Optimisation with the r-pareto process

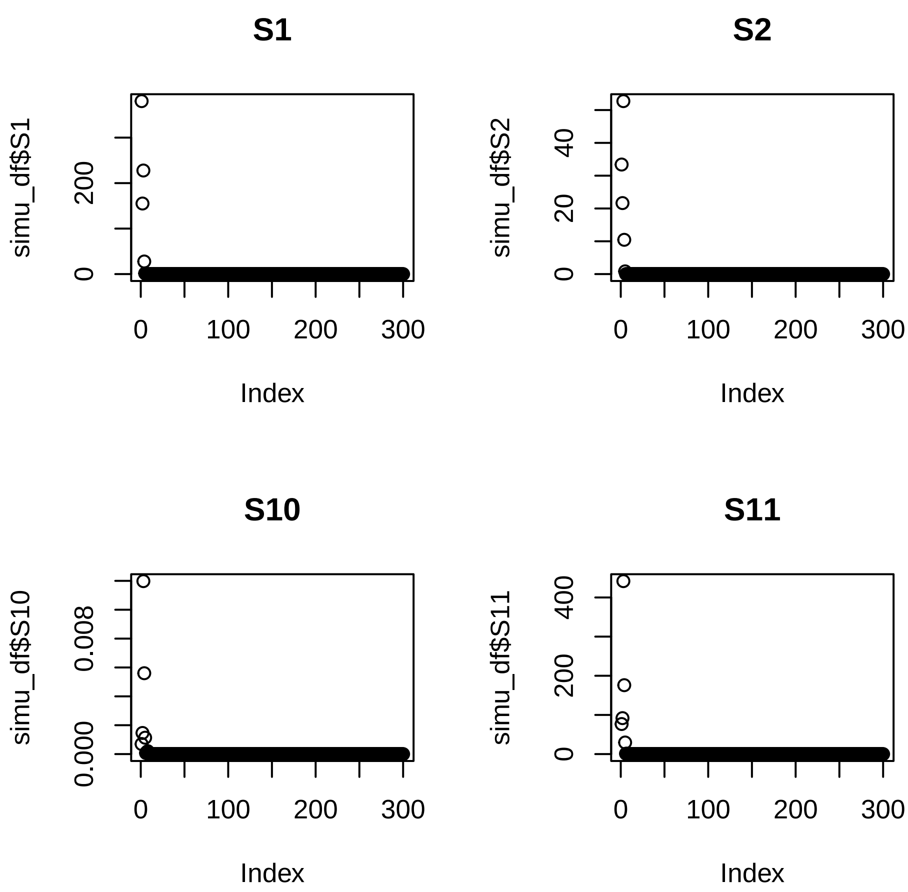
2024-09-24

Simulation of rpareto process

Risk function $r(X_{s,t}) = X_{s_0,t_0}$.

```
adv <- c(0.5, 0.3)
params <- c(0.4, 0.2, 1, 1.2) # ok verif sur simu
true_param <- c(params, adv)
beta1 <- 0.4
beta2 <- 0.2
alpha1 <- 1.5
alpha2 <- 1
ngrid <- 5
spa <- 1:ngrid
nsites <- ngrid^2 # if the grid is squared
temp <- 1:300

# Conditional point
s0 <- c(1, 1)
t0 <- 1

# Number of realizations
nres <- 100

# Simulate the process
# simu <- sim_rpareto(beta1, beta2, alpha1, alpha2, spa, spa, temp, adv, s0,
#                      t0, nres)

adv_int <- adv * 10
adv_str <- sprintf("%02d_%02d", adv_int[1], adv_int[2])

# Save the data
foldername <- paste0("../data/simulations_rpar/sim_", ngrid^2, "s_",
                                length(temp), "t_", adv_str, "/")


if (!dir.exists(foldername)) {
  dir.create(foldername, recursive = TRUE)
}

# save_simulations(simu, ngrid, nres, folder = foldername,
#         file = paste0("rpar_", ngrid^2, "s_", length(temp), "t"))
```

```r
list_simu <- list()
for (i in 1:nres) {
    file_name <- paste0(foldername, "rpar_", ngrid^2, "s_",
                        length(temp), "t_", i, ".csv")
  list_simu[[i]] <- read.csv(file_name)
}

# Plot the first realization
simu_df <- list_simu[[1]]
par(mfrow = c(2, 2))
plot(simu_df$S1, main = "S1")
plot(simu_df$S2, main = "S2")
plot(simu_df$S10, main = "S10")
plot(simu_df$S11, main = "S11")
```




```r
# library(animation)
# create_simu_gif(simu_df, true_param, "rpar", forcedtemp = 30)
```

Get conditional lag vectors

```r
sites_coords <- generate_grid_coords(ngrid)
df_lags <- get_conditional_lag_vectors(sites_coords, true_param, s0, t0,
```

```r
                                    hmax = sqrt(17), tau_vect = 0:10)
head(df_lags)
```

```
##   s1 s2   h1   h2 tau     hnorm
## 1  1  1  0.0  0.0   0 0.0000000
## 2  1  1 -0.5 -0.3   1 0.5830952
## 3  1  1 -1.0 -0.6   2 1.1661904
## 4  1  1 -1.5 -0.9   3 1.7492856
## 5  1  1 -2.0 -1.2   4 2.3323808
## 6  1  1 -2.5 -1.5   5 2.9154759
```

Optimisation of variogram parameters

```r
# Optimisation of the variogram parameters
q_values <- seq(0.90, 0.97, 0.01)

result_table <- data.frame(q = numeric(), beta1 = numeric(), beta2 = numeric(),
                           alpha1 = numeric(), alpha2 = numeric(),
                           adv1 = numeric(), adv2 = numeric())

# For each quantile
for (q in q_values) {
  # Get the empirical excesses
  excesses <- empirical_excesses(simu_df, quantile = q, df_lags = df_lags)

  # Optimization
  result <- optim(par = c(true_param), fn = neg_ll,
                  data = simu_df,
                  quantile = q,
                  df_lags = df_lags,
                  excesses = excesses,
                  locations = sites_coords,
                  hmax = sqrt(17),
                  s0 = s0,
                  t0 = t0,
                  method = "BFGS",
                  control = list(parscale = c(1, 1, 1, 1, 1, 1),
                                 maxit = 10000))

  # Check convergence
  if (result$convergence == 0) {
    result_table <- rbind(result_table,
                          data.frame(q = q,
                                     beta1 = result$par[1],
                                     beta2 = result$par[2],
                                     alpha1 = result$par[3],
                                     alpha2 = result$par[4],
                                     adv1 = result$par[5],
                                     adv2 = result$par[6]))
  } else {
```

```r
    # In case of non-convergence, store NAs
    result_table <- rbind(result_table,
        data.frame(q = q, beta1 = NA, beta2 = NA, alpha1 = NA, alpha2 = NA,
                   adv1 = NA, adv2 = NA))
  }
}

result_table <- round(result_table, 5)
df_rmse <- data.frame(q = result_table$q,
                rmse_beta1 = sqrt((result_table$beta1 - true_param[1])^2),
                rmse_beta2 = sqrt((result_table$beta2 - true_param[2])^2),
                rmse_alpha1 = sqrt((result_table$alpha1 - true_param[3])^2),
                rmse_alpha2 = sqrt((result_table$alpha2 - true_param[4])^2),
                rmse_adv1 = sqrt((result_table$adv1 - true_param[5])^2),
                rmse_adv2 = sqrt((result_table$adv2 - true_param[6])^2))

kable(result_table, "latex", booktabs = TRUE,
      caption = "Optim estimations for each quantile for one simulation")  %>%
  kable_styling(latex_options = "H",
    bootstrap_options = c("striped", "hover", "condensed", "responsive"))
```

Table 1: Optim estimations for each quantile for one simulation

q	beta1	beta2	alpha1	alpha2	adv1	adv2
0.90	0.32074	0.00031	0.59598	1.13230	0.09993	0.29509
0.91	0.19384	0.00055	1.06560	1.15107	0.47634	0.29555
0.92	0.31666	0.00035	0.42563	1.16208	0.20403	0.22730
0.93	0.31693	0.00065	0.97335	1.17467	0.47813	0.29579
0.94	0.31553	0.00057	0.97517	1.20036	0.48058	0.29645
0.95	0.31535	0.00080	0.97568	1.20802	0.48134	0.29666
0.96	0.31166	0.00062	0.97690	1.23374	0.48270	0.29652
0.97	0.31259	0.00099	0.97672	1.23108	0.48293	0.28140

```r
kable(df_rmse, "latex", booktabs = TRUE,
      caption = "RMSE for each parameter and
                different quantiles for one simulation")  %>%
  kable_styling(latex_options = "H",
     bootstrap_options = c("striped", "hover", "condensed", "responsive"))
```

Table 2: RMSE for each parameter and different quantiles for one simulation

q	rmse_beta1	rmse_beta2	rmse_alpha1	rmse_alpha2	rmse_adv1	rmse_adv2
0.90	0.07926	0.19969	0.40402	0.06770	0.40007	0.00491
0.91	0.20616	0.19945	0.06560	0.04893	0.02366	0.00445
0.92	0.08334	0.19965	0.57437	0.03792	0.29597	0.07270
0.93	0.08307	0.19935	0.02665	0.02533	0.02187	0.00421
0.94	0.08447	0.19943	0.02483	0.00036	0.01942	0.00355
0.95	0.08465	0.19920	0.02432	0.00802	0.01866	0.00334
0.96	0.08834	0.19938	0.02310	0.03374	0.01730	0.00348
0.97	0.08741	0.19901	0.02328	0.03108	0.01707	0.01860

Combining simulations together

```
neg_ll_composite <- function(params, data, df_lags, locations, quantile,
                    excesses, latlon = FALSE, hmax = NA, nsample = 1, s0 = NA,
                    t0 = NA) {
#   print(params)
  ntemp <- nrow(data) / nsample # number of time steps in each simulation
  ind_s0 <- which(locations$Latitude == s0[1] && locations$Longitude == s0[2])
  nmarg <- get_marginal_excess(data, quantile, ind_s0, t0)
  pmarg <- nmarg / nrow(data)
  nll_composite <- 0 # composite negative log-likelihood
  for (i in 1:nsample) {
    # extract simulation data from i-th simulation
    simu <- data[((i - 1) * ntemp + 1):(i * ntemp), ]
    # excesses <- list_excesses[[i]]
    nll_i <- neg_ll(params, simu, df_lags, locations, quantile,
                    latlon = latlon, hmax = hmax,
                    excesses = excesses, s0 = s0, t0 = t0, pmarg = pmarg)
    nll_composite <- nll_composite + nll_i
  }

  return(nll_composite)
}

# Combine all simulations
simu_all <- do.call(rbind, list_simu)

quantile <- 0.9
excesses_all <- empirical_excesses(simu_all, quantile, df_lags)

result <- optim(par = c(true_param), fn = neg_ll_composite,
                  data = simu_all,
                  quantile = quantile,
                  df_lags = df_lags,
                  excesses = excesses_all,
                  locations = sites_coords,
                  hmax = sqrt(17),
```

```
                  s0 = s0,
                  t0 = t0,
                  nsample = nres,
                  method = "BFGS",
                  control = list(parscale = c(1, 1, 1, 1, 1, 1),
                                 maxit = 10000))

df_result <- data.frame(beta1 =  result$par[1],
                        beta2 = result$par[2],
                        alpha1 = result$par[3],
                        alpha2 = result$par[4],
                        adv1 = result$par[5],
                        adv2 = result$par[6])

df_rmse <- data.frame(beta1 = sqrt((result$par[1] - true_param[1])^2),
                beta2 = sqrt((result$par[2] - true_param[2])^2),
                alpha1 = sqrt((result$par[3] - true_param[3])^2),
                alpha2 = sqrt((result$par[4] - true_param[4])^2),
                adv1 = sqrt((result$par[5] - true_param[5])^2),
                adv2 = sqrt((result$par[6] - true_param[6])^2))

df_result <- rbind(df_result, df_rmse)
rownames(df_result) <- c("estim", "rmse")
kable(df_result, "latex", booktabs = TRUE,
        caption = "RMSE for all simulations together")  %>%
    kable_styling(latex_options = "H",
        bootstrap_options = c("striped", "hover", "condensed", "responsive"))
```

Table 3: RMSE for all simulations together

	beta1	beta2	alpha1	alpha2	adv1	adv2
estim	0.3191349	0.0009114	0.9719815	1.1515308	0.4763334	0.2954677
rmse	0.0808651	0.1990886	0.0280185	0.0484692	0.0236666	0.0045323