Package: mrtsSphere (via r-universe)

October 7, 2026

Type Package

Title Multi-Resolution Thin-Plate Splines on the Sphere

Version 0.2.0

Description Constructs multi-resolution thin-plate spline (MRTS) basis functions on the sphere and fits the corresponding multi-resolution mixed model (MRMM) by generalised least squares with a tapered Matérn residual covariance, including kriging-style prediction with closed-form standard errors. Implements the system described in Huang, Huang, and Ing (2025) ``Multi-Resolution Spatial Methods on the Sphere: Efficient Prediction for Global Data'', Environmetrics, <doi:10.1002/env.70092>. Heavy basis computations are written in 'C++' via 'Rcpp' with optional 'OpenMP' parallelism.

License GPL (>= 2)

Encoding UTF-8

Depends R (>= 4.0)

Imports Rcpp, RSpectra

LinkingTo Rcpp, RcppEigen, RcppNumerical

Suggests fields, testthat (>= 3.0.0)

SystemRequirements GNU make

RoxygenNote 7.3.2

Roxygen list(markdown = TRUE)

URL https://github.com/STLABTW/multi-resolution-sphere

BugReports https://github.com/STLABTW/multi-resolution-sphere/issues

Config/testthat/edition 3

Config/pak/sysreqs make

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

Date/Publication 2026-05-01 08:24:18 UTC

RemoteUrl https://github.com/stlabtw/multi-resolution-sphere

RemoteRef HEAD

RemoteSha 206bccb5d8ddaf0f5cd56b31c93cf31121e18d11

Contents

mrmm_fit 2
mrts_sphere 4
predict.mrmm 5
tapered_matern_sphere 7

Index **9**

`mrmm_fit` *Fit a multi-resolution mixed model (MRMM) on the sphere*

Description

Combines the `mrts_sphere()` basis with a tapered Matérn residual covariance and obtains the basis coefficients by generalised least squares (GLS).

Usage

```
mrmm_fit(
  z,
  loc,
  knots,
  k,
  taper_range,
  matern_range,
  sill,
  nugget,
  smoothness = 0.5
)
```

Arguments

`z`	Numeric vector of observations at the training locations.
`loc`	Two-column numeric matrix of `(latitude, longitude)` in degrees giving the training locations. Must have `length(z)` rows.
`knots`	Two-column numeric matrix of `(latitude, longitude)` in degrees giving the knot locations of the MRTS basis.
`k`	Integer. Number of MRTS basis functions to use (`2 <= k <= nrow(knots)`).
`taper_range`	Wendland compact-support range a (radians on the unit sphere).
`matern_range`	Matérn range parameter c.
`sill`	Sill of the structured residual component (σ_y^2).
`nugget`	Nugget variance (σ_ε^2); diagonal added to the covariance.
`smoothness`	Matérn smoothness ν. Default `0.5`.

Details

Given training observations z at locations $s_1, \ldots, s_n$ on the sphere, the model is

$$z(s) = B(s)\,\beta + \varepsilon(s), \qquad \mathrm{Cov}(\varepsilon(s_i), \varepsilon(s_j)) = \Sigma_{ij},$$

where $B(s)$ is the MRTS basis at location s and Σ is the tapered-Matérn covariance built by `tapered_matern_sphere()` using the user-supplied parameters. The GLS estimate

$$\hat{\beta} = (B^\top \Sigma^{-1} B)^{-1} B^\top \Sigma^{-1} z$$

is computed via a Cholesky factorisation of Σ. Predictions (mean and standard error) at new locations are obtained with `predict.mrmm()`.

Covariance parameters must be supplied. A separate parameter-estimation routine (`mrmm_estimate_params()`) is planned for a future release.

Value

An object of class `"mrmm"`: a list with components

`coefficients` GLS estimate $\hat{\beta}$, length `k`.

`fitted`, `residuals` Fitted values and residuals at the training locations.

`coef_cov` The matrix $(B^\top \Sigma^{-1} B)^{-1}$, used by `predict.mrmm()` for standard errors.

`sigma_chol` Upper-triangular Cholesky factor of Σ at the training locations.

`basis_train`, `train_loc`, `train_z` Stored training basis, locations, and observations.

`knots`, `k`, `params`, `call` Inputs preserved on the fit.

References

Huang, H.-Y., Huang, H.-C., and Ing, C.-K. (2025). Multi-Resolution Spatial Methods on the Sphere: Efficient Prediction for Global Data. *Environmetrics*. doi:10.1002/env.70092

See Also

`predict.mrmm()`, `mrts_sphere()`, `tapered_matern_sphere()`.

Examples

```
set.seed(1)
n <- 80
loc <- cbind(runif(n, -60, 60), runif(n, -180, 180))
z   <- 2 + sin(loc[, 1] * pi / 180) + rnorm(n, sd = 0.3)

knots <- loc[sample.int(n, 30), ]
fit <- mrmm_fit(z = z, loc = loc, knots = knots, k = 8,
                taper_range  = 1.0,
                matern_range = 0.2,
                sill         = 0.5,
                nugget       = 0.1,
                smoothness   = 0.5)
fit
length(fit$coefficients)   # k
```

`mrts_sphere` *Multi-resolution thin-plate spline basis on the sphere*

Description

Builds a set of `k` multi-resolution thin-plate spline (MRTS) basis functions on the sphere from a set of knot locations, and evaluates them at the prediction locations `X`.

Usage

```
mrts_sphere(knot, k, X)
```

Arguments

`knot`	Numeric matrix with two columns giving knot locations as `(latitude, longitude)` in degrees.
`k`	Integer. Number of basis functions to construct (the rank of the basis). Must satisfy `2 <= k <= nrow(knot)`.
`X`	Numeric matrix with two columns giving prediction locations as `(latitude, longitude)` in degrees, where the basis is evaluated.

Details

The first basis function is constant (`sqrt(1/n)`); the remaining `k - 1` basis functions are obtained from the eigen-decomposition of the centered knot kernel matrix, following the construction described in the reference.

Value

A list with one element:

`mrts` An `nrow(X) x k` numeric matrix whose columns are the basis functions evaluated at the rows of `X`.

References

Multi-resolution approximations of Gaussian processes for large spatial datasets on the sphere. *Environmetrics*, 2025. doi:10.1002/env.70092

Examples

```
## Build a small global grid in (lat, lon) degrees.
n_lon <- 12
n_lat <- 8
lon_seq <- seq(-180, 150, length.out = n_lon)
lat_seq <- seq( -80,  80, length.out = n_lat)
grid <- as.matrix(expand.grid(lat = lat_seq, lon = lon_seq))
```

```
## Pick 30 knots and evaluate the MRTS basis at every grid point.
set.seed(1)
knots <- grid[sample(nrow(grid), 30), ]
res <- mrts_sphere(knots, k = 5, X = grid)
dim(res$mrts)   # nrow(grid) x k

## Recovering a simulated spherical exponential field with the basis.
if (requireNamespace("fields", quietly = TRUE)) {
  # Great-circle distance (km) -> exponential covariance.
  d_grid    <- fields::rdist.earth(grid[, 2:1], miles = FALSE)
  cov_field <- exp(-d_grid / 2000)
  y_true    <- as.numeric(t(chol(cov_field + diag(1e-8, nrow(grid)))) %*%
                          rnorm(nrow(grid)))

  # Noisy observations at the knot locations.
  obs_idx <- match(data.frame(t(knots)), data.frame(t(grid)))
  z_obs   <- y_true[obs_idx] + rnorm(nrow(knots), sd = 0.3)

  # Project into the MRTS basis (least squares) and predict on the grid.
  B_obs    <- res$mrts[obs_idx, , drop = FALSE]
  beta_hat <- qr.solve(B_obs, z_obs)
  y_hat    <- res$mrts %*% beta_hat
  sqrt(mean((y_hat - y_true)^2))   # RMSE
}
```

`predict.mrmm` *Predict from a multi-resolution mixed model fit*

Description

Computes kriging-style predictions and (optionally) prediction standard errors at new spherical locations from an `mrmm_fit()` object.

Usage

```
## S3 method for class 'mrmm'
predict(object, newdata, se.fit = TRUE, ...)
```

Arguments

`object`	An object of class `"mrmm"` from `mrmm_fit()`.
`newdata`	Two-column numeric matrix of (`latitude`, `longitude`) in degrees giving the prediction locations.
`se.fit`	Logical; if `TRUE` (default), also return the prediction standard error at each new location.
`...`	Currently unused.

Details

Given the GLS coefficient $\hat{\beta}$ stored in `object`, the prediction at a new location s^* is

$$\hat{z}(s^*) = B(s^*)\hat{\beta} + \Sigma(s^*, S_{\text{train}})\,\Sigma_{\text{train}}^{-1}\,(z - B\hat{\beta}).$$

The prediction variance is the closed-form expression that arises in kriging with linear-trend basis (sometimes called "universal kriging"); see the reference.

Value

A list with

`fit` Numeric vector of predicted values, one per row of `newdata`.

`se.fit` Numeric vector of prediction standard errors (only if `se.fit = TRUE`).

References

Huang, H.-Y., Huang, H.-C., and Ing, C.-K. (2025). Multi-Resolution Spatial Methods on the Sphere: Efficient Prediction for Global Data. *Environmetrics*. doi:10.1002/env.70092

See Also

`mrmm_fit()`.

Examples

```
set.seed(1)
n <- 80
loc <- cbind(runif(n, -60, 60), runif(n, -180, 180))
z   <- 2 + sin(loc[, 1] * pi / 180) + rnorm(n, sd = 0.3)
knots <- loc[sample.int(n, 30), ]
fit <- mrmm_fit(z = z, loc = loc, knots = knots, k = 8,
                taper_range = 1.0, matern_range = 0.2,
                sill = 0.5, nugget = 0.1)

new_loc <- cbind(seq(-50, 50, length.out = 5),
                 seq(-100, 100, length.out = 5))
p <- predict(fit, newdata = new_loc, se.fit = TRUE)
p$fit
p$se.fit
```

`tapered_matern_sphere` *Tapered Matérn covariance on the sphere*

Description

Builds a covariance matrix of the form

Usage

```
tapered_matern_sphere(
  loc1,
  loc2 = NULL,
  taper_range,
  matern_range,
  smoothness = 0.5,
  sill = 1,
  nugget = 0
)
```

Arguments

`loc1`	Two-column numeric matrix of `(latitude, longitude)` locations in degrees.
`loc2`	Optional second set of locations. If `NULL`, the covariance is computed within `loc1` (symmetric output) and the nugget is added on the diagonal.
`taper_range`	Compact-support range a of the Wendland taper, in radians on the unit sphere.
`matern_range`	Range parameter c of the Matérn correlation.
`smoothness`	Smoothness ν of the Matérn correlation. Default `0.5` (exponential).
`sill`	Marginal variance of the structured component. Must be non-negative.
`nugget`	Nugget variance, added on the diagonal when `loc2 = NULL`. Must be non-negative.

Details

$$\text{sill} \cdot \rho_{\text{Mat}}(d;\, c, \nu) \cdot W(d;\, a) \;+\; \text{nugget} \cdot I$$

where d is the great-circle distance on the unit sphere (radians), ρ_{Mat} is the Matérn correlation with range c and smoothness ν, and W is the Wendland C^2 taper of compact support a (degree $k = 2$, dimension $= 2$). The nugget term is added only when the matrix is symmetric, i.e. when `loc2 = NULL`.

Value

An `nrow(loc1) x nrow(loc2)` numeric covariance matrix.

Examples

```
set.seed(1)
loc <- cbind(runif(20, -60, 60), runif(20, -180, 180))
Sigma <- tapered_matern_sphere(loc,
    taper_range = 0.4,
    matern_range = 0.1,
    smoothness = 0.5,
    sill = 1.0,
    nugget = 0.05)
isSymmetric(Sigma)
```

Index

mrmm_fit, 2
mrmm_fit(), *5*, *6*
mrts_sphere, 4
mrts_sphere(), *2*, *3*

predict.mrmm, 5
predict.mrmm(), *3*

tapered_matern_sphere, 7
tapered_matern_sphere(), *3*