SpatialGEV: Fast Bayesian inference for spatial extreme value models in R

April 22, 2024

Summary

Extreme weather phenomena such as floods and hurricanes are of great concern due to their potential to cause extensive damage. To develop more reliable damage prevention protocols, statistical models are often used to infer the chance of observing an extreme weather event at a given location (Coles and Casson 1998; Cooley, Nychka, and Naveau 2007; Sang and Gelfand 2010). Here we present `SpatialGEV`, an R package providing a fast and convenient toolset for analyzing spatial extreme values using a hierarchical Bayesian modeling framework. In this framework, the marginal behavior of the extremes is given by a generalized extreme value (GEV) distribution, whereas the spatial dependence between locations is captured by modeling the GEV parameters as spatially varying random effects following a Gaussian process (GP). Users are provided with a streamlined way to build and fit various GEV-GP models in R, which are compiled in C++ under the hood. For downstream analyses, the package offers methods for Bayesian parameter estimation and forecasting of extreme events.

Statement of need

The GEV-GP model has important applications in meteorological studies. For example, let $y = y(\boldsymbol{x})$ denote the amount of rainfall at a spatial location $\boldsymbol{x}$. To forecast extreme rainfalls, it is often of interest for meteorologists to estimate the $1/p$-year rainfall return value $z_p(\boldsymbol{x})$, which is the value above which precipitation levels at location $\boldsymbol{x}$ occur with probability p, i.e.,

$$\Pr\big(y(\boldsymbol{x}) > z_p(\boldsymbol{x})\big) = 1 - F_{y\mid\boldsymbol{x}}\big(z_p(\boldsymbol{x})\big) = p, \qquad (1)$$

where the CDF is that of the GEV distribution specific to location $\boldsymbol{x}$. The value $r = 1/p$ is known as the return period. When p is chosen to be a small value, $z_p(\boldsymbol{x})$ indicates how extreme the precipitation level might be at location $\boldsymbol{x}$.

In a Bayesian context, the posterior distribution $p(z_p(\boldsymbol{x}) \mid \boldsymbol{y})$, where $\boldsymbol{y} = (\boldsymbol{y}(\boldsymbol{x}_1), \ldots, \boldsymbol{y}(\boldsymbol{x}_n))$ represents rainfall measurements at n different locations, is very useful for forecasting extreme weather events. Traditionally, Markov Chain Monte Carlo (MCMC) methods are used to sample from the posterior distribution of the GEV model (e.g., Cooley, Nychka, and Naveau 2007; Schliep et al. 2010; Dyrrdal et al. 2015). However, this can be extremely computationally intensive when the number of locations is large. The `SpatialGEV` package implements Bayesian inference based on the Laplace approximation as an alternative to MCMC, making large-scale spatial analyses orders of magnitude faster while achieving roughly the same accuracy as MCMC. The Laplace approximation is carried out using the R/C++ package `TMB` (Kristensen et al. 2016). Details of the inference method can be found in Chen, Ramezan, and Lysy (2022).

Statement of field

The R package `SpatialExtremes` (Ribatet, Singleton, and R Core team 2022) is one of the most popular software for fitting spatial extreme value models, which employs an efficient Gibbs sampler. The Stan programming language and its R interface `RStan` (Stan Development Team 2020) provides off-the-shelf implementations for Hamiltonian Monte Carlo and its variants (Neal 2011; Hoffman and Gelman 2014), which are considered state-of-the-art MCMC algorithms and often used for fitting hierarchical spatial models. Chen, Ramezan, and Lysy (2022) compares the speed and accuracy of the Laplace method implemented in `SpatialGEV` to `RStan`. It is found that `SpatialGEV` is three orders of magnitude faster than `RStan`. A well-known alternative to MCMC is the `R-INLA` package (Lindgren and Rue 2015) which implements the integrated nested Laplace approximation (INLA) approach. As an extension of the Laplace approximation, INLA is often considerably more accurate. However, the `R-INLA` implementation is inapplicable to GEV-GP models in which two or more GEV parameter are modeled as random effects following different Gaussian processes. In contrast, `SpatialGEV` offers more flexibility as it is straightforward for the user to choose what GEV parameters are spatial random effects. Wood (2023) and Youngman (2022) provide another means for estimating spatially varying GEV parameters in the framework of generalized additive models (GAMs), as opposed to Bayesian hierarchical models in `SpatialGEV`. Though the latent spatial effects are modelled differently, connections between the GAM approach and the GEV-GP model with a Matérn-SPDE kernel are described in Miller, Glennie, and Seaton (2020).

Example

Model fitting

The main functions of the `SpatialGEV` package are `spatialGEV_fit()`, `spatialGEV_sample()`, and `spatialGEV_predict()`. This example shows how to apply these functions to analyze a simulated dataset using the GEV-GP model. The spatial domain is a 20×20 regular lattice on $[0, 10] \times [0, 10] \subset \mathbb{R}^2$, such that there are $n = 400$ locations in total. The GEV location parameter $a(\boldsymbol{x})$ and the scale parameter $b(\boldsymbol{x})$ are generated from surfaces depicted in Figure 1, whereas the shape parameter s is a constant $exp(-2)$ across space. 10 to 30 observations per location are simulated from the GEV distribution conditional on the GEV parameters $(a(\boldsymbol{x}), b(\boldsymbol{x}), s)$. The simulated data is provided by the package as a list called `simulatedData`.

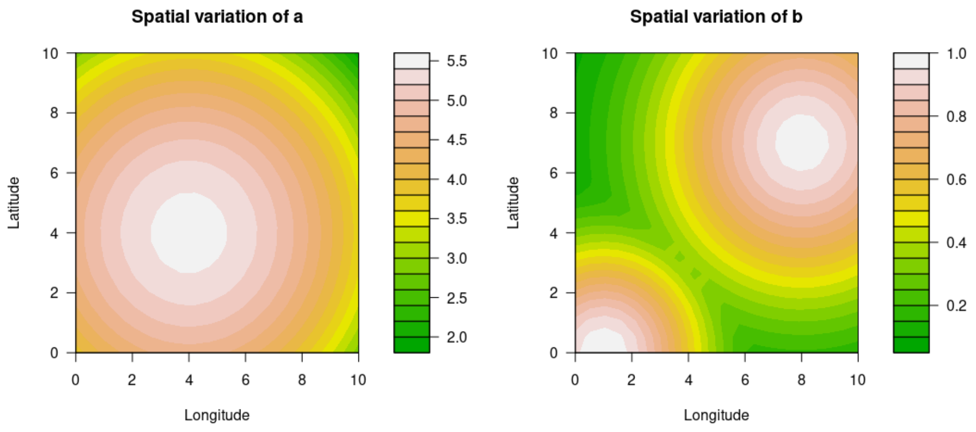

Figure 1: The simulated GEV location parameters $a(\boldsymbol{x}_i)$ and scale parameters $(b(\boldsymbol{x}_i))$ plotted on regular lattices.

The GEV-GP model is fitted by calling `spatialGEV_fit()`. By specifying `random="ab"`, only the GEV parameters a and b are considered spatial random effects. Initial parameter values are passed to `init_param`, where `log_sigma_{a/b}` and `log_ell_{a/b}` are hyperparameters in the GP exponential kernel functions, and `beta_{a/b}` are regression coefficients in the GP mean function. The GP kernel function is chosen using `kernel="exp"`. Other kernel function options are the Matérn kernel and the SPDE approximation to the Matérn described in Lindgren, Rue, and Lindström (2011). The argument `reparam_s="positive"` means we constrain the shape parameter to be positive, i.e., its estimation is done on the log scale. Covariates to include in the mean functions can be provided in a matrix form to `X_{a/b}`. In this example, we only include the intercepts. The posterior mean estimates of the spatial random effects can be accessed from `mod_fit$report$par.random`, whereas the fixed effects can be obtained from `mod_fit$report$par.fixed`.

```r
set.seed(123)                          # set seed for reproducible results
library(SpatialGEV)                    # load package
n_loc <- 50                            # number of locations
locs <- simulatedData$locs[1:n_loc,]   # location coordinates
a <- simulatedData$a[1:n_loc]          # true GEV location parameters
logb <- simulatedData$logb[1:n_loc]    # true GEV (log) scale parameters
logs <- simulatedData$logs             # true GEV (log) shape parameter
y <- simulatedData$y[1:n_loc]          # simulated observations
# Model fitting
fit <- spatialGEV_fit(data = y, locs = locs, random = "ab",
                      init_param = list(a = rep(4, n_loc),
                                        log_b = rep(0,n_loc),
                                        s = -2,
                                        beta_a = 4, beta_b = 0,
                                        log_sigma_a = 0, log_ell_a = 1,
                                        log_sigma_b = 0, log_ell_b = 1),
                      reparam_s = "positive", kernel="exp",
                      X_a = matrix(1, nrow=n_loc, ncol=1),
                      X_b = matrix(1, nrow=n_loc, ncol=1),
                      silent=T)
print(fit)
#> Model fitting took 10.1354069709778 seconds
#> The model has reached relative convergence
#> The model uses a exp kernel
#> Number of fixed effects in the model is 7
#> Number of random effects in the model is 100
#> Hessian matrix is positive definite. Use spatialGEV_sample to obtain posterior samples
```

Sampling from the joint posterior

Now, we show how to sample 2000 times from the joint posterior distribution of the GEV parameters using the function `spatialGEV_sample()`. Only three arguments need to be passed to this function: `model` takes in the list output by `spatialGEV_fit()`, `n_draw` is the number of samples to draw from the posterior distribution, and `observation` indicates whether to draw from the posterior predictive distribution of the data at the observed locations. Call `summary()` on the sample object to obtain summary statistics of the posterior samples.

```r
sam <- spatialGEV_sample(model = fit, n_draw = 2000, observation = T)
print(sam)
#> The samples contains 2000 draws of 107 parameters
#> The samples contains 2000 draws of response at 50 locations
#> Use summary() to obtain summary statistics of the samples
pos_summary <- summary(sam)
```

The samples are then used to calculate the posterior mean estimate of the 10-year return level $z_{10}(\boldsymbol{x})$ at each location, which are plotted against their true values in Figure 2.

```r
library(evd)
# True return levels
z_true <- unlist(Map(evd::qgev, p=0.1, loc=a, scale=exp(logb),
             shape=exp(logs), lower.tail=F))
# Posterior samples of return levels at all locations
```

```r
return_period <- 10
z_draws <- apply(sam$parameter_draws, 1,
                 function(all_draw){
                 mapply(evd::qgev, p=1/return_period,
                        loc=all_draw[paste0("a", 1:n_loc)],
                        scale=exp(all_draw[paste0("log_b", 1:n_loc)]),
                        shape=exp(all_draw["s"]),
                        lower.tail=F)
                 })
z_mean <- apply(z_draws, 1, mean)
plot(z_true, z_mean, xlab="True 10-year return levels",
     ylab="Posterior mean estimates of 10-year return levels")
abline(0, 1, lty="dashed", col="blue")
```

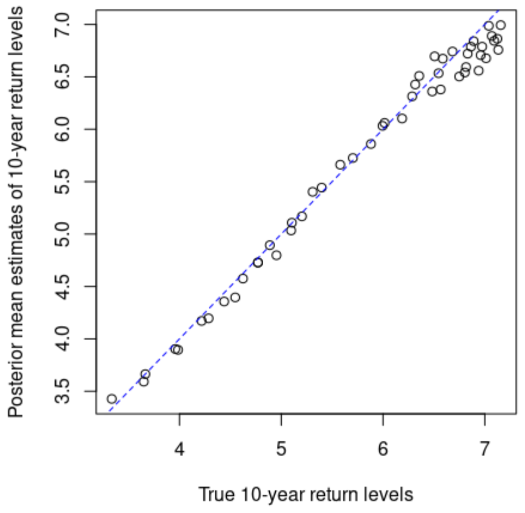

Figure 2: Posterior mean estimates of the 10-year return level $z_{10}(\boldsymbol{x})$ plotted against the true values at different locations.

Prediction at new locations

Next, we show how to predict the values of the extreme event at test locations. First, we divide the simulated dataset into training and test sets, and fit the model to the training dataset using the Matérn-SPDE kernel. We can simulate from the posterior predictive distribution of observations at the test locations using the `spatialGEV_predict()` function, which requires the fitted model to the training data passed to `model`, a matrix of the coordinates of the test locations passed to `locs_new`, and the number of simulation draws passed to `n_draw`. Figure 3 plots the 90% quantile values of the posterior predictive distributions against the 90% quantile values of the observations at all test locations.

```r
set.seed(123)
n_test <- 100                        # number of test locations
test_ind <- sample(1:400, n_test)    # indices of the test locations
locs_test <- simulatedData$locs[test_ind,]  # coordinates of the test locations
y_test <- simulatedData$y[test_ind]         # observations at the test locations
locs_train <- simulatedData$locs[-test_ind,]# coordinates of the training locations
```

```
y_train <- simulatedData$y[-test_ind]       # observations at the training locations

# Fit the GEV-GP model to the training set
train_fit <- spatialGEV_fit(data = y_train, locs = locs_train, random = "ab",
                            init_param = list(a = simulatedData$a[-test_ind],
                                              log_b = simulatedData$logb[-test_ind],
                                              s = -2,
                                              beta_a = 60, beta_b = 2,
                                              log_sigma_a = 0, log_kappa_a = -1,
                                              log_sigma_b = 0, log_kappa_b = -1),
                            reparam_s = "positive", kernel="spde", silent=T)

# Make predictions at the test locations
pred <- spatialGEV_predict(model = train_fit, locs_new = locs_test,
                           n_draw = 2000)
plot(sapply(y_test, quantile, probs=0.9),
     apply(pred$pred_y_draws, 2, quantile, probs=0.9),
     xlim=c(3,10), ylim=c(3,10),
     xlab="Observed 90% quantiles of responses at test locations",
     ylab="Predicted 90% quantiles of responses")
abline(0, 1, lty="dashed", col="blue")
```

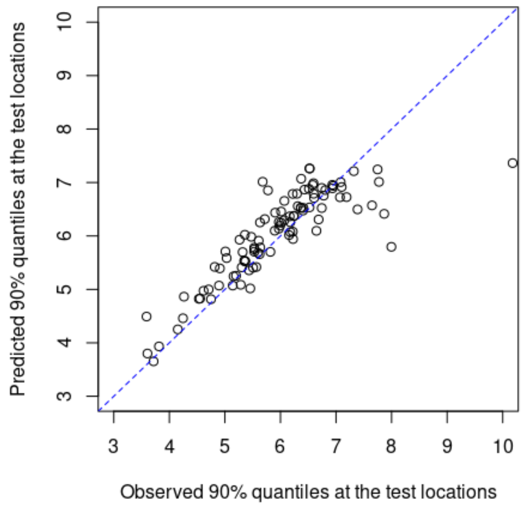

Figure 3: 90% quantile values of posterior predictive distributions at test locations plotted against the observed 90% quantile values at the corresponding locations. Each circle corresponds to a test location.

Acknowledgements

This work was supported by the Natural Sciences and Engineering Research Council of Canada, grant numbers RGPIN-2018-04376 (Ramezan), DGECR-2018-00349 (Ramezan) and RGPIN-2020-04364 (Lysy).

References

Chen, M., R. Ramezan, and M. Lysy. 2022. "Fast Approximate Inference for Spatial Extreme Value Models." https://arxiv.org/abs/2110.07051.

Coles, S. G., and E. Casson. 1998. "Extreme Value Modelling of Hurricane Wind Speeds." *Structural Safety* 20: 283–96.

Cooley, D., D. Nychka, and P. Naveau. 2007. "Bayesian Spatial Modeling of Extreme Precipitation Return Levels." *Journal of the American Statistical Association* 102: 824–40.

Dyrrdal, A. V., A. Lenkoski, T. L. Thorarinsdottir, and F. Stordal. 2015. "Bayesian Hierarchical Modeling of Extreme Hourly Precipitation in Norway." *Environmetrics* 26: 89–106.

Hoffman, M. D., and A. Gelman. 2014. "The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo." *Journal of Machine Learning Research* 15: 1593–623.

Kristensen, K., A. Nielsen, C. W. Berg, H. Skaug, and B. M Bell. 2016. "TMB: Automatic Differentiation and Laplace Approximation." *Journal of Statistical Software* 70 (5): 1–21.

Lindgren, F. K., and H. Rue. 2015. "Bayesian Spatial Modelling with R-INLA." *Journal of Statistical Software* 63: 1–25.

Lindgren, F. K., H. Rue, and J. Lindström. 2011. "An Explicit Link Between Gaussian Fields and Gaussian Markov Random Fields: The Stochastic Partial Differential Equation Approach." *Journal of the Royal Statistical Society, Series B* 73: 423–98.

Miller, D. L., R. Glennie, and A. E. Seaton. 2020. "Understanding the Stochastic Partial Differential Equation Approach to Smoothing." *Journal of Agricultural, Biological and Environmental Statistics* 25: 1–16.

Neal, R. M. 2011. "MCMC Using Hamiltonian Dynamics." In *The Handbook of Markov Chain Monte Carlo*. Chapman & Hall / CRC Press.

Ribatet, M., R. Singleton, and R Core team. 2022. *SpatialExtremes: Modelling Spatial Extremes*. Version 2.1-0. https://CRAN.R-project.org/package=SpatialExtremes.

Sang, H., and A. E. Gelfand. 2010. "Continuous Spatial Process Models for Spatial Extreme Values." *Journal of Agricultural, Biological, and Environmental Statistics* 15: 49–56.

Schliep, E. M., D. Cooley, S. R. Sain, and J. A. Hoeting. 2010. "A Comparison Study of Extreme Precipitation from Six Different Regional Climate Models via Spatial Hierarchical Modeling." *Extremes* 13: 219–39.

Stan Development Team. 2020. "RStan: The R Interface to Stan." http://mc-stan.org/.

Wood, S. N. 2023. *mgcv: Mixed GAM Computation Vehicle with Automatic Smoothness Estimation*. Version 1.9-1. https://CRAN.R-project.org/package=mgcv.

Youngman, Benjamin D. 2022. "Evgam: An R Package for Generalized Additive Extreme Value Models." *Journal of Statistical Software* 103: 1–26.