(8, 8))
regions <- st_sf(geometry = grid)
regions$x1 <- as.numeric(scale(st_coordinates(st_centroid(regions))[, 1]))
regions$pop <- round(runif(nrow(regions), 500, 3000))
regions$cases <- rpois(nrow(regions), regions$pop * exp(-6 + 0.5 * regions$x1))

fit <- sdalgcp(cases ~ x1 + offset(log(pop)), data = regions) # that's it
summary(fit)
rr <- predict(fit) # an sf you can plot() directly
plot(fit) # default relative-risk map
```

sdalgcp_control

Control settings for sdalgcp

Description

Bundles the technical knobs so that a default fit needs none of them.

Usage

```
sdalgcp_control(
  delta = NULL,
  points_per_region = 16,
  point_method = c("regular", "uniform", "ssi"),
  scale = c("continuous", "grid"),
```

```

phi = NULL,
kappa = 0.5,
kappa_t = 0.5,
nugget = FALSE,
confounding = c("none", "restricted"),
reanchor = 2L,
n_sim = 10000L,
burnin = 2000L,
thin = 8L,
tilt_spatial = FALSE,
nthreads = 0L
)

```

Arguments

delta	candidate-point spacing. If NULL (default) it is chosen automatically to place roughly <code>points_per_region</code> points in a typical region.
points_per_region	target number of candidate points per region used to pick <code>delta</code> automatically.
point_method	how candidate points are laid out: "regular" (deterministic grid, default), "uniform" or "ssi".
scale	how the spatial scale ϕ is estimated: "continuous" (optimised directly, no grid – the default) or "grid" (profiled over ϕ). Spatio-temporal fits always profile ϕ on a grid.
phi	optional ϕ grid (only used when <code>scale = "grid"</code> or for spatio-temporal fits); chosen from the geometry if NULL.
kappa	spatial Matern smoothness (0.5, 1.5 or 2.5).
kappa_t	temporal Matern smoothness (spatio-temporal fits).
nugget	logical; add an unstructured region-level term (overdispersion). Requires <code>scale = "continuous"</code> .
confounding	"none" (default) or "restricted". With "restricted", restricted spatial regression is used: the spatial random effect is constrained to the orthogonal complement of the fixed-effect design so it cannot absorb a spatially structured covariate (avoids spatial confounding / attenuation of beta). Spatial models only.
reanchor	number of re-anchoring passes (re-simulate the latent field at the optimum and refit) for reliable variance estimates. Default 2.
n_sim, burnin, thin	MCMC length controls for the latent-field sampler.
tilt_spatial	logical; for raster covariates, use the fully covariate-tilted correlation (see SDALGCP2_raster).
nthreads	OpenMP threads for the correlation assembly (0 = default).

Value

a list of control settings.

See Also[sdalgcp](#)**Examples**

```
## defaults, then a faster grid-based fit with a nugget term
str(sdalgcp_control())
ctrl <- sdalgcp_control(scale = "grid", nugget = FALSE, n_sim = 4000,
                        burnin = 1000, thin = 6)
```

`sdalgcp_data`*Simulated aggregated disease-count data*

Description

A small, self-contained example dataset used throughout the help pages and vignettes. It is simulated from the model the package fits: an 8x8 lattice of regions, a spatially structured covariate, a latent Gaussian spatial field with exponential covariance, and Poisson counts with a population offset. The true fixed effects are (Intercept) = -6 and $x_1 = 0.6$; the latent field has variance $\sigma^2 = 0.3$ and exponential scale $\phi = 4$.

Usage`sdalgcp_data`**Format**

An `sf` object of 64 POLYGON regions with columns:

region integer region identifier (1-64).

cases observed disease count in the region.

x1 a standardised, spatially structured covariate.

pop population at risk (the offset; use `offset(log(pop))`).

geometry the region polygon.

Source

Simulated; see `data-raw/sdalgcp_data.R` in the package sources.

See Also

[liver](#) for a real disease-count example.

Examples

```
data(sdalgcp_data)
summary(sdalgcp_data$cases)
plot(sdalgcp_data["cases"])
```

SDALGCP2

*Fit a spatial SDA-LGCP model***Description**

End-to-end user entry point: generates candidate points inside each region, assembles the aggregated region-level correlation array (C++), and estimates parameters by Monte Carlo maximum likelihood.

Usage

```
SDALGCP2(
  formula,
  data,
  my_shp,
  delta,
  phi = NULL,
  method = 1L,
  weighted = FALSE,
  pop_shp = NULL,
  kappa = 0.5,
  par0 = NULL,
  control.mcmc = NULL,
  phi_method = c("grid", "direct"),
  nugget = FALSE,
  confounding = c("none", "restricted"),
  reanchor = 0L,
  rho = 0.55,
  giveup = 1000L,
  nthreads = 0L,
  messages = FALSE
)
```

Arguments

formula	model formula, e.g. $\text{cases} \sim x1 + \text{offset}(\log(\text{pop}))$.
data	data frame with the model variables (one row per region).
my_shp	sf polygons (or anything coercible via <code>st_as_sf</code>).
delta	candidate-point spacing.
phi	numeric vector of spatial scale parameters to profile; if NULL, a default grid from $\sqrt{\text{min area}}$ to $\text{extent}/10$.
method	point method: 1 = SSI, 2 = uniform, 3 = regular grid.
weighted	logical; population-weighted aggregation using <code>pop_shp</code> .
pop_shp	population-density <code>SpatRaster</code> (needed if <code>weighted</code>).
kappa	Matern smoothness for the spatial kernel (0.5 default).

par0	optional starting values c(beta, sigma2, phi).
control.mcmc	list from control_mcmc .
phi_method	how the spatial scale is estimated: "grid" (profile over the supplied phi grid, the robust default) or "direct" (optimise phi continuously inside the MCML objective; exponential kernel only). See the package vignette/PDF on the double-integral derivation.
nugget	logical; if TRUE (requires phi_method = "direct") add an unstructured region-level term, fitting covariance $\sigma^2(R(\phi) + \nu I)$ and estimating the relative nugget $\nu = \tau^2/\sigma^2$ with a standard error. Absorbs overdispersion not explained by the spatial structure.
confounding	"none" (default) or "restricted" for restricted spatial regression (constrains the spatial random effect orthogonal to the fixed-effect design; fitted by a Laplace-approximate marginal likelihood).
reanchor	number of re-anchoring passes: after fitting, the latent field is re-simulated at the current optimum and the model refit, which keeps the importance weights near-uniform (raises the MC effective sample size). 0 (default) fits once; 2-3 is usually ample.
rho, giveup	point-generation controls.
nthreads	OpenMP threads for the correlation build.
messages	logical; print optimiser progress.

Value

an object of class "SDALGCP2".

See Also

[mcm1_fit](#), [precompute_corr](#), [sda_points](#)

Examples

```
library(sf)
## ---- simulate a lattice of regions and aggregated counts ----
set.seed(1)
bound <- st_as_sfc(st_bbox(c(xmin = 0, ymin = 0, xmax = 20, ymax = 20)))
shp <- st_sf(geometry = st_make_grid(bound, n = c(8, 8)))
N <- nrow(shp)

pts <- sda_points(shp, delta = 1.2, method = 3) # regular grid points
phi_grid <- seq(1, 5, length.out = 8)
corr <- precompute_corr(pts, phi_grid)
Sig <- 0.5 * corr$R[, , which.min(abs(phi_grid - 2.5))]
x1 <- as.numeric(scale(st_coordinates(st_centroid(shp))[, 1]))
pop <- round(runif(N, 500, 3000))
y <- rpois(N, pop * exp(cbind(1, x1) %*% c(-6, 0.5) +
                           as.numeric(t(chol(Sig)) %*% rnorm(N))))
dat <- data.frame(y = y, x1 = x1, pop = pop)
```

```
## ---- fit ----
ctrl <- control_mcmc(n.sim = 6000, burnin = 1500, thin = 6, h = 1.65 / N^(1/6))
fit <- SDALGCP2(y ~ x1 + offset(log(pop)), dat, shp, delta = 1.2,
               phi = phi_grid, method = 3, control.mcmc = ctrl)
summary(fit)

## ---- predict ----
pred_d <- predict(fit, type = "discrete", sampler = "mcmc", control.mcmc = ctrl)
pred_c <- predict(fit, type = "continuous", sampler = "laplace", cellsize = 1,
                 control.mcmc = ctrl)
```

SDALGCP2_misaligned *Fit an SDA-LGCP with covariates measured on a different support*

Description

Covariates observed on a *different support* from the outcome (e.g. air-quality monitors at point locations) are kriged to the candidate points and enter the model on the intensity scale with a Berkson correction that propagates the prediction uncertainty (see [math/confounding-and-misalignment.pdf](#)).

Usage

```
SDALGCP2_misaligned(
  formula,
  data,
  delta,
  covariates,
  phi = NULL,
  method = 3L,
  weighted = FALSE,
  pop_shp = NULL,
  berkson = TRUE,
  control.mcmc = NULL,
  max_iter = 10L,
  tol = 0.001,
  messages = FALSE
)
```

Arguments

formula	model formula; the covariate names appear on the right-hand side.
data	sf polygons holding the response and offset (one row/region).
delta	candidate-point spacing.
covariates	a named list; each element is an sf carrying a column of the same name – the covariate’s observed values on its own support, either points (e.g. monitors; plain kriging) or polygons (areal averages on a different partition; aggregated areal kriging).

phi spatial-scale grid for the outcome model (default from geometry).
 method, weighted, pop_shp point-generation controls.
 berkson logical; include the Berkson uncertainty correction (default TRUE). FALSE gives the naive kriged-mean plug-in.
 control.mcmc list from [control_mcmc](#).
 max_iter, tol outer Gauss-Newton controls.
 messages logical; print progress.

Value

an object of class "SDALGCP2" with `misaligned = TRUE`.

See Also

[SDALGCP2_raster](#), [sda1gcp](#)

Examples

```

data(sda1gcp_data)
set.seed(1)
## a covariate z observed only at 40 scattered monitor points (a different support)
mon <- sf::st_as_sf(data.frame(x = runif(40, 0, 20), y = runif(40, 0, 20)),
  coords = c("x", "y"))
mon$z <- scale(sf::st_coordinates(mon)[, 1])[, 1]
fit <- SDALGCP2_misaligned(cases ~ z + offset(log(pop)), sda1gcp_data, delta = 1.5,
  covariates = list(z = mon),
  control.mcmc = control_mcmc(n.sim = 2000, burnin = 500, thin = 5))
summary(fit)

```

SDALGCP2_raster

Fit an SDA-LGCP with spatially continuous (raster) covariates

Description

Covariates supplied as rasters enter the model at the candidate-point level and are aggregated on the intensity (exp) scale via a log-sum-exp offset $b_i(\beta) = \log \sum_k w_{ik} \exp(z(x_{ik})^\top \beta)$ – the statistically correct alternative to averaging the predictor over each polygon. Estimation is a Gauss-Newton fixed point that reuses `mcml_fit` with the intensity-tilted effective design.

Usage

```
SDALGCP2_raster(
  formula,
  data,
  my_shp,
  delta,
  rasters,
  phi = NULL,
  method = 3L,
  weighted = FALSE,
  pop_shp = NULL,
  kappa = 0.5,
  tilt_spatial = FALSE,
  control.mcmc = NULL,
  max_iter = 10L,
  tol = 0.001,
  messages = FALSE
)
```

Arguments

formula	model formula; right-hand-side names must match raster layer names. The response and an offset(log(pop)) come from data.
data	data frame with the response and offset (one row per region).
my_shp	sf polygons.
delta	candidate-point spacing.
rasters	a terra::SpatRaster (or object coercible by terra::rast) whose layers are the spatially varying covariates named in formula.
phi	spatial-scale grid (default chosen from the geometry).
method, weighted, pop_shp	point-generation controls (see sda_points).
kappa	Matern smoothness for the spatial correlation.
tilt_spatial	logical; if FALSE (default) the spatial correlation uses the population weights and is precomputed once (covariates enter only through the log-sum-exp offset). If TRUE, the correlation $R^c(\beta)$ is rebuilt each iteration from the intensity-tilted weights $c_{ik}(\beta)$ and a log-normal aggregation correction $\frac{1}{2}\sigma^2(1 - R_{ii}^c)$ is added – the fully tilted model (more accurate, more costly).
control.mcmc	list from control_mcmc .
max_iter, tol	outer Gauss-Newton iteration controls.
messages	logical; print progress.

Value

an object of class "SDALGCP2" (as [mcml_fit](#)) with extra fields raster = TRUE and n_iter.

See Also[SDALGCP2, mcm1_fit](#)**Examples**

```

data(sdalgcp_data)
## a spatially continuous covariate supplied as a raster layer named "z"
r <- terra::rast(terra::ext(0, 20, 0, 20), resolution = 0.5)
terra::values(r) <- as.numeric(scale(terra::crds(r)[, 1])) # west-east gradient
names(r) <- "z"
df <- sf::st_drop_geometry(sdalgcp_data)
fit <- SDALGCP2_raster(cases ~ z + offset(log(pop)), df, sdalgcp_data,
                      delta = 1.5, rasters = r,
                      control.mcmc = control_mcmc(n.sim = 2000, burnin = 500, thin = 5))
summary(fit)

```

SDALGCP2_ST

*Fit a spatio-temporal SDA-LGCP model (Kronecker-free)***Description**

Separable space-time SDA-LGCP for aggregated counts observed over the same N regions at T times. The spatial scale ϕ is profiled on a grid; the temporal Matern range ν is estimated continuously. The likelihood never forms the $(NT) \times (NT)$ covariance.

Usage

```

SDALGCP2_ST(
  formula,
  data,
  my_shp,
  times,
  delta,
  phi = NULL,
  kappa = 0.5,
  kappa_t = 0.5,
  method = 3L,
  weighted = FALSE,
  pop_shp = NULL,
  control.mcmc = NULL,
  reanchor = 0L,
  rasters = NULL,
  covariates = NULL,
  confounding = c("none", "restricted"),
  berkson = TRUE,
  max_iter = 10L,
  tol = 0.001,

```

```

    messages = FALSE
  )

```

Arguments

formula	model formula (with optional <code>offset(log(pop))</code>).
data	data frame of $N \times T$ rows ordered by time then region (rows $(t-1) \times N + 1 : N$ are time t).
my_shp	sf polygons for the N regions.
times	numeric vector of length T of observation times.
delta	candidate-point spacing.
phi	spatial-scale grid (default from geometry).
kappa	spatial Matern smoothness.
kappa_t	temporal Matern smoothness.
method, weighted, pop_shp	point-generation controls.
control.mcmc	list from control_mcmc .
reanchor	number of re-anchoring passes (re-simulate the latent field at the current optimum and refit); improves the variance-parameter estimates.
rasters	optional terra: :SpatRaster of spatially continuous, time-invariant covariates (layers named in formula); they enter on the intensity scale as in SDALGCP2_raster , fitted by a Gauss-Newton tilting loop around the space-time likelihood.
covariates	optional named list of sf covariate layers measured on a different (time-invariant) support; each is kriged to the candidate points with a Berkson correction as in SDALGCP2_misaligned .
confounding	"none" (default) or "restricted" for restricted spatial regression against space-time confounding (see Details).
berkson	logical; include the Berkson uncertainty correction for covariates (default TRUE).
max_iter, tol	Gauss-Newton controls for the rasters/covariates tilting loop.
messages	logical; print progress.

Details

With `rasters` or `covariates` the covariate surface is taken to be constant over time (time-varying covariates can still be supplied as ordinary columns of data). `confounding = "restricted"` constrains the space-time random effect to the orthogonal complement of the fixed-effect design and is fitted by an analytic Laplace-marginal likelihood; it reduces to the spatial restricted fit when $T = 1$ and is not currently combined with `rasters/covariates`.

Value

an object of class `c("SDALGCP2_ST", "SDALGCP2")` with `phi_opt`, `nu_opt`, coefficient table and covariance.

See Also

[sdalgcp](#) (friendly wrapper), [predict.SDALGCP2_ST](#)

Examples

```

data(sdalgcp_data)
shp <- sdalgcp_data
## build a 3-time panel (data frame, N*T rows ordered by time then region)
times <- 1:3
dat <- do.call(rbind, lapply(times, function(t) {
  d <- sf::st_drop_geometry(shp); d$time <- t
  d$cases <- rpois(nrow(d), d$pop * exp(-6 + 0.6 * d$x1 + 0.1 * (t - 2)))
  d
}))
fit <- SDALGCP2_ST(cases ~ x1 + offset(log(pop)), dat, shp, times = times,
                  delta = 1.5,
                  control.mcmc = control_mcmc(n.sim = 2000, burnin = 500, thin = 5))
fit$phi_opt; fit$nu_opt

## restricted spatial regression against space-time confounding
fit_c <- SDALGCP2_ST(cases ~ x1 + offset(log(pop)), dat, shp, times = times,
                    delta = 1.5, phi = c(2, 4, 6), confounding = "restricted")
fit_c$beta_opt

## a spatially continuous (raster) covariate, aggregated on the intensity scale
r <- terra::rast(terra::ext(0, 20, 0, 20), resolution = 0.5)
terra::values(r) <- as.numeric(scale(terra::crds(r)[, 1])); names(r) <- "z"
fit_r <- SDALGCP2_ST(cases ~ z + offset(log(pop)), dat, shp, times = times,
                    delta = 1.5, phi = c(2, 4, 6), rasters = r, max_iter = 4,
                    control.mcmc = control_mcmc(n.sim = 2000, burnin = 500, thin = 5))
fit_r$beta_opt

```

summary.SDALGCP2

Summary of an SDALGCP2 fit

Description

Summary of an SDALGCP2 fit

Usage

```
## S3 method for class 'SDALGCP2'
summary(object, ...)
```

Arguments

object	an object of class "SDALGCP2" from <code>mcml_fit</code> .
...	unused.

Value

an object of class "summary.SDALGCP2" with a coefficient table.

Index

- * **datasets**
 - liver, [7](#)
 - sda1gcp_data, [27](#)
- coef_plot, [3](#)
- confint.SDALGCP2, [3](#)
- control_mcmc, [4](#), [6](#), [10](#), [29](#), [31](#), [32](#), [34](#)
- exceedance, [5](#), [8](#), [18](#), [19](#)
- glm, [24](#)
- laplace_sampling, [5](#), [6](#)
- liver, [7](#), [27](#)
- map_exceedance, [5](#), [8](#), [19](#)
- mc_diagnostics, [9](#)
- mcml_fit, [9](#), [17](#), [19](#), [29](#), [31–33](#), [35](#)
- model_check, [11](#)
- phi_profile, [12](#), [14](#)
- plot.sda1gcp, [12](#)
- plot.SDALGCP2, [14](#)
- plot.SDALGCP2_pred, [14](#), [19](#)
- plot.SDALGCP2_ST_pred, [15](#), [20](#)
- precompute_corr, [10](#), [17](#), [23](#), [24](#), [29](#)
- predict.sda1gcp, [18](#), [25](#)
- predict.SDALGCP2, [5](#), [14](#), [19](#), [20](#), [22](#)
- predict.SDALGCP2_ST, [18](#), [20](#), [35](#)
- print.SDALGCP2, [21](#)
- print.summary.SDALGCP2, [22](#)
- report, [22](#)
- sda_points, [17](#), [23](#), [29](#), [32](#)
- sda1gcp, [24](#), [25](#), [27](#), [31](#), [35](#)
- SDALGCP2, [10](#), [19](#), [25](#), [28](#), [33](#)
- SDALGCP2_misaligned, [25](#), [30](#), [34](#)
- SDALGCP2_raster, [24–26](#), [31](#), [31](#), [34](#)
- SDALGCP2_ST, [25](#), [33](#)
- sda1gcp_control, [25](#), [25](#)
- sda1gcp_data, [7](#), [27](#)
- sf, [7](#), [20](#), [27](#)
- summary.SDALGCP2, [35](#)