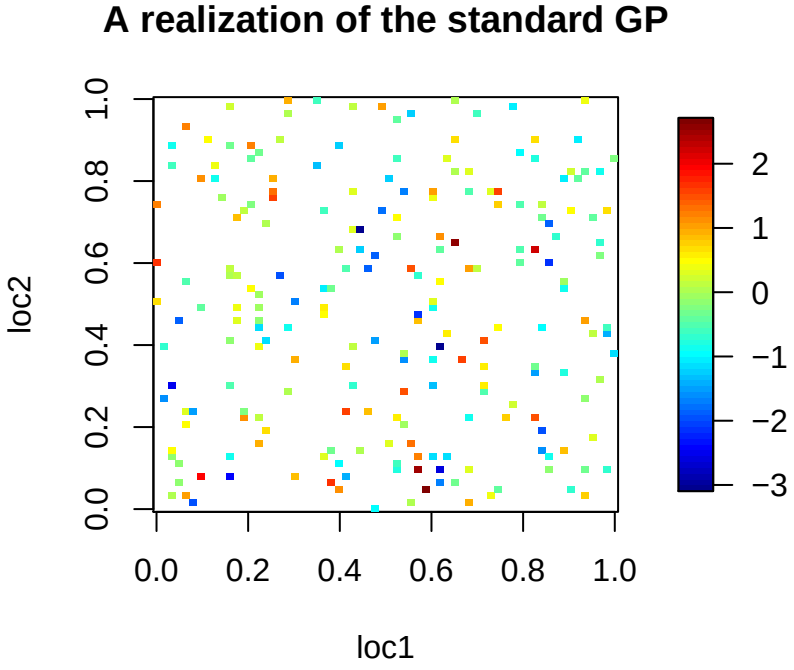
Basic use of GPsim

This vignette is aimed at breifly introducing how to use functions in GPsim. This package has five main functions: simulate_gp, corrvecchia_knownCovparms, cov_expo_iso, cov_expo_aniso, and kldiv.

generating a realization of a Gaussian Process (GP)

In this section, we generate a realization of a gaussian process. Its mean function is a zero function and its covariance function is an anisotropic exponential covariance function with variance of 1, range of 0.1, and degree of anisotropy of 10. This output can be visualized using the quilt.plot function in the package fields.

```
library(GPsim)

set.seed(2019)

covparms <- c(1, 0.1, 10)  # (variance, range, degree of anisotropy)

realization <- simulate_gp(locs = "random", n = 15^2, p = 2, meanmodel = function(locs,
    meanparms) rep(0, nrow(locs)), meanparms = NULL, covmodel = cov_expo_aniso,
    covparms = covparms, pivot = FALSE, correction = NULL, tol = .Machine$double.eps,
    seed = 2020)
#> The decomposition error of the covariance matrix is 1.98210542803315e-15 in the Frobenius norm.

fields::quilt.plot(realization$locs[, 1], realization$locs[, 2], realization$y,
    main = "A realization of the standard GP", xlab = "loc1", ylab = "loc2")
```



Applying the Vecchia approximation

In this section, we apply two differnt vecchia approaches: one is the Vecchia approach with Euclidean distance (standard Vecchia apporximation) and the other is the Vecchia approximation with correlation-based distance $1 - \rho$ (correlation-based Vecchia approximation).

```
vecchia.fit.euclidean <- corrvecchia_knownCovparms(locs = realization$locs,
    m = 10, ordering = "maxmin", coordinate = NULL, dist.ordering = "euclidean",
    dist.conditioning = "euclidean", covmodel = realization$covmat.correted,
    covparms = c(1))

vecchia.fit.correlation <- corrvecchia_knownCovparms(locs = realization$locs,
    m = 10, ordering = "maxmin", coordinate = NULL, dist.ordering = "correlation",
    dist.conditioning = "correlation", covmodel = realization$covmat.correted,
    covparms = c(1))
```

Comparing performance of euclidean and correlation-based Vecchia approaches

In this section, we compare the above two Vecchia approaches. To evaluate them, the GPvecchia package must be used. But unfortunately, this package is unavailable online now because of several technical issues. I will update this vignette after the issues are fixed.

As you can see below, we can estimate the covariance matrix of the GP using each Vecchia approach. For the standard Vecchia approximation, Kullback Leiber (KL) divergence between true GP and approximate GP is 0.404. On the other hand, for the correlation-based Vecchia approximation, that is only 0.000556. In sum, it is clear that the correlation-based Vecchia approximation performs better than the stadard one in cases of anisotropic covariance functions.

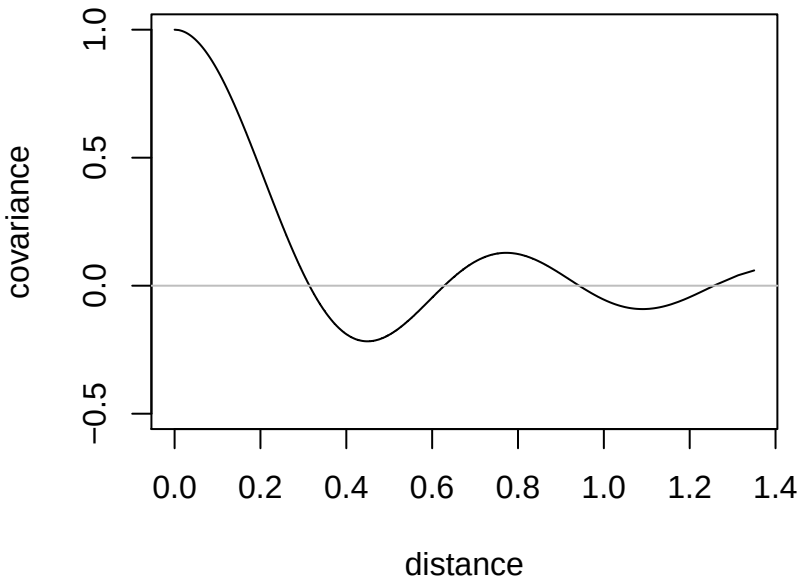
```
library(GPvecchia)

sigma.ord <- cov_expo_aniso(locs = vecchia.fit.euclidean$locsord, covparms = covparms)
U <- createU(vecchia.fit.euclidean, covparms = covparms, nugget = 0, covmodel = sigma.ord)
revord <- order(vecchia.fit.euclidean$ord)
sigma.hat <- as.matrix(solve(Matrix::tcrossprod(U$U)))[revord, revord]
kls.euclidean <- kldiv(realization$covmat.correted, sigma.hat)

sigma.ord <- cov_expo_aniso(locs = vecchia.fit.correlation$locsord, covparms = covparms)
U <- createU(vecchia.fit.correlation, covparms = covparms, nugget = 0, covmodel = sigma.ord)
revord <- order(vecchia.fit.correlation$ord)
sigma.hat <- as.matrix(solve(Matrix::tcrossprod(U$U)))[revord, revord]
kls.correlation <- kldiv(realization$covmat.correted, sigma.hat)

c(kls.euclidean, kls.correlation)
```

generating a realization of a GP with user-specific mean and covariance functions

Suppose one hope to generate a realization of a GP with his/her own mean and covariance functions. Let us assume that the mean function is the zero function and the covariance function is the wave covariance function. Then, to begin with, one should declare those two functions. The wave covariance function can be plotted as below:

```
mean_zero <- function(locs, meanparms) rep(0, nrow(locs))

# covparms = c(sigma, period)
```

```r
cov_wave <- function(locs, covparms) {

    h <- fields::rdist(locs)

    ind <- which(h < 1e-08, arr.ind = T)
    c <- covparms[1] * sin(h/covparms[2]) * (covparms[2]/h)
    c[ind] <- covparms[1]

    return(c)
}

n <- 10^2
locs <- matrix(runif(n * 2, 0, 1), n, 2)
h <- fields::rdist(locs)
ind <- order(unlist(h))
hord <- unlist(h)[ind]

covparms <- c(1, 0.1)
covh <- cov_wave(locs = locs, covparms = covparms)
covhord <- unlist(covh)[ind]
plot(hord, covhord, type = "l", ylim = c(-0.5, 1), col = 1, xlab = "distance",
    ylab = "covariance")
abline(h = 0, col = "gray")
```



Then, we can generate a realization of the gaussian process with the above mean and covariance functions only using the simulate_gp() function. The realization can be plotted as below:

```r
realization <- simulate_gp(locs = "random", n = 15^2, p = 2, meanmodel = mean_zero,
    meanparms = NULL, covmodel = cov_wave, covparms = covparms, pivot = FALSE,
```

```
    correction = "GMW81", tol = sqrt(.Machine$double.eps), seed = 2020)
#> The decomposition error of the covariance matrix is 1.71408783850666e-07 in the Frobenius norm.

fields::quilt.plot(realization$locs[, 1], realization$locs[, 2], realization$y,
    main = "A realization of the standard GP", xlab = "loc1", ylab = "loc2")
```