Drought prediction in continental Italy: an SpVAR approach

Drought prediction in continental Italy: an SpVAR approach

We set the seed for reproducibility and import the necessary libraries and functions.

```
set.seed(123)

suppressPackageStartupMessages({
  library(dplyr)
  library(tidyr)
  library(ggplot2)
  library(systemfit)
  library(plm)
  library(ncdf4)
  library(abind)
  library(ggmap)
  library(Rmpfr)
  library(gridExtra)
  library(geometry)
  library(caret)
  library(cowplot)
  library(forecast)
})

setwd(
  "/Users/enrico/Library/Mobile Documents/com~apple~CloudDocs/Documents/Carriera/Scuola:Università/Bocconi Master/Thesis"
)

source("functions.r")
```

We read the SPEI data and prepare it.

```
# Open the netCDF file with monthly SPEI data
SPEI <- nc_open("Data/spei12.nc")

# Get the data from the first variable
SPEI_data <- ncvar_get(SPEI, attributes(SPEI$var)$names[1])

# Get the longitude and latitude values
SPEI_lon <- ncvar_get(SPEI, attributes(SPEI$dim)$names[1])
SPEI_lat <- ncvar_get(SPEI, attributes(SPEI$dim)$names[2])
```

```r
# Close the file
nc_close(SPEI)

# Subset the data for the relevant time steps
SPEI_last_year<-SPEI_data[,,1367:1368]
SPEI_relevant_years <- SPEI_data[, , 967:1368]
```

We read the SMoist data and prepare it.

```r
# Open the netCDF file with Soil Moisture data
SMoist <- nc_open("Data/SoilMoisture.nc")

# Get the data from the first variable
SMoist_data <- ncvar_get(SMoist, attributes(SMoist$var)$names[1])

# Order the data with the same ordering as the SPEI
SMoist_ordered <- SMoist_data[361:720, ,]
SMoist_ordered <-
  abind(SMoist_ordered, SMoist_data[1:360, ,], along = 1)

# Get the longitude and latitude values
SMoist_lon <- ncvar_get(SMoist, attributes(SMoist$dim)$names[1])
SMoist_lat <- ncvar_get(SMoist, attributes(SMoist$dim)$names[3])

# Print a message to check if the longitudes and latitudes match with the ones from SPEI
print(paste(
  "Check on longitudes",
  toString(all(SMoist_lon - 180 == SPEI_lon)),
  "check on latitudes",
  toString(all(SMoist_lat == SPEI_lat))
))
```

```
## [1] "Check on longitudes TRUE check on latitudes TRUE"
```

```r
# Close the file
nc_close(SMoist)

# Subset the data for the relevant time steps
SMoist_last_year <- SMoist_ordered[, , 659:660]
SMoist_relevant_years <- SMoist_ordered[, , 259:660]
```

We read the NDVI data and prepare it.

```r
# Open the netCDF file with NDVI data
NDVI_read <- nc_open("Data/NDVI.nc")

# Get the data from the first variable
NDVI <- ncvar_get(NDVI_read, attributes(NDVI_read$var)$names[1])

# Get the longitude and latitude values
NDVI_lon <- ncvar_get(NDVI_read, attributes(NDVI_read$dim)$names[1])
```

```r
NDVI_lat <- ncvar_get(NDVI_read, attributes(NDVI_read$dim)$names[2])

# Print a message to check if the longitudes and latitudes match with the ones from SPEI
print(paste(
  "Check on longitudes",
  all(NDVI_lon == SPEI_lon),
  "check on latitudes",
  all(NDVI_lat == SPEI_lat)
))
```

```
## [1] "Check on longitudes TRUE check on latitudes TRUE"
```

```r
# Close the file
nc_close(NDVI_read)

# Subset the data for the relevant time steps
NDVI_last_year <- NDVI[, , 401:402]
```

We read the PET data and prepare it.

```r
# Open the first netCDF file with monthly PET data and store it
PET <- nc_open("Data/cru_ts4.03.1981.1990.pet.dat.nc")
PET_data <- ncvar_get(PET, attributes(PET$var)$names[1])

# Subset the data for the relevant time steps
PET_data <- PET_data[, , 7:120]

# Close the file
nc_close(PET)

# Open the second netCDF file with monthly PET data and store it
PET <- nc_open("Data/cru_ts4.03.1991.2000.pet.dat.nc")
PET_data_2 <- ncvar_get(PET, attributes(PET$var)$names[1])

# Close the file
nc_close(PET)

# Open the third netCDF file with monthly PET data and store it
PET <- nc_open("Data/cru_ts4.03.2001.2010.pet.dat.nc")
PET_data_3 <- ncvar_get(PET, attributes(PET$var)$names[1])

# Close the file
nc_close(PET)

# Open the fourth netCDF file with monthly PET data and store it
PET <- nc_open("Data/cru_ts4.03.2011.2018.pet.dat.nc")
PET_data_4 <- ncvar_get(PET, attributes(PET$var)$names[1])

# Subset the data for the relevant time steps
PET_data_4 <- PET_data_4[, , 1:48]
```

```r
# Get the longitude and latitude values
PET_lon <- ncvar_get(PET, attributes(PET$dim)$names[1])
PET_lat <- ncvar_get(PET, attributes(PET$dim)$names[2])

# Print a message to check if the longitudes and latitudes match with the ones from SPEI
print(paste(
  "Check on longitudes",
  all(PET_lon == SPEI_lon),
  "check on latitudes",
  all(PET_lat == SPEI_lat)
))
```

```
## [1] "Check on longitudes TRUE check on latitudes TRUE"
```

```r
# Close the file
nc_close(PET)

# Merge the 4 PET datasets together
PET <- array(0, dim = c(720, 360, 402))
PET[, , ] <- c(PET_data, PET_data_2, PET_data_3, PET_data_4)
```

We read the PRE (Monthly Precipitation) data and prepare it.

```r
# Open the first netCDF file with monthly Precipitation dataand store it
PRE <- nc_open("Data/cru_ts4.03.1981.1990.pre.dat.nc")
PRE_data <- ncvar_get(PRE, attributes(PRE$var)$names[1])

# Subset the data for the relevant time steps
PRE_data<-PRE_data[,,7:120]

# Close the file
nc_close(PRE)

# Open the second netCDF file with monthly Precipitation data and store it
PRE<-nc_open("Data/cru_ts4.03.1991.2000.pre.dat.nc")
PRE_data_2 <- ncvar_get(PRE, attributes(PRE$var)$names[1])

# Close the file
nc_close(PRE)

# Open the third netCDF file with monthly Precipitation data and store it
PRE<-nc_open("Data/cru_ts4.03.2001.2010.pre.dat.nc")
PRE_data_3 <- ncvar_get(PRE, attributes(PRE$var)$names[1])

# Close the file
nc_close(PRE)

# Open the fourth netCDF file with monthly Precipitation data and store it
PRE<-nc_open("Data/cru_ts4.03.2011.2018.pre.dat.nc")
PRE_data_4 <- ncvar_get(PRE, attributes(PRE$var)$names[1])
```

```r
# Subset the data for the relevant time steps
PRE_data_4<-PRE_data_4[,,1:48]

# Get the longitude and latitude values
PRE_lon <- ncvar_get(PRE, attributes(PRE$dim)$names[1])
PRE_lat <- ncvar_get(PRE, attributes(PRE$dim)$names[2])

# Print a message to check if the longitudes and latitudes match with the ones from SPEI
print(paste(
  "Check on longitudes",
  all(PRE_lon == SPEI_lon),
  "check on latitudes",
  all(PRE_lat == SPEI_lat)
))
```

```
## [1] "Check on longitudes TRUE check on latitudes TRUE"
```

```r
# Close the file
nc_close(PRE)

# Merge the 4 Precipitation datasets together
PRE<-array(0, dim=c(720,360,402))
PRE[,,]<-c(PRE_data, PRE_data_2,PRE_data_3,PRE_data_4)
```

We clean-up the workspace by removing unnecessary objects.

```r
# Remove the variables SPEI and SMoist from the workspace
rm(SPEI, SMoist)

# Rename the variables SPEI and Smoist variables
SPEI <- SPEI_relevant_years
SMoist <- SMoist_relevant_years

# Assign the variables lats and lons to the latitude and longitude of SPEI
lats <- SPEI_lat
lons <- SPEI_lon

# Remove variables that won't be used again from the workspace
rm(
  SPEI_relevant_years,
  SMoist_relevant_years,
  SPEI_lat,
  SMoist_lat,
  SPEI_lon,
  SMoist_lon,
  SMoist_ordered,
  NDVI_lon,
  NDVI_lat,
  SPEI_data,
  SMoist_data,
  NDVI_read,
```

```
  PET_lat,
  PET_lon,
  PRE_lat,
  PRE_lon,
  PET_data,
  PET_data_2,
  PET_data_3,
  PET_data_4,
  PRE_data,
  PRE_data_2,
  PRE_data_3,
  PRE_data_4
)
```

We create arrays containing the latitudes and longitudes of the region (Continental Italy) we'll analyze.

```
reg_latitudes <-
  c(
    46.75,
    46.25,
    46.25,
    45.75,
    45.25,
    44.75,
    44.25,
    43.75,
    43.75,
    43.25,
    42.75,
    42.25,
    41.75,
    41.25,
    40.75,
    40.25,
    39.75,
    39.25,
    38.75,
    38.25
  )
reg_long_min <-
  c(
    10.25,
    8.25,
    9.25,
    6.75,
    6.75,
    6.75,
    6.75,
    7.25,
    10.25,
```

```r
    10.25,
    10.75,
    11.75,
    12.25,
    13.25,
    14.25,
    15.25,
    15.75,
    16.25,
    16.25,
    15.75
  )

reg_long_max <-
  c(
    12.75,
    8.25,
    13.75,
    13.75,
    12.25,
    12.25,
    12.25,
    7.75,
    13.75,
    13.75,
    14.25,
    14.25,
    15.75,
    16.75,
    17.75,
    18.25,
    16.25,
    16.75,
    16.75,
    16.25
  )
```

We now scale the data and then format it as needed.

```r
test_df <- create_test_df(reg_latitudes,
                          reg_long_min,
                          reg_long_max,
                          lats,
                          lons,
                          SPEI,
                          SMoist,
                          NDVI)

#We store the scaling factors for future re-use
scale_SMoist <- sd(test_df$SMoist)
```

```r
scale_NDVI <- sd(test_df$NDVI, na.rm = TRUE)
#Since we don't scale the SPEI value we set the scale factor to 1
scale_SPEI <- 1

#We scale the data
test_df$SMoist <- (test_df$SMoist) / (scale_SMoist)
test_df$NDVI <- (test_df$NDVI) / (scale_NDVI)

#We create a dataframe for each of the variables
test_df_spei <-
  data.frame(split(test_df$SPEI, as.factor(test_df$index)))
test_df_ndvi <-
  data.frame(split(test_df$NDVI, as.factor(test_df$index)))
test_df_smoist <-
  data.frame(split(test_df$SMoist, as.factor(test_df$index)))
```

We create two other objects needed in the future analysis (the regions indexes and the weights matrix).

```r
#index_pairs contains the Longitudes and Latitudes indexes of all the regions under consideration
index_pairs <- unique(test_df$index)

#weights is the connectivity matrix
weights <- prepare_weights(index_pairs, lats, lons)
```

Unit Root testing

We now perform unit root testing as described in Section 3.3. We first look at the estimated \(\lambda\) for the SPEI variable on our data.

```r
compute_lambda(test_df_spei, weights)
```

```
## [1] 0.9071363
```

We then look at the estimated \(\lambda\) for the SMoist variable.

```r
compute_lambda(test_df_smoist, weights)
```

```
## [1] 0.8726805
```

We also look at the estimated \(\lambda\) for the NDVI variable.

```r
compute_lambda(test_df_ndvi, weights)
```

```
## [1] 0.3985638
```

Finally, we perform a Monte Carlo simulation to obtain the critical values of our test.

```r
lambda_mc <-
  mc_critical_values(10000, 139, 500, weights, 3061792, FALSE)
```

```
## [1] "We reject in the test 1 if lambda < 0.991387964490184"
## [1] "We reject in the test 2 if lambda < 0.996196614372254"
## [1] "We reject in the test 3 if lambda < 1.02291769144339"
## [1] "We reject in the test 4 if lambda < 1.05695372345613"
## [1] "We reject in the test 5 if lambda < 1.09510614689225"
## [1] "We reject in the test 6 if lambda < 1.13546696878769"
```

```
## [1] "We reject in the test 7 if lambda < 1.17463616654881"
## [1] "We reject in the test 8 if lambda < 1.20721156481164"
## [1] "We reject in the test 9 if lambda < 1.2234286905795"
## [1] "We reject in the test 10 if lambda < 1.2067064741227"
```

Model fitting

We fit our model using SUR (implemented in the systemfit package) after having prepared the data for it.

```
spvar_df <-
  prepare_spvar_df(test_df, test_df_spei, test_df_smoist, test_df_ndvi, lags =
                     1)


spvar_fit <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist,
      eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist
    ),
    data = spvar_df,
    method = "SUR"
  )
```

We also fit a model with two temporal lags to compare them.

```
spvar_df_2 <-
  prepare_spvar_df(test_df, test_df_spei, test_df_smoist, test_df_ndvi, lags =
                     2)

spvar_fit_2 <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist,
      eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist
    ),
    data = spvar_df_2,
    method = "SUR"
  )
```

To make this comparison and choose which one to use we use BIC.

```
print(paste( "The BIC of the first model is", BIC(spvar_fit)))
```

```
## [1] "The BIC of the first model is 131207.193675043"
```

```
print(paste( "The BIC of the second model is", BIC(spvar_fit_2)))
```

```
## [1] "The BIC of the second model is 143753.030804394"
```

Since the BIC of the first model is substantially lower we choose it. We now look at the coefficients of this chosen model.

```r
summary(spvar_fit)
```

```
##
## systemfit results
## method: SUR
##
##             N     DF     SSR  detRCov   OLS-R2 McElroy-R2
## system 145287 144852 53020.1 0.010642 0.617155         NA
##
##              N    DF      SSR      MSE     RMSE       R2   Adj R2
## eqspei   48789 48644  6872.68 0.141285 0.375879 0.840740 0.840269
## eqndvi   47709 47564 41672.47 0.876135 0.936021 0.120492 0.117830
## eqsmoist 48789 48644  4474.95 0.091994 0.303305 0.906683 0.906407
##
## The covariance matrix of the residuals used for estimation
##               eqspei     eqndvi    eqsmoist
## eqspei    0.14142911 -0.0227940  0.00510905
## eqndvi   -0.02279404  0.8761230 -0.07153691
## eqsmoist  0.00510905 -0.0715369  0.09216100
##
## The covariance matrix of the residuals
##               eqspei     eqndvi    eqsmoist
## eqspei    0.14142885 -0.0227889  0.00510902
## eqndvi   -0.02278892  0.8761347 -0.07155297
## eqsmoist  0.00510902 -0.0715530  0.09216346
##
## The correlations of the residuals
##              eqspei     eqndvi  eqsmoist
## eqspei    1.0000000 -0.0647382  0.044745
## eqndvi   -0.0647382  1.0000000 -0.251801
## eqsmoist  0.0447450 -0.2518006  1.000000
##
##
## SUR estimates for 'eqspei' (equation 1)
## Model Formula: spei ~ 0 + (ndvi + smoist + fe.1 + fe.2 + fe.3 + fe.4 + fe.5 +
##     fe.6 + fe.7 + fe.8 + fe.9 + fe.10 + fe.11 + fe.12 + fe.13 +
##     fe.14 + fe.15 + fe.16 + fe.17 + fe.18 + fe.19 + fe.20 + fe.21 +
##     fe.22 + fe.23 + fe.24 + fe.25 + fe.26 + fe.27 + fe.28 + fe.29 +
##     fe.30 + fe.31 + fe.32 + fe.33 + fe.34 + fe.35 + fe.36 + fe.37 +
##     fe.38 + fe.39 + fe.40 + fe.41 + fe.42 + fe.43 + fe.44 + fe.45 +
##     fe.46 + fe.47 + fe.48 + fe.49 + fe.50 + fe.51 + fe.52 + fe.53 +
##     fe.54 + fe.55 + fe.56 + fe.57 + fe.58 + fe.59 + fe.60 + fe.61 +
##     fe.62 + fe.63 + fe.64 + fe.65 + fe.66 + fe.67 + fe.68 + fe.69 +
##     fe.70 + fe.71 + fe.72 + fe.73 + fe.74 + fe.75 + fe.76 + fe.77 +
##     fe.78 + fe.79 + fe.80 + fe.81 + fe.82 + fe.83 + fe.84 + fe.85 +
##     fe.86 + fe.87 + fe.88 + fe.89 + fe.90 + fe.91 + fe.92 + fe.93 +
##     fe.94 + fe.95 + fe.96 + fe.97 + fe.98 + fe.99 + fe.100 +
##     fe.101 + fe.102 + fe.103 + fe.104 + fe.105 + fe.106 + fe.107 +
##     fe.108 + fe.109 + fe.110 + fe.111 + fe.112 + fe.113 + fe.114 +
```

```
##     fe.115 + fe.116 + fe.117 + fe.118 + fe.119 + fe.120 + fe.121 +
##     fe.122 + fe.123 + fe.124 + fe.125 + fe.126 + fe.127 + fe.128 +
##     fe.129 + fe.130 + fe.131 + fe.132 + fe.133 + fe.134 + fe.135 +
##     fe.136 + fe.137 + fe.138 + fe.139 + spei_lag + ndvi_lag +
##     smoist_lag + sp_spei + sp_ndvi + sp_smoist) - spei - ndvi -
##     smoist
##
##               Estimate  Std. Error   t value   Pr(>|t|)
## fe.1       -0.01183545  0.02259684  -0.52377 0.60044381
## fe.2       -0.02262707  0.02203690  -1.02678 0.30452861
## fe.3       -0.03930379  0.02185573  -1.79833 0.07213104 .
## fe.4       -0.03464724  0.02189803  -1.58221 0.11360852
## fe.5       -0.04546455  0.02174939  -2.09038 0.03658863 *
## fe.6       -0.04069721  0.02159535  -1.88454 0.05949851 .
## fe.7       -0.04652168  0.02156243  -2.15753 0.03096891 *
## fe.8       -0.05870931  0.02161661  -2.71594 0.00661121 **
## fe.9       -0.06149561  0.02141683  -2.87137 0.00408876 **
## fe.10      -0.07710600  0.02200197  -3.50450 0.00045787 ***
## fe.11      -0.07749231  0.02205000  -3.51439 0.00044117 ***
## fe.12      -0.07795694  0.02149009  -3.62758 0.00028639 ***
## fe.13      -0.06723482  0.02095264  -3.20890 0.00133332 **
## fe.14      -0.05821449  0.02083883  -2.79356 0.00521520 **
## fe.15      -0.03351792  0.02153550  -1.55640 0.11961886
## fe.16      -0.06305272  0.02155620  -2.92504 0.00344572 **
## fe.17      -0.07262124  0.02179201  -3.33247 0.00086142 ***
## fe.18      -0.09100797  0.02222406  -4.09502 4.2281e-05 ***
## fe.19      -0.09544689  0.02241314  -4.25852 2.0617e-05 ***
## fe.20      -0.09263960  0.02214717  -4.18291 2.8831e-05 ***
## fe.21      -0.08390869  0.02160800  -3.88322 0.00010322 ***
## fe.22      -0.06571049  0.02093290  -3.13910 0.00169568 **
## fe.23      -0.05389713  0.02097544  -2.56954 0.01018646 *
## fe.24      -0.06313391  0.02156410  -2.92773 0.00341602 **
## fe.25      -0.07319709  0.02202954  -3.32268 0.00089223 ***
## fe.26      -0.08625032  0.02260214  -3.81602 0.00013579 ***
## fe.27      -0.09131502  0.02237140  -4.08178 4.4764e-05 ***
## fe.28      -0.10294516  0.02254870  -4.56546 4.9962e-06 ***
## fe.29      -0.10172229  0.02231996  -4.55746 5.1902e-06 ***
## fe.30      -0.08339154  0.02134283  -3.90724 9.3483e-05 ***
## fe.31      -0.06610976  0.02088618  -3.16524 0.00155050 **
## fe.32      -0.06163741  0.02158465  -2.85561 0.00429720 **
## fe.33      -0.07392344  0.02235515  -3.30677 0.00094445 ***
## fe.34      -0.08405280  0.02306361  -3.64439 0.00026830 ***
## fe.35      -0.09037754  0.02342383  -3.85836 0.00011430 ***
## fe.36      -0.09274141  0.02304804  -4.02383 5.7345e-05 ***
## fe.37      -0.08913074  0.02201931  -4.04784 5.1771e-05 ***
## fe.38      -0.09040799  0.02164445  -4.17696 2.9595e-05 ***
## fe.39      -0.07283610  0.02101528  -3.46586 0.00052899 ***
## fe.40      -0.05827551  0.02182718  -2.66986 0.00759082 **
## fe.41      -0.08432885  0.02333687  -3.61355 0.00030234 ***
```

```
## fe.42      −0.09149210  0.02385832  -3.83481 0.00012582 ***
## fe.43      −0.09561791  0.02376359  -4.02372 5.7374e-05 ***
## fe.44      −0.09580446  0.02370435  -4.04164 5.3160e-05 ***
## fe.45      −0.08525938  0.02264917  -3.76435 0.00016718 ***
## fe.46      −0.06628662  0.02179106  -3.04192 0.00235201 **
## fe.47      −0.07446181  0.02274654  -3.27354 0.00106282 **
## fe.48      −0.08637753  0.02400696  -3.59802 0.00032097 ***
## fe.49      −0.09219954  0.02464140  -3.74165 0.00018303 ***
## fe.50      −0.09588738  0.02450102  -3.91361 9.1051e-05 ***
## fe.51      −0.09946895  0.02413193  -4.12188 3.7641e-05 ***
## fe.52      −0.09599468  0.02400490  -3.99896 6.3715e-05 ***
## fe.53      −0.05947229  0.02348294  -2.53257 0.01132597 *
## fe.54      −0.04477553  0.02360326  -1.89701 0.05783308 .
## fe.55      −0.08141773  0.02427065  -3.35458 0.00079548 ***
## fe.56      −0.08864370  0.02453237  -3.61334 0.00030259 ***
## fe.57      −0.09287429  0.02499849  -3.71520 0.00020327 ***
## fe.58      −0.09230880  0.02491885  -3.70438 0.00021215 ***
## fe.59      −0.08791247  0.02429194  -3.61900 0.00029605 ***
## fe.60      −0.08990450  0.02398859  -3.74780 0.00017860 ***
## fe.61      −0.08271892  0.02298823  -3.59832 0.00032060 ***
## fe.62      −0.06361303  0.02147161  -2.96266 0.00305143 **
## fe.63      −0.06611646  0.02137874  -3.09263 0.00198505 **
## fe.64      −0.07980833  0.02379633  -3.35381 0.00079768 ***
## fe.65      −0.08925187  0.02490797  -3.58327 0.00033966 ***
## fe.66      −0.09278291  0.02537060  -3.65710 0.00025535 ***
## fe.67      −0.09085287  0.02514619  -3.61299 0.00030300 ***
## fe.68      −0.09225742  0.02476734  -3.72496 0.00019556 ***
## fe.69      −0.07438808  0.02321059  -3.20492 0.00135187 **
## fe.70      −0.07956095  0.02298645  -3.46121 0.00053821 ***
## fe.71      −0.04698539  0.02371432  -1.98131 0.04756227 *
## fe.72      −0.05720536  0.02154458  -2.65521 0.00792852 **
## fe.73      −0.08914541  0.02248838  -3.96407 7.3789e-05 ***
## fe.74      −0.09311164  0.02299749  -4.04877 5.1566e-05 ***
## fe.75      −0.09047707  0.02284122  -3.96113 7.4702e-05 ***
## fe.76      −0.08452781  0.02306750  -3.66437 0.00024821 ***
## fe.77      −0.08322574  0.02342017  -3.55359 0.00038037 ***
## fe.78      −0.07847685  0.02373804  -3.30595 0.00094722 ***
## fe.79      −0.09232268  0.02532833  -3.64504 0.00026763 ***
## fe.80      −0.09253221  0.02560068  -3.61444 0.00030130 ***
## fe.81      −0.09503188  0.02600620  -3.65420 0.00025826 ***
## fe.82      −0.09138468  0.02488921  -3.67166 0.00024124 ***
## fe.83      −0.08505867  0.02396912  -3.54868 0.00038754 ***
## fe.84      −0.05931886  0.02200658  -2.69551 0.00703059 **
## fe.85      −0.08657876  0.02313011  -3.74312 0.00018196 ***
## fe.86      −0.09904682  0.02386416  -4.15044 3.3240e-05 ***
## fe.87      −0.10404560  0.02450323  -4.24620 2.1783e-05 ***
## fe.88      −0.09895034  0.02471956  -4.00292 6.2659e-05 ***
## fe.89      −0.10528932  0.02496481  -4.21751 2.4746e-05 ***
## fe.90      −0.10406835  0.02574255  -4.04266 5.2930e-05 ***
```

```
## fe.91      -0.10016060  0.02582251  -3.87881 0.00010511 ***
## fe.92      -0.10211773  0.02658243  -3.84155 0.00012241 ***
## fe.93      -0.10232607  0.02697612  -3.79321 0.00014889 ***
## fe.94      -0.09516635  0.02567061  -3.70721 0.00020979 ***
## fe.95      -0.08085568  0.02380964  -3.39592 0.00068452 ***
## fe.96      -0.05054149  0.02197040  -2.30044 0.02142774 *
## fe.97      -0.07664014  0.02336601  -3.27998 0.00103886 **
## fe.98      -0.08772090  0.02398581  -3.65720 0.00025525 ***
## fe.99      -0.09590126  0.02483334  -3.86179 0.00011270 ***
## fe.100     -0.09761283  0.02524094  -3.86724 0.00011022 ***
## fe.101     -0.10447717  0.02631025  -3.97097 7.1684e-05 ***
## fe.102     -0.10685948  0.02661054  -4.01568 5.9363e-05 ***
## fe.103     -0.10377297  0.02673175  -3.88201 0.00010373 ***
## fe.104     -0.09674882  0.02659154  -3.63833 0.00027469 ***
## fe.105     -0.09231261  0.02621846  -3.52090 0.00043048 ***
## fe.106     -0.08917392  0.02558100  -3.48594 0.00049084 ***
## fe.107     -0.07811449  0.02392870  -3.26447 0.00109745 **
## fe.108     -0.02082546  0.02286359  -0.91086 0.36237529
## fe.109     -0.04485275  0.02218745  -2.02154 0.04322970 *
## fe.110     -0.05909096  0.02263084  -2.61108 0.00902842 **
## fe.111     -0.07932768  0.02432752  -3.26082 0.00111167 **
## fe.112     -0.07144802  0.02350738  -3.03939 0.00237186 **
## fe.113     -0.08505697  0.02484408  -3.42363 0.00061841 ***
## fe.114     -0.09034513  0.02580058  -3.50167 0.00046276 ***
## fe.115     -0.09571612  0.02602492  -3.67786 0.00023545 ***
## fe.116     -0.09486270  0.02601958  -3.64582 0.00026682 ***
## fe.117     -0.08852947  0.02553935  -3.46640 0.00052794 ***
## fe.118     -0.09408643  0.02554357  -3.68337 0.00023042 ***
## fe.119     -0.09340345  0.02484120  -3.76002 0.00017010 ***
## fe.120     -0.08414831  0.02324697  -3.61975 0.00029518 ***
## fe.121     -0.06943038  0.02264973  -3.06540 0.00217502 **
## fe.122     -0.04809657  0.02259785  -2.12837 0.03331147 *
## fe.123     -0.01762394  0.02545126  -0.69246 0.48865277
## fe.124     -0.04590827  0.02354873  -1.94950 0.05124142 .
## fe.125     -0.06015963  0.02354141  -2.55548 0.01060711 *
## fe.126     -0.08391095  0.02526263  -3.32154 0.00089587 ***
## fe.127     -0.07356572  0.02441697  -3.01289 0.00258903 **
## fe.128     -0.07807990  0.02514235  -3.10551 0.00190058 **
## fe.129     -0.08406474  0.02463689  -3.41215 0.00064505 ***
## fe.130     -0.08639713  0.02445491  -3.53292 0.00041139 ***
## fe.131     -0.08685821  0.02380544  -3.64867 0.00026388 ***
## fe.132     -0.06610577  0.02289880  -2.88687 0.00389271 **
## fe.133     -0.04939431  0.02356808  -2.09581 0.03610384 *
## fe.134     -0.03457177  0.02218661  -1.55823 0.11918595
## fe.135     -0.05478925  0.02279275  -2.40380 0.01622927 *
## fe.136     -0.06035373  0.02287976  -2.63787 0.00834562 **
## fe.137     -0.06129471  0.02269938  -2.70028 0.00693047 **
## fe.138     -0.05798174  0.02283436  -2.53923 0.01111269 *
## fe.139     -0.04568695  0.02318442  -1.97059 0.04877662 *
```

```
## spei_lag    0.94191511  0.00548691 171.66579 < 2.22e-16 ***
## ndvi_lag   -0.00344450  0.00395246  -0.87148 0.38349521
## smoist_lag -0.00739863  0.00515436  -1.43541 0.15117613
## sp_spei    -0.04201634  0.00721678  -5.82204 5.8499e-09 ***
## sp_ndvi     0.01682692  0.00549154   3.06415 0.00218407 **
## sp_smoist   0.02404245  0.00759167   3.16695 0.00154141 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.375879 on 48644 degrees of freedom
## Number of observations: 48789 Degrees of Freedom: 48644
## SSR: 6872.676626 MSE: 0.141285 Root MSE: 0.375879
## Multiple R-Squared: 0.84074 Adjusted R-Squared: 0.840269
##
##
## SUR estimates for 'eqndvi' (equation 2)
## Model Formula: ndvi ~ 0 + (spei + smoist + fe.1 + fe.2 + fe.3 + fe.4 + fe.5 +
##     fe.6 + fe.7 + fe.8 + fe.9 + fe.10 + fe.11 + fe.12 + fe.13 +
##     fe.14 + fe.15 + fe.16 + fe.17 + fe.18 + fe.19 + fe.20 + fe.21 +
##     fe.22 + fe.23 + fe.24 + fe.25 + fe.26 + fe.27 + fe.28 + fe.29 +
##     fe.30 + fe.31 + fe.32 + fe.33 + fe.34 + fe.35 + fe.36 + fe.37 +
##     fe.38 + fe.39 + fe.40 + fe.41 + fe.42 + fe.43 + fe.44 + fe.45 +
##     fe.46 + fe.47 + fe.48 + fe.49 + fe.50 + fe.51 + fe.52 + fe.53 +
##     fe.54 + fe.55 + fe.56 + fe.57 + fe.58 + fe.59 + fe.60 + fe.61 +
##     fe.62 + fe.63 + fe.64 + fe.65 + fe.66 + fe.67 + fe.68 + fe.69 +
##     fe.70 + fe.71 + fe.72 + fe.73 + fe.74 + fe.75 + fe.76 + fe.77 +
##     fe.78 + fe.79 + fe.80 + fe.81 + fe.82 + fe.83 + fe.84 + fe.85 +
##     fe.86 + fe.87 + fe.88 + fe.89 + fe.90 + fe.91 + fe.92 + fe.93 +
##     fe.94 + fe.95 + fe.96 + fe.97 + fe.98 + fe.99 + fe.100 +
##     fe.101 + fe.102 + fe.103 + fe.104 + fe.105 + fe.106 + fe.107 +
##     fe.108 + fe.109 + fe.110 + fe.111 + fe.112 + fe.113 + fe.114 +
##     fe.115 + fe.116 + fe.117 + fe.118 + fe.119 + fe.120 + fe.121 +
##     fe.122 + fe.123 + fe.124 + fe.125 + fe.126 + fe.127 + fe.128 +
##     fe.129 + fe.130 + fe.131 + fe.132 + fe.133 + fe.134 + fe.135 +
##     fe.136 + fe.137 + fe.138 + fe.139 + spei_lag + ndvi_lag +
##     smoist_lag + sp_spei + sp_ndvi + sp_smoist) - spei - ndvi -
##     smoist
##
##             Estimate Std. Error  t value   Pr(>|t|)
## fe.1       1.2120519  0.0569747 21.27349 < 2.22e-16 ***
## fe.2       1.0573564  0.0555474 19.03522 < 2.22e-16 ***
## fe.3       1.2663599  0.0550890 22.98753 < 2.22e-16 ***
## fe.4       0.8753603  0.0551325 15.87740 < 2.22e-16 ***
## fe.5       1.2647504  0.0547700 23.09202 < 2.22e-16 ***
## fe.6       1.1302447  0.0543893 20.78066 < 2.22e-16 ***
## fe.7       1.1581469  0.0542943 21.33091 < 2.22e-16 ***
## fe.8       1.0843162  0.0543759 19.94112 < 2.22e-16 ***
## fe.9       1.1854510  0.0539368 21.97852 < 2.22e-16 ***
## fe.10      1.0900740  0.0553557 19.69219 < 2.22e-16 ***
```

```
## fe.11       1.0196376  0.0554796 18.37860 < 2.22e-16 ***
## fe.12       0.9001672  0.0540747 16.64672 < 2.22e-16 ***
## fe.13       0.8833363  0.0527854 16.73447 < 2.22e-16 ***
## fe.14       0.9385153  0.0524293 17.90060 < 2.22e-16 ***
## fe.15       0.9467465  0.0541071 17.49763 < 2.22e-16 ***
## fe.16       0.9951563  0.0543015 18.32648 < 2.22e-16 ***
## fe.17       1.1828420  0.0548921 21.54849 < 2.22e-16 ***
## fe.18       1.1214414  0.0559187 20.05485 < 2.22e-16 ***
## fe.19       1.0828807  0.0563936 19.20218 < 2.22e-16 ***
## fe.20       0.9833455  0.0557243 17.64661 < 2.22e-16 ***
## fe.21       1.0206563  0.0543670 18.77345 < 2.22e-16 ***
## fe.22       0.9985327  0.0526650 18.96008 < 2.22e-16 ***
## fe.23       1.0322180  0.0526981 19.58739 < 2.22e-16 ***
## fe.24       1.1546038  0.0542671 21.27630 < 2.22e-16 ***
## fe.25       1.2572113  0.0555047 22.65054 < 2.22e-16 ***
## fe.26       1.2546151  0.0569460 22.03168 < 2.22e-16 ***
## fe.27       1.1008521  0.0563599 19.53256 < 2.22e-16 ***
## fe.28       0.9786703  0.0567391 17.24860 < 2.22e-16 ***
## fe.29       0.9759688  0.0561592 17.37861 < 2.22e-16 ***
## fe.30       0.9275800  0.0536996 17.27350 < 2.22e-16 ***
## fe.31       0.8636603  0.0525443 16.43680 < 2.22e-16 ***
## fe.32       1.1435120  0.0542670 21.07195 < 2.22e-16 ***
## fe.33       1.2018609  0.0562747 21.35704 < 2.22e-16 ***
## fe.34       1.1633356  0.0580668 20.03445 < 2.22e-16 ***
## fe.35       1.0220237  0.0590290 17.31392 < 2.22e-16 ***
## fe.36       1.1125924  0.0580701 19.15947 < 2.22e-16 ***
## fe.37       1.1057843  0.0554102 19.95634 < 2.22e-16 ***
## fe.38       1.0071070  0.0544618 18.49200 < 2.22e-16 ***
## fe.39       1.0735827  0.0528694 20.30633 < 2.22e-16 ***
## fe.40       1.2768473  0.0548771 23.26740 < 2.22e-16 ***
## fe.41       1.3264694  0.0587044 22.59572 < 2.22e-16 ***
## fe.42       1.2875222  0.0600708 21.43341 < 2.22e-16 ***
## fe.43       1.0018790  0.0598354 16.74393 < 2.22e-16 ***
## fe.44       1.0063588  0.0597460 16.84396 < 2.22e-16 ***
## fe.45       1.1099621  0.0570714 19.44866 < 2.22e-16 ***
## fe.46       1.2467130  0.0547796 22.75870 < 2.22e-16 ***
## fe.47       1.2427840  0.0572030 21.72585 < 2.22e-16 ***
## fe.48       1.2051098  0.0603996 19.95228 < 2.22e-16 ***
## fe.49       1.2377734  0.0620064 19.96204 < 2.22e-16 ***
## fe.50       1.2177184  0.0616932 19.73829 < 2.22e-16 ***
## fe.51       1.0027907  0.0607534 16.50591 < 2.22e-16 ***
## fe.52       1.0479241  0.0604876 17.32462 < 2.22e-16 ***
## fe.53       0.6359064  0.0601081 10.57938 < 2.22e-16 ***
## fe.54       1.0492095  0.0596337 17.59425 < 2.22e-16 ***
## fe.55       1.3473167  0.0610542 22.06755 < 2.22e-16 ***
## fe.56       1.1930170  0.0617281 19.32697 < 2.22e-16 ***
## fe.57       1.0879523  0.0629063 17.29480 < 2.22e-16 ***
## fe.58       1.1075361  0.0627078 17.66184 < 2.22e-16 ***
## fe.59       1.0867516  0.0611688 17.76644 < 2.22e-16 ***
```

```
## fe.60       1.0509511  0.0604556 17.38385 < 2.22e-16 ***
## fe.61       0.9150440  0.0579258 15.79684 < 2.22e-16 ***
## fe.62       1.1485239  0.0540385 21.25380 < 2.22e-16 ***
## fe.63       1.1310983  0.0537337 21.05007 < 2.22e-16 ***
## fe.64       1.3061491  0.0599063 21.80320 < 2.22e-16 ***
## fe.65       1.2382424  0.0626647 19.75982 < 2.22e-16 ***
## fe.66       1.2052923  0.0638440 18.87871 < 2.22e-16 ***
## fe.67       1.1632090  0.0632715 18.38441 < 2.22e-16 ***
## fe.68       1.1180571  0.0623166 17.94157 < 2.22e-16 ***
## fe.69       1.0072137  0.0584318 17.23744 < 2.22e-16 ***
## fe.70       0.8800142  0.0579269 15.19179 < 2.22e-16 ***
## fe.71       0.6386129  0.0604971 10.55609 < 2.22e-16 ***
## fe.72       0.6834496  0.0541428 12.62310 < 2.22e-16 ***
## fe.73       0.8518486  0.0565441 15.06521 < 2.22e-16 ***
## fe.74       1.0247893  0.0578849 17.70390 < 2.22e-16 ***
## fe.75       1.1326726  0.0575054 19.69680 < 2.22e-16 ***
## fe.76       0.9910942  0.0580197 17.08203 < 2.22e-16 ***
## fe.77       1.2579619  0.0589011 21.35718 < 2.22e-16 ***
## fe.78       1.3052090  0.0597061 21.86056 < 2.22e-16 ***
## fe.79       1.1630878  0.0637298 18.25031 < 2.22e-16 ***
## fe.80       1.1507491  0.0644418 17.85718 < 2.22e-16 ***
## fe.81       1.2434039  0.0654427 18.99987 < 2.22e-16 ***
## fe.82       1.1855180  0.0626236 18.93087 < 2.22e-16 ***
## fe.83       1.0034460  0.0602859 16.64478 < 2.22e-16 ***
## fe.84       0.5967273  0.0553697 10.77714 < 2.22e-16 ***
## fe.85       1.0108272  0.0581612 17.37976 < 2.22e-16 ***
## fe.86       1.2469102  0.0600156 20.77644 < 2.22e-16 ***
## fe.87       1.2136059  0.0616283 19.69233 < 2.22e-16 ***
## fe.88       1.1672299  0.0621848 18.77035 < 2.22e-16 ***
## fe.89       1.1497458  0.0628152 18.30363 < 2.22e-16 ***
## fe.90       1.1909007  0.0647692 18.38683 < 2.22e-16 ***
## fe.91       1.1937804  0.0649760 18.37264 < 2.22e-16 ***
## fe.92       1.1795183  0.0668945 17.63251 < 2.22e-16 ***
## fe.93       1.0756458  0.0678874 15.84456 < 2.22e-16 ***
## fe.94       1.0858226  0.0645903 16.81093 < 2.22e-16 ***
## fe.95       0.8448379  0.0598904 14.10639 < 2.22e-16 ***
## fe.96       0.5397137  0.0552077  9.77606 < 2.22e-16 ***
## fe.97       0.7938626  0.0587495 13.51266 < 2.22e-16 ***
## fe.98       1.2260813  0.0603080 20.33033 < 2.22e-16 ***
## fe.99       1.1430462  0.0624578 18.30110 < 2.22e-16 ***
## fe.100      1.1593143  0.0634965 18.25792 < 2.22e-16 ***
## fe.101      1.2201937  0.0661989 18.43223 < 2.22e-16 ***
## fe.102      1.2606224  0.0669638 18.82544 < 2.22e-16 ***
## fe.103      1.2750852  0.0672682 18.95524 < 2.22e-16 ***
## fe.104      1.2396389  0.0669057 18.52815 < 2.22e-16 ***
## fe.105      1.1556115  0.0659681 17.51772 < 2.22e-16 ***
## fe.106      1.1217694  0.0642955 17.44710 < 2.22e-16 ***
## fe.107      0.7448362  0.0601211 12.38894 < 2.22e-16 ***
## fe.108      0.6079940  0.0574346 10.58584 < 2.22e-16 ***
```

```
## fe.109      0.5766444  0.0557592 10.34168 < 2.22e-16 ***
## fe.110      0.6695410  0.0568977 11.76744 < 2.22e-16 ***
## fe.111      1.1358543  0.0611818 18.56524 < 2.22e-16 ***
## fe.112      1.1615647  0.0591084 19.65142 < 2.22e-16 ***
## fe.113      1.1005487  0.0624916 17.61116 < 2.22e-16 ***
## fe.114      1.0925548  0.0649083 16.83228 < 2.22e-16 ***
## fe.115      1.0111831  0.0654748 15.44386 < 2.22e-16 ***
## fe.116      1.0604944  0.0654582 16.20110 < 2.22e-16 ***
## fe.117      1.0519830  0.0642522 16.37271 < 2.22e-16 ***
## fe.118      1.1992921  0.0642442 18.66772 < 2.22e-16 ***
## fe.119      1.2040892  0.0624212 19.28975 < 2.22e-16 ***
## fe.120      1.0916963  0.0584469 18.67842 < 2.22e-16 ***
## fe.121      1.0283254  0.0569921 18.04331 < 2.22e-16 ***
## fe.122      1.1576583  0.0567850 20.38670 < 2.22e-16 ***
## fe.123      0.5785537  0.0639908  9.04121 < 2.22e-16 ***
## fe.124      0.7564577  0.0591920 12.77973 < 2.22e-16 ***
## fe.125      0.5793884  0.0591915  9.78837 < 2.22e-16 ***
## fe.126      0.6604226  0.0635289 10.39563 < 2.22e-16 ***
## fe.127      0.7685322  0.0614054 12.51572 < 2.22e-16 ***
## fe.128      1.1888408  0.0631734 18.81869 < 2.22e-16 ***
## fe.129      0.7805692  0.0619057 12.60901 < 2.22e-16 ***
## fe.130      0.9217613  0.0614434 15.00179 < 2.22e-16 ***
## fe.131      1.0431865  0.0598535 17.42899 < 2.22e-16 ***
## fe.132      1.0195162  0.0575432 17.71739 < 2.22e-16 ***
## fe.133      0.8589186  0.0592140 14.50534 < 2.22e-16 ***
## fe.134      0.4172274  0.0557446  7.48462 7.2831e-14 ***
## fe.135      0.4697165  0.0572543  8.20404 2.2204e-16 ***
## fe.136      0.8320678  0.0574599 14.48085 < 2.22e-16 ***
## fe.137      0.7570454  0.0570026 13.28089 < 2.22e-16 ***
## fe.138      0.5982570  0.0573352 10.43438 < 2.22e-16 ***
## fe.139      0.6151329  0.0581944 10.57031 < 2.22e-16 ***
## spei_lag   -0.0100348  0.0137880 -0.72779  0.4667437
## ndvi_lag    0.0416286  0.0099459  4.18550 2.8504e-05 ***
## smoist_lag  0.0347218  0.0129525  2.68071  0.0073491 **
## sp_spei     0.0588650  0.0181155  3.24944  0.0011571 **
## sp_ndvi     0.2895066  0.0138422 20.91478 < 2.22e-16 ***
## sp_smoist  -0.1112368  0.0190841 -5.82878 5.6195e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.936021 on 47564 degrees of freedom
## Number of observations: 47709 Degrees of Freedom: 47564
## SSR: 41672.469808 MSE: 0.876135 Root MSE: 0.936021
## Multiple R-Squared: 0.120492 Adjusted R-Squared: 0.11783
##
##
## SUR estimates for 'eqsmoist' (equation 3)
## Model Formula: smoist ~ 0 + (spei + ndvi + fe.1 + fe.2 + fe.3 + fe.4 + fe.5 +
##     fe.6 + fe.7 + fe.8 + fe.9 + fe.10 + fe.11 + fe.12 + fe.13 +
```

```
##     fe.14 + fe.15 + fe.16 + fe.17 + fe.18 + fe.19 + fe.20 + fe.21 +
##     fe.22 + fe.23 + fe.24 + fe.25 + fe.26 + fe.27 + fe.28 + fe.29 +
##     fe.30 + fe.31 + fe.32 + fe.33 + fe.34 + fe.35 + fe.36 + fe.37 +
##     fe.38 + fe.39 + fe.40 + fe.41 + fe.42 + fe.43 + fe.44 + fe.45 +
##     fe.46 + fe.47 + fe.48 + fe.49 + fe.50 + fe.51 + fe.52 + fe.53 +
##     fe.54 + fe.55 + fe.56 + fe.57 + fe.58 + fe.59 + fe.60 + fe.61 +
##     fe.62 + fe.63 + fe.64 + fe.65 + fe.66 + fe.67 + fe.68 + fe.69 +
##     fe.70 + fe.71 + fe.72 + fe.73 + fe.74 + fe.75 + fe.76 + fe.77 +
##     fe.78 + fe.79 + fe.80 + fe.81 + fe.82 + fe.83 + fe.84 + fe.85 +
##     fe.86 + fe.87 + fe.88 + fe.89 + fe.90 + fe.91 + fe.92 + fe.93 +
##     fe.94 + fe.95 + fe.96 + fe.97 + fe.98 + fe.99 + fe.100 +
##     fe.101 + fe.102 + fe.103 + fe.104 + fe.105 + fe.106 + fe.107 +
##     fe.108 + fe.109 + fe.110 + fe.111 + fe.112 + fe.113 + fe.114 +
##     fe.115 + fe.116 + fe.117 + fe.118 + fe.119 + fe.120 + fe.121 +
##     fe.122 + fe.123 + fe.124 + fe.125 + fe.126 + fe.127 + fe.128 +
##     fe.129 + fe.130 + fe.131 + fe.132 + fe.133 + fe.134 + fe.135 +
##     fe.136 + fe.137 + fe.138 + fe.139 + spei_lag + ndvi_lag +
##     smoist_lag + sp_spei + sp_ndvi + sp_smoist) - spei - ndvi -
##     smoist
##
##               Estimate  Std. Error   t value Pr(>|t|)
## fe.1        0.25570974  0.01822538  14.03042  < 2e-16 ***
## fe.2        0.32663423  0.01777409  18.37699  < 2e-16 ***
## fe.3        0.41264187  0.01762800  23.40832  < 2e-16 ***
## fe.4        0.33970642  0.01766345  19.23217  < 2e-16 ***
## fe.5        0.43800145  0.01754331  24.96687  < 2e-16 ***
## fe.6        0.35724484  0.01741890  20.50904  < 2e-16 ***
## fe.7        0.39475841  0.01739260  22.69692  < 2e-16 ***
## fe.8        0.45192146  0.01743747  25.91669  < 2e-16 ***
## fe.9        0.40682996  0.01727497  23.55026  < 2e-16 ***
## fe.10       0.50900518  0.01774811  28.67940  < 2e-16 ***
## fe.11       0.49774811  0.01778679  27.98414  < 2e-16 ***
## fe.12       0.39350469  0.01733505  22.69995  < 2e-16 ***
## fe.13       0.33386160  0.01690017  19.75492  < 2e-16 ***
## fe.14       0.30394696  0.01680984  18.08149  < 2e-16 ***
## fe.15       0.21282476  0.01737341  12.25003  < 2e-16 ***
## fe.16       0.43158937  0.01738709  24.82240  < 2e-16 ***
## fe.17       0.48199884  0.01757737  27.42156  < 2e-16 ***
## fe.18       0.52091658  0.01792717  29.05738  < 2e-16 ***
## fe.19       0.51832788  0.01807971  28.66903  < 2e-16 ***
## fe.20       0.48548318  0.01786517  27.17485  < 2e-16 ***
## fe.21       0.41843741  0.01743026  24.00638  < 2e-16 ***
## fe.22       0.34084683  0.01688574  20.18548  < 2e-16 ***
## fe.23       0.27821955  0.01692163  16.44165  < 2e-16 ***
## fe.24       0.44648664  0.01739462  25.66809  < 2e-16 ***
## fe.25       0.51160409  0.01776866  28.79249  < 2e-16 ***
## fe.26       0.56989207  0.01823054  31.26029  < 2e-16 ***
## fe.27       0.54796164  0.01804453  30.36719  < 2e-16 ***
## fe.28       0.50844539  0.01818897  27.95349  < 2e-16 ***
```

```
## fe.29       0.47211562  0.01800455  26.22202  < 2e-16 ***
## fe.30       0.40480183  0.01721635  23.51264  < 2e-16 ***
## fe.31       0.32945217  0.01684812  19.55424  < 2e-16 ***
## fe.32       0.45367749  0.01741230  26.05501  < 2e-16 ***
## fe.33       0.53914532  0.01803237  29.89876  < 2e-16 ***
## fe.34       0.61685566  0.01860365  33.15778  < 2e-16 ***
## fe.35       0.61001954  0.01889305  32.28804  < 2e-16 ***
## fe.36       0.58260917  0.01859019  31.33961  < 2e-16 ***
## fe.37       0.48196875  0.01776186  27.13503  < 2e-16 ***
## fe.38       0.41838241  0.01745959  23.96289  < 2e-16 ***
## fe.39       0.35096711  0.01695226  20.70327  < 2e-16 ***
## fe.40       0.48363926  0.01760795  27.46710  < 2e-16 ***
## fe.41       0.61546170  0.01882514  32.69360  < 2e-16 ***
## fe.42       0.66506816  0.01924462  34.55865  < 2e-16 ***
## fe.43       0.68503948  0.01916814  35.73844  < 2e-16 ***
## fe.44       0.63242351  0.01911910  33.07811  < 2e-16 ***
## fe.45       0.53825590  0.01826834  29.46387  < 2e-16 ***
## fe.46       0.46635822  0.01757895  26.52936  < 2e-16 ***
## fe.47       0.54524317  0.01834929  29.71467  < 2e-16 ***
## fe.48       0.63657145  0.01936550  32.87142  < 2e-16 ***
## fe.49       0.68501725  0.01987706  34.46271  < 2e-16 ***
## fe.50       0.70747043  0.01976296  35.79780  < 2e-16 ***
## fe.51       0.69913353  0.01946546  35.91663  < 2e-16 ***
## fe.52       0.64114423  0.01936185  33.11378  < 2e-16 ***
## fe.53       0.53104549  0.01892196  28.06504  < 2e-16 ***
## fe.54       0.57930864  0.01903458  30.43454  < 2e-16 ***
## fe.55       0.65465172  0.01957839  33.43746  < 2e-16 ***
## fe.56       0.67455445  0.01978918  34.08703  < 2e-16 ***
## fe.57       0.70779311  0.02016507  35.09995  < 2e-16 ***
## fe.58       0.71857203  0.02010079  35.74844  < 2e-16 ***
## fe.59       0.70314751  0.01959427  35.88537  < 2e-16 ***
## fe.60       0.65471782  0.01934850  33.83817  < 2e-16 ***
## fe.61       0.56292224  0.01854182  30.35960  < 2e-16 ***
## fe.62       0.37616205  0.01731992  21.71847  < 2e-16 ***
## fe.63       0.39950996  0.01724651  23.16468  < 2e-16 ***
## fe.64       0.65346850  0.01919481  34.04403  < 2e-16 ***
## fe.65       0.71720526  0.02009235  35.69544  < 2e-16 ***
## fe.66       0.72429619  0.02046521  35.39158  < 2e-16 ***
## fe.67       0.73314297  0.02028436  36.14326  < 2e-16 ***
## fe.68       0.70863986  0.01997878  35.46962  < 2e-16 ***
## fe.69       0.63228993  0.01872232  33.77199  < 2e-16 ***
## fe.70       0.58492264  0.01854027  31.54877  < 2e-16 ***
## fe.71       0.52949611  0.01911237  27.70436  < 2e-16 ***
## fe.72       0.40940366  0.01738047  23.55539  < 2e-16 ***
## fe.73       0.54748979  0.01814124  30.17929  < 2e-16 ***
## fe.74       0.59248634  0.01855065  31.93884  < 2e-16 ***
## fe.75       0.58039704  0.01842430  31.50172  < 2e-16 ***
## fe.76       0.62440212  0.01860800  33.55557  < 2e-16 ***
## fe.77       0.67114704  0.01889262  35.52430  < 2e-16 ***
```

```
## fe.78       0.70983576  0.01914892  37.06923  < 2e-16 ***
## fe.79       0.78628394  0.02043129  38.48431  < 2e-16 ***
## fe.80       0.76779729  0.02065041  37.18073  < 2e-16 ***
## fe.81       0.72835511  0.02097794  34.72006  < 2e-16 ***
## fe.82       0.70321674  0.02007709  35.02583  < 2e-16 ***
## fe.83       0.60989259  0.01933536  31.54286  < 2e-16 ***
## fe.84       0.48860964  0.01775177  27.52455  < 2e-16 ***
## fe.85       0.59225085  0.01865885  31.74101  < 2e-16 ***
## fe.86       0.64749977  0.01925083  33.63491  < 2e-16 ***
## fe.87       0.68596365  0.01976624  34.70380  < 2e-16 ***
## fe.88       0.73879063  0.01994049  37.04978  < 2e-16 ***
## fe.89       0.78947154  0.02013804  39.20300  < 2e-16 ***
## fe.90       0.80617283  0.02076548  38.82274  < 2e-16 ***
## fe.91       0.81353815  0.02082986  39.05634  < 2e-16 ***
## fe.92       0.76362634  0.02144272  35.61239  < 2e-16 ***
## fe.93       0.70833361  0.02176024  32.55174  < 2e-16 ***
## fe.94       0.67043467  0.02070740  32.37658  < 2e-16 ***
## fe.95       0.57389141  0.01920659  29.87993  < 2e-16 ***
## fe.96       0.49001664  0.01772410  27.64691  < 2e-16 ***
## fe.97       0.58583901  0.01884925  31.08023  < 2e-16 ***
## fe.98       0.64303802  0.01934924  33.23324  < 2e-16 ***
## fe.99       0.70272544  0.02003255  35.07918  < 2e-16 ***
## fe.100      0.74608927  0.02036107  36.64294  < 2e-16 ***
## fe.101      0.78487513  0.02122339  36.98161  < 2e-16 ***
## fe.102      0.79232721  0.02146543  36.91179  < 2e-16 ***
## fe.103      0.79623657  0.02156321  36.92570  < 2e-16 ***
## fe.104      0.76740489  0.02145032  35.77592  < 2e-16 ***
## fe.105      0.72902179  0.02114934  34.47018  < 2e-16 ***
## fe.106      0.69076460  0.02063660  33.47280  < 2e-16 ***
## fe.107      0.60415465  0.01930411  31.29669  < 2e-16 ***
## fe.108      0.44639346  0.01844503  24.20129  < 2e-16 ***
## fe.109      0.53869297  0.01789908  30.09613  < 2e-16 ***
## fe.110      0.59947332  0.01825626  32.83659  < 2e-16 ***
## fe.111      0.70095682  0.01962459  35.71829  < 2e-16 ***
## fe.112      0.70596970  0.01896322  37.22837  < 2e-16 ***
## fe.113      0.79879983  0.02004107  39.85814  < 2e-16 ***
## fe.114      0.79708154  0.02081243  38.29834  < 2e-16 ***
## fe.115      0.79508778  0.02099336  37.87330  < 2e-16 ***
## fe.116      0.79469041  0.02098912  37.86202  < 2e-16 ***
## fe.117      0.77965837  0.02060168  37.84441  < 2e-16 ***
## fe.118      0.76114827  0.02060549  36.93911  < 2e-16 ***
## fe.119      0.72397267  0.02004009  36.12621  < 2e-16 ***
## fe.120      0.64746420  0.01875330  34.52535  < 2e-16 ***
## fe.121      0.60185764  0.01827050  32.94150  < 2e-16 ***
## fe.122      0.48913056  0.01823026  26.83069  < 2e-16 ***
## fe.123      0.52520655  0.02053143  25.58062  < 2e-16 ***
## fe.124      0.64677946  0.01899700  34.04640  < 2e-16 ***
## fe.125      0.70745885  0.01899072  37.25287  < 2e-16 ***
## fe.126      0.74050153  0.02037904  36.33643  < 2e-16 ***
```

```
## fe.127      0.75699649  0.01969678  38.43249  < 2e-16 ***
## fe.128      0.74561411  0.02028314  36.76029  < 2e-16 ***
## fe.129      0.76356595  0.01987532  38.41779  < 2e-16 ***
## fe.130      0.74716447  0.01972862  37.87211  < 2e-16 ***
## fe.131      0.70897756  0.01920377  36.91867  < 2e-16 ***
## fe.132      0.61333371  0.01847301  33.20161  < 2e-16 ***
## fe.133      0.49802008  0.01901317  26.19343  < 2e-16 ***
## fe.134      0.52475971  0.01789866  29.31838  < 2e-16 ***
## fe.135      0.57090833  0.01838795  31.04795  < 2e-16 ***
## fe.136      0.58133085  0.01845842  31.49407  < 2e-16 ***
## fe.137      0.61702259  0.01831299  33.69317  < 2e-16 ***
## fe.138      0.61982246  0.01842202  33.64574  < 2e-16 ***
## fe.139      0.54661787  0.01870485  29.22333  < 2e-16 ***
## spei_lag    0.00362411  0.00442644   0.81874  0.41294
## ndvi_lag   -0.00247898  0.00318826  -0.77753  0.43685
## smoist_lag  0.95780169  0.00415816 230.34290  < 2e-16 ***
## sp_spei     0.00281464  0.00582238   0.48342  0.62880
## sp_ndvi    -0.08618798  0.00442928 -19.45871  < 2e-16 ***
## sp_smoist  -0.13823932  0.00612426 -22.57241  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.303305 on 48644 degrees of freedom
## Number of observations: 48789 Degrees of Freedom: 48644
## SSR: 4474.94813 MSE: 0.091994 Root MSE: 0.303305
## Multiple R-Squared: 0.906683 Adjusted R-Squared: 0.906407
```

Predictive performance

To evaluate the predictive capabilities of our model, we fit it on the first 300 months of data. We also fit 2 simpler models (VAR and SAR) with which we will compare ours. More details on this are in Section 4.2.

```
# Prepare the dataset containing only the first 300 months
df_short <-
  prepare_spvar_df(test_df, test_df_spei, test_df_smoist, test_df_ndvi, T =
                   301)

spvar_fit_short <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist,
      eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist
    ),
    data = df_short,
    method = "SUR"
  )

var_fit_short <-
  systemfit(
```

```
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_smoist -sp_ndvi,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_smoist -sp_ndvi,
      eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_smoist -sp_ndvi
    ),
    data = df_short,
    method = "SUR"
  )

sar_fit_short <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist -sp_smoist -sp_ndvi -ndvi_lag -smoist_lag,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist -sp_smoist -sp_spei -spei_lag -smoist_lag,
      eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_ndvi -ndvi_lag -spei_lag
    ),
    data = df_short,
    method = "SUR"
  )
```

We now look at the adjusted R^2 of the three models for the SPEI equation.

```
print(paste("The adjusted R Squared of the SpVAR is:",adj_r2(spvar_fit_short$eq[[1]])))

## [1] "The adjusted R Squared of the SpVAR is: 0.82613029802549"

print(paste("The adjusted R Squared of the VAR is:",adj_r2(var_fit_short$eq[[1]])))

## [1] "The adjusted R Squared of the VAR is: 0.822843111752807"

print(paste("The adjusted R Squared of the SAR is:",adj_r2(sar_fit_short$eq[[1]])))

## [1] "The adjusted R Squared of the SAR is: 0.823165740366314"
```

We subset the data as needed for making predictions and set all the missing values to 0 so that they don't impact our predictions.

```
data_pred_short <- spvar_df[((299 * 139) + 1):dim(spvar_df)[1],]

data_pred_short[is.na(data_pred_short)] = 0
```

We now look at predictive performances of the three models using the \(SMAPE\) and the \(RMSE\). We start by comparing the error rates using the three models for predictions up to a year in advance

```
for (i in c(1, 3, 6, 12)) {
  print(paste(i, "steps ahead:"))
  spvar_err <-
    compute_pred_steps(102 - i, i, spvar_fit_short, data_pred_short)
  print(paste("The SMAPE and the RMSE are as follows for the SpVAR:", spvar_err[[2]],spvar_err[[1]]))
  var_err <-
    compute_pred_steps(102 - i, i, var_fit_short, data_pred_short, i+1)
  print(paste("The SMAPE and the RMSE are as follows for the VAR:", var_err[[2]],var_err[[1]] ))
  sar_err <-
    compute_pred_steps(102 - i, i, sar_fit_short, data_pred_short, i)
```

```
  print(paste("The SMAPE and the RMSE are as follows for the SAR:", sar_err[[2]], sar_err[[1]]))
}
```

```
## [1] "1 steps ahead:"
## [1] "The SMAPE and the RMSE are as follows for the SpVAR: 0.519449953439214 0.354568699970673"
## [1] "The SMAPE and the RMSE are as follows for the VAR: 0.544368037695199 0.373058356467052"
## [1] "The SMAPE and the RMSE are as follows for the SAR: 0.53543895506312 0.365162180946665"
## [1] "3 steps ahead:"
## [1] "The SMAPE and the RMSE are as follows for the SpVAR: 0.841867634373804 0.632879764211257"
## [1] "The SMAPE and the RMSE are as follows for the VAR: 0.907297560555501 0.716630402536862"
## [1] "The SMAPE and the RMSE are as follows for the SAR: 0.900163896789106 0.711343337616749"
## [1] "6 steps ahead:"
## [1] "The SMAPE and the RMSE are as follows for the SpVAR: 1.13616085581854 0.865707314827308"
## [1] "The SMAPE and the RMSE are as follows for the VAR: 1.21334638604964 1.08265638615537"
## [1] "The SMAPE and the RMSE are as follows for the SAR: 1.20819845037974 1.09194854260833"
## [1] "12 steps ahead:"
## [1] "The SMAPE and the RMSE are as follows for the SpVAR: 1.43219968494042 1.04396822410734"
## [1] "The SMAPE and the RMSE are as follows for the VAR: 1.46570276462238 1.67487168383349"
## [1] "The SMAPE and the RMSE are as follows for the SAR: 1.47879993729122 1.87327111660701"
```

We check if the differences are significant using the Diebold-Mariano test.

```
for (i in c(1, 3, 6, 12)) {
  spvar_res <-
    compute_pred_steps(102 - i,
                       i,
                       spvar_fit_short,
                       data_pred_short,
                       value = "res")
  sar_res <-
    compute_pred_steps(102 - i,
                       i,
                       sar_fit_short,
                       data_pred_short,
                       i,
                       value = "res")
  var_res <-
    compute_pred_steps(102 - i,
                       i,
                       var_fit_short,
                       data_pred_short,
                       i + 1,
                       value = "res")
  spvar_res <- spvar_res[2:((102 - i) * 44)]
  sar_res <- sar_res[2:((102 - i) * 44)]
  var_res <- var_res[2:((102 - i) * 44)]
  print(dm.test(spvar_res, sar_res, h = i))
  print(dm.test(spvar_res, var_res, h = i))
}
```

```
##
```

```
##  Diebold-Mariano Test
##
## data:  spvar_ressar_res
## DM = -2.4919, Forecast horizon = 1, Loss function power = 2, p-value =
## 0.01274
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_resvar_res
## DM = -5.0626, Forecast horizon = 1, Loss function power = 2, p-value =
## 4.304e-07
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_ressar_res
## DM = -9.128, Forecast horizon = 3, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_resvar_res
## DM = -9.8962, Forecast horizon = 3, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_ressar_res
## DM = -14.681, Forecast horizon = 6, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_resvar_res
## DM = -13.679, Forecast horizon = 6, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
```

```
##
## data:  spvar_ressar_res
## DM = -19.269, Forecast horizon = 12, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_resvar_res
## DM = -19.414, Forecast horizon = 12, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
```

We now fit the models on a shorter data-set (containing only the first 150 months) to see the performance on longer term predictions (60 and 120-months ahead).

```
# Prepare the dataset containing only the first 150 months
df_super_short <-
  prepare_spvar_df(test_df, test_df_spei, test_df_smoist, test_df_ndvi, T =
                   151)


spvar_fit_super_short <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist,
      eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist
      ),
    data = df_super_short,
    method = "SUR"
  )


var_fit_super_short <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_ndvi -sp_smoist,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_ndvi -sp_smoist,
      eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_ndvi -sp_smoist
    ),
    data = df_super_short,
    method = "SUR"
  )

sar_fit_super_short <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . -spei -ndvi -smoist -sp_smoist -sp_ndvi -ndvi_lag -smoist_lag,
      eqndvi = ndvi ~ 0 + . -spei -ndvi -smoist -sp_smoist -sp_spei -spei_lag -smoist_lag,
```

```
    eqsmoist = smoist ~ 0 + . -spei -ndvi -smoist -sp_spei -sp_ndvi -ndvi_lag -spei_lag
  ),
  data = df_super_short,
  method = "SUR"
)
```

We again subset the data as needed and set all the missing values to 0 so that they don't impact our predictions.

```
data_pred_super_short <- spvar_df[((149 * 139) + 1):dim(spvar_df)[1],]

data_pred_super_short[is.na(data_pred_super_short)] = 0
```

Finally, we check the performances on these longer-term predictions and check if the differences between models are significant with the Diebold-Mariano test.

```
for (i in c(60, 120)) {
  print(paste(i, "steps ahead:"))
  spvar_err <-
    compute_pred_steps(252 - i, i, spvar_fit_super_short, data_pred_super_short)
  print(paste(
    "The SMAPE and RMSE are as follows for the SpVar",
    spvar_err[[2]],
    spvar_err[[1]]
  ))
  var_err <-
    compute_pred_steps(252 - i, i, var_fit_super_short, data_pred_super_short, i+1)
  print(paste(
    "The SMAPE and RMSE are as follows for the VAR",
    var_err[[2]],
    var_err[[1]]
  ))
  sar_err <-
    compute_pred_steps(252 - i, i, sar_fit_super_short, data_pred_super_short, i)
  print(paste(
    "The SMAPE and RMSE are as follows for the SAR",
    sar_err[[2]],
    sar_err[[1]]
  ))
  spvar_res <-
    compute_pred_steps(252 - i,
                       i,
                       spvar_fit_super_short,
                       data_pred_super_short,
                       value = "res")
  sar_res <-
    compute_pred_steps(252 - i,
                       i,
                       sar_fit_super_short,
                       data_pred_super_short,
                       i,
                       value = "res")
  var_res <-
```

```r
    compute_pred_steps(252 - i,
                       i,
                       var_fit_super_short,
                       data_pred_super_short,
                       i + 1,
                       value = "res")
  spvar_res <- spvar_res[2:((252 - i) * 139)]
  sar_res <- sar_res[2:((252 - i) * 139)]
  var_res <- var_res[2:((252 - i) * 139)]
  print(dm.test(spvar_res, sar_res), h = i)
  print(dm.test(spvar_res, var_res), h = i)
}
```

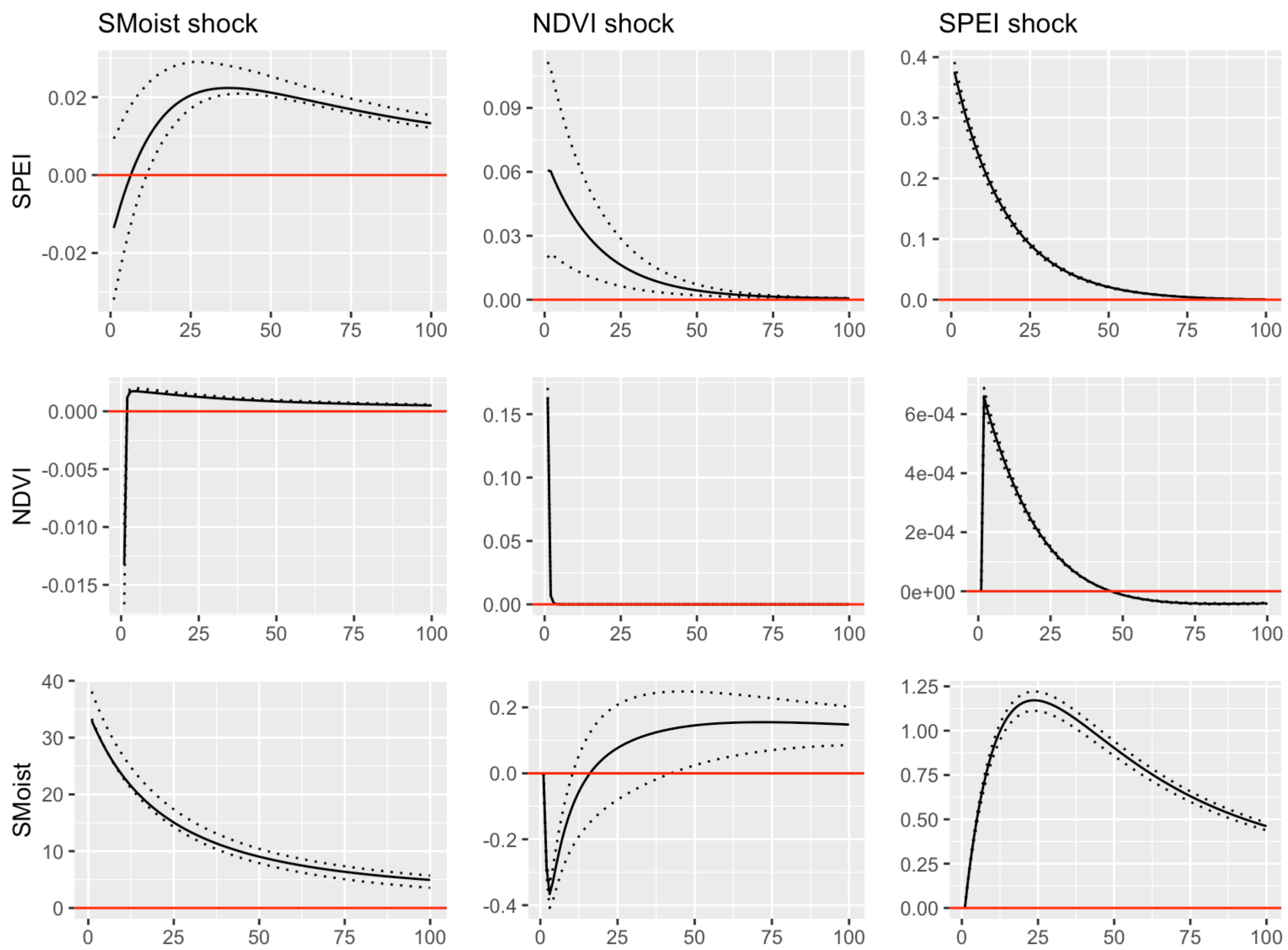
```
## [1] "60 steps ahead:"
## [1] "The SMAPE and RMSE are as follows for the SpVar 1.49133291703587 1.03689778569715"
## [1] "The SMAPE and RMSE are as follows for the VAR 1.52336136792023 1.9454493372135"
## [1] "The SMAPE and RMSE are as follows for the SAR 1.55726484587559 2.67271385831612"
##
##  Diebold-Mariano Test
##
## data:  spvar_ressar_res
## DM = -50.84, Forecast horizon = 1, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_resvar_res
## DM = -49.32, Forecast horizon = 1, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
## [1] "120 steps ahead:"
## [1] "The SMAPE and RMSE are as follows for the SpVar 1.5289622005954 1.08035932682959"
## [1] "The SMAPE and RMSE are as follows for the VAR 1.53116762201427 2.33478155223426"
## [1] "The SMAPE and RMSE are as follows for the SAR 1.57837456210087 3.32292953764326"
##
##  Diebold-Mariano Test
##
## data:  spvar_ressar_res
## DM = -41.619, Forecast horizon = 1, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
##
##
##  Diebold-Mariano Test
##
## data:  spvar_resvar_res
```

```
## DM = -41.788, Forecast horizon = 1, Loss function power = 2, p-value <
## 2.2e-16
## alternative hypothesis: two.sided
```

Impulse Response Functions

We now look at impulse response functions starting from same region IRFs.

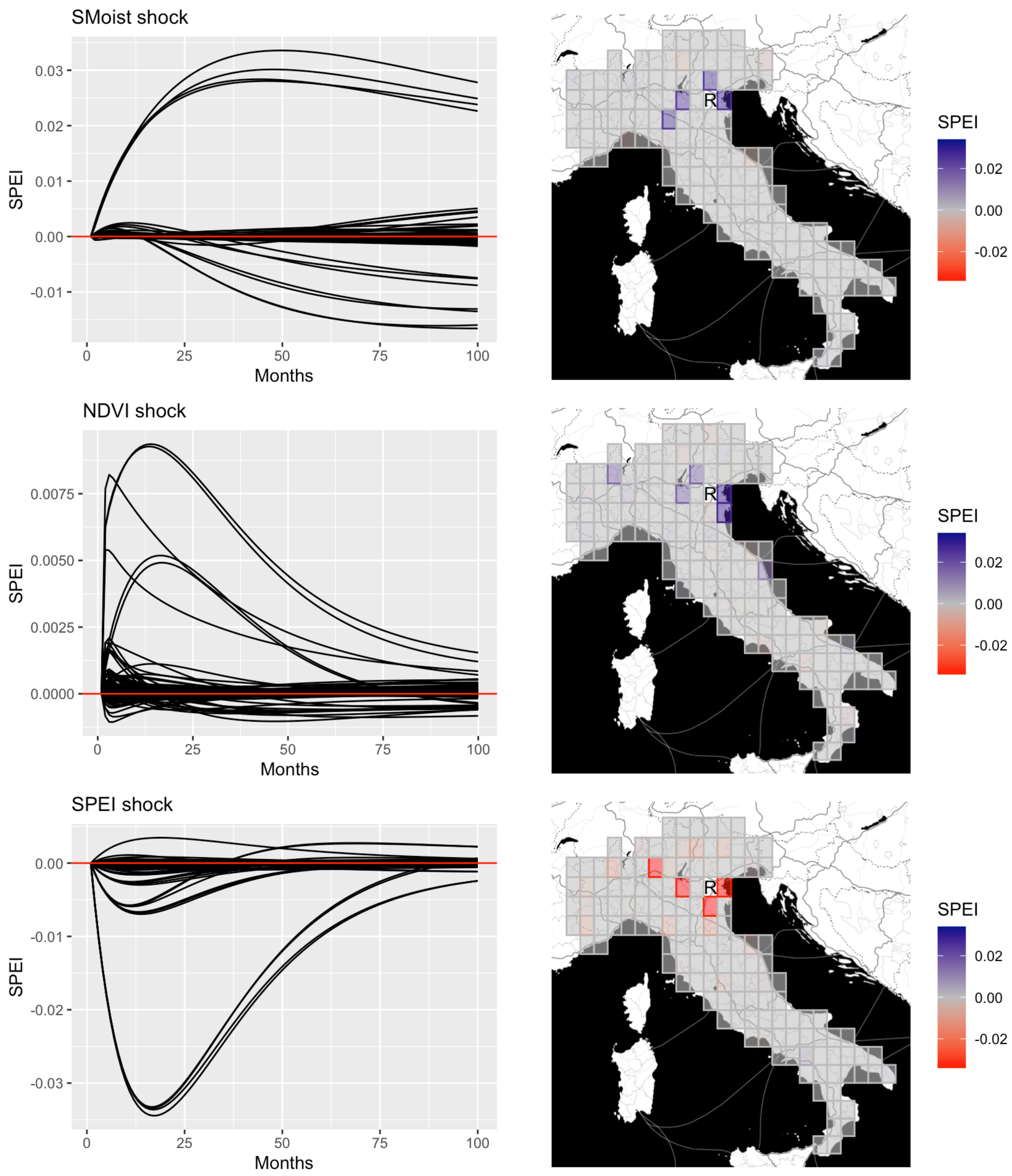
```r
reg_2 <- 106
t_irf <- 100

quantiles <- bootstrap(t_irf, reg_2, spvar_fit, spvar_df, 1000)
multiplot_irf_sameregion(spvar_fit, spvar_df, reg_2, 100, quantiles)
```



We now look at the IRF in neighbouring regions when shocking a region having transitional climate.

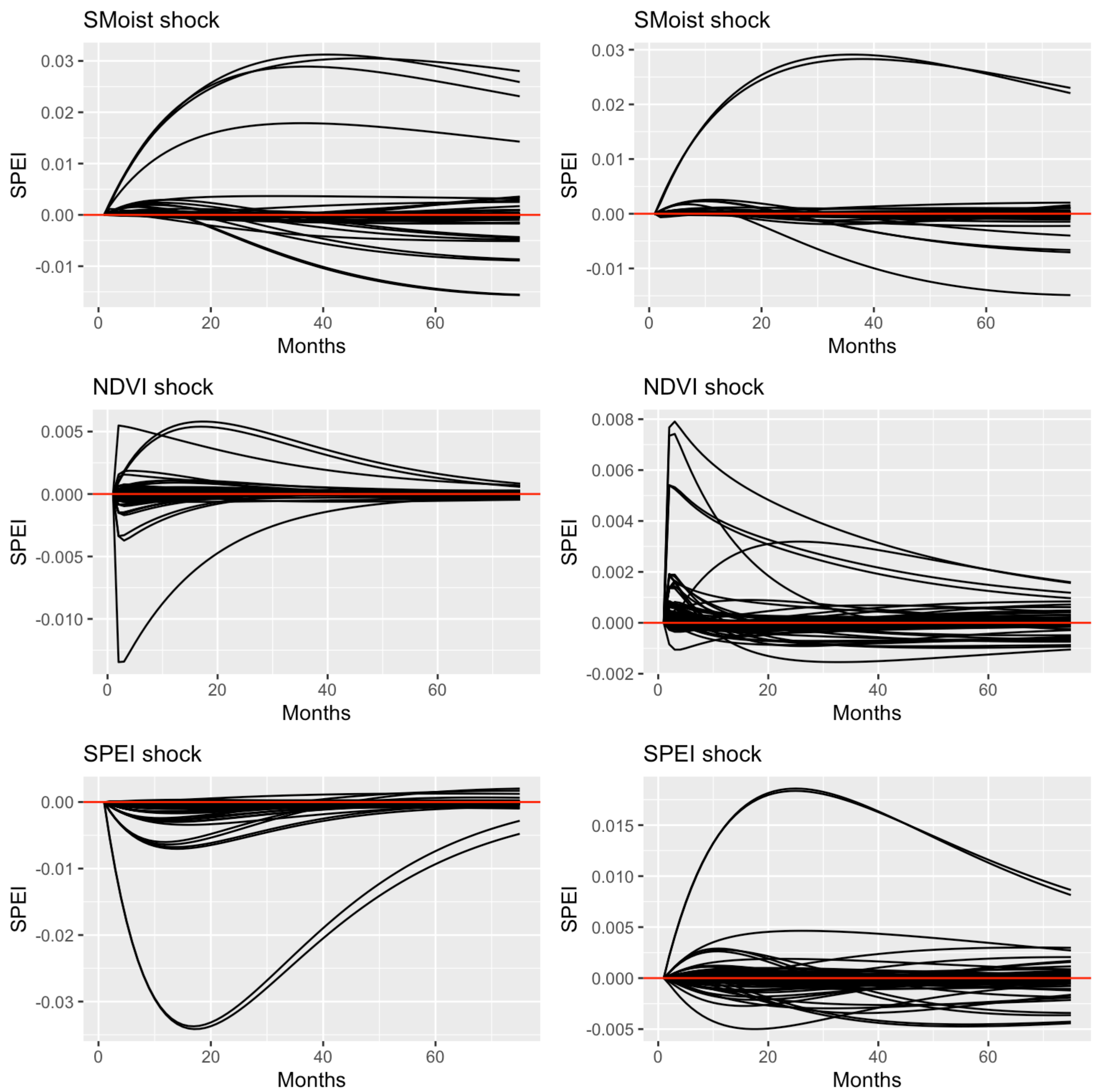
```
plot <- multiplot_irf_all(spvar_fit, 100, reg_2, plot_all = FALSE)

map <- multiplot_irf_map(spvar_fit, index_pairs, test_df, 50, reg_2, 20)
map[[1]] <-
  map[[1]] + theme(plot.margin = unit(c(
    b = 0.3,
    l = 0,
    t = 0.3,
    r = 0
  ), "cm"))
map[[2]] <-
```

```
  map[[2]] + theme(plot.margin = unit(c(
    b = 0.3,
    l = 0,
    t = 0.3,
    r = 0
  ), "cm"))
map[[3]] <-
  map[[3]] + theme(plot.margin = unit(c(
    b = 0.3,
    l = 0,
    t = 0.3,
    r = 0
  ), "cm"))


grid.arrange(
  plot[[1]],
  map[[1]],
  plot[[2]],
  map[[2]],
  plot[[3]],
  map[[3]],
  nrow = 3,
  ncol = 2,
  widths = c(1, 1.1)
)
```


SMoist shock
SPEI
0.03
0.02
0.01
0.00
-0.01
0
25
50
75
100
Months
R
SPEI
0.02
0.00
-0.02
NDVI shock
SPEI
0.0075
0.0050
0.0025
0.0000
0
25
50
75
100
Months
R
SPEI
0.02
0.00
-0.02
SPEI shock
SPEI
0.00
-0.01
-0.02
-0.03
0
25
50
75
100
Months
R
SPEI
0.02
0.00
-0.02

Finally, we plot the IRFs for neighbouring regions when shocking regions with alpine and Mediterranean climate.

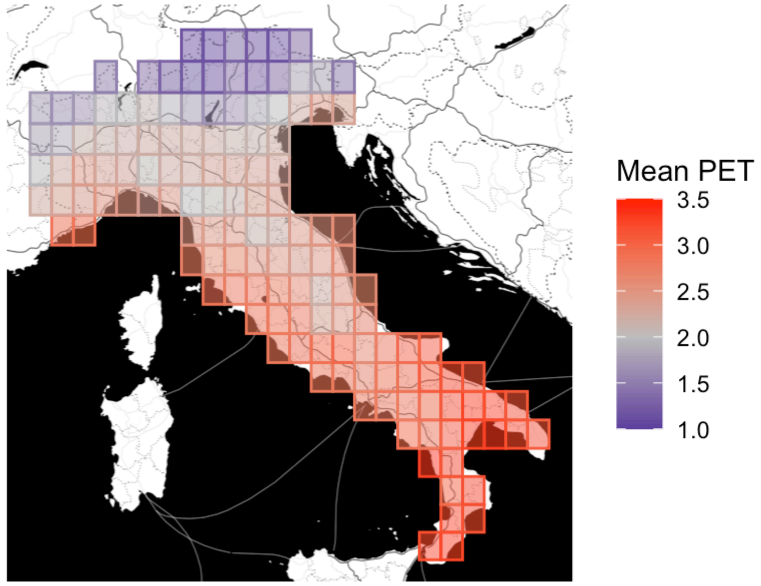
```
reg_1 <- 19
reg_3 <- 137

plot_1 <- multiplot_irf_all(spvar_fit, 75, reg_1, plot_all = FALSE)
plot_2 <- multiplot_irf_all(spvar_fit, 75, reg_3, plot_all = FALSE)

grid.arrange(plot_1[[1]], plot_2[[1]],   plot_1[[2]], plot_2[[2]],
             plot_1[[3]], plot_2[[3]],
             ncol = 2)
```

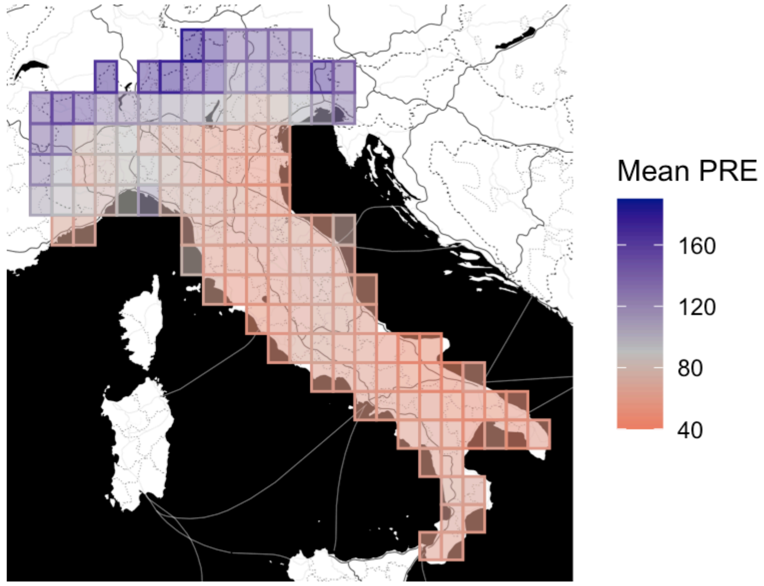

SMoist shock
SMoist shock
NDVI shock
NDVI shock
SPEI shock
SPEI shock
SPEI
Months
0
20
40
60
0.03
0.02
0.01
0.00
-0.01
0.005
0.000
-0.005
-0.010
0.008
0.006
0.004
0.002
-0.002
-0.02
-0.03
0.015
0.010

Other plots

The following plots the mean monthly precipitations and the average PET for each region.

```
# Prepare a dataframe for plotting
plot_df <-
  create_plot_df(1,
                 reg_latitudes ,
                 reg_long_min,
                 reg_long_max,
                 lats,
                 lons,
                 SPEI,
                 NDVI,
                 SMoist)

# Compute the mean precipitation and PET and store them in the dataframe
for (i in 1:139) {
  lon_ind <- as.numeric(strsplit(index_pairs[i], ",")[[1]][2])
  lat_ind <- as.numeric(strsplit(index_pairs[i], ",")[[1]][1])
  plot_df[plot_df$lat_ind == lat_ind &
            plot_df$lon_ind == lon_ind, 5] <-
    mean(PET[lon_ind, lat_ind, 6:402])
  plot_df[plot_df$lat_ind == lat_ind &
            plot_df$lon_ind == lon_ind, 6] <-
    mean(PRE[lon_ind, lat_ind, 6:402])
}

#Duplicate the dataframe
plot_df_2 <- plot_df

#Rename a column for plotting
colnames(plot_df_2)[5] <- "weights"
colnames(plot_df)[6] <- "weights"

plot_1 <-
  create_plot(plot_df_2,
              var = "Mean PET", scale_fixed = TRUE)

plot_2 <-
  create_plot(plot_df,
              var = "Mean PRE", scale_fixed = TRUE)


grid.arrange(plot_1, plot_2, ncol = 2)
```

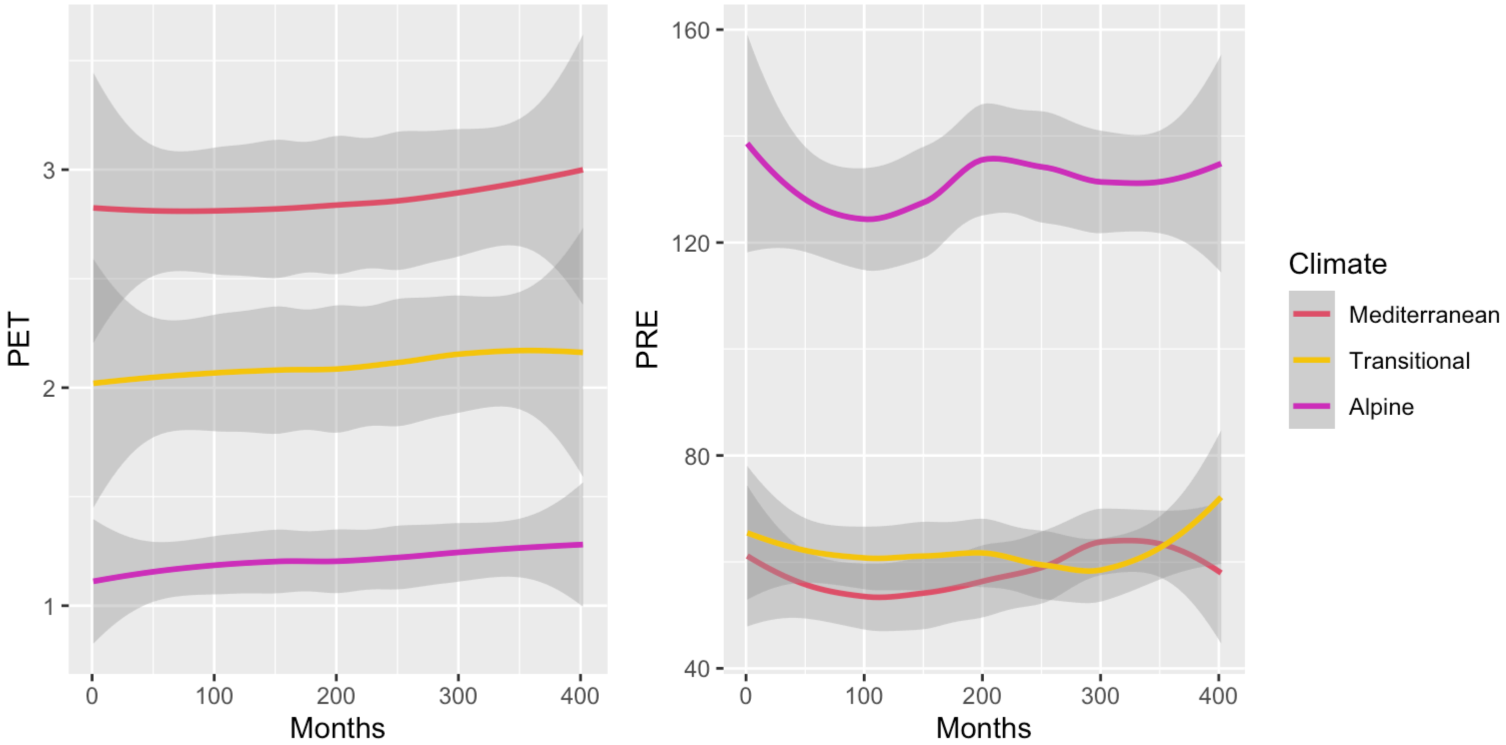




The following plots the evolution of PET and PRE (monthly precipitations) in certain regions (one for each climatic area) to highlight the differences between them.

```
# Prepare a dataframe with the data
test_df_pet <- create_test_df(reg_latitudes,
                              reg_long_min,
                              reg_long_max,
                              lats,
                              lons,
                              PET,
                              PRE,
                              SPEI)

# Rename relevant columns
names(test_df_pet)[6] <- "PET"
names(test_df_pet)[7] <- "PRE"

plot_1_pre <-
  ggplot(data = test_df_pet[test_df_pet$index == index_pairs[reg_1] |
                              test_df_pet$index == index_pairs[reg_2] |
                              test_df_pet$index == index_pairs[reg_3], ]) +
  geom_smooth(aes(x = month, y = PET, color = index)) +
  guides(colour = "none") +
  scale_color_manual(
    name = "Climate",
    labels = c("Mediterranean", "Transitional", "Alpine"),
    values = c("274,384",  "271,384", "262,392")
  ) + labs(x = "Months")


plot_2_pre <-
  ggplot(data = test_df_pet[test_df_pet$index == index_pairs[reg_1] |
```

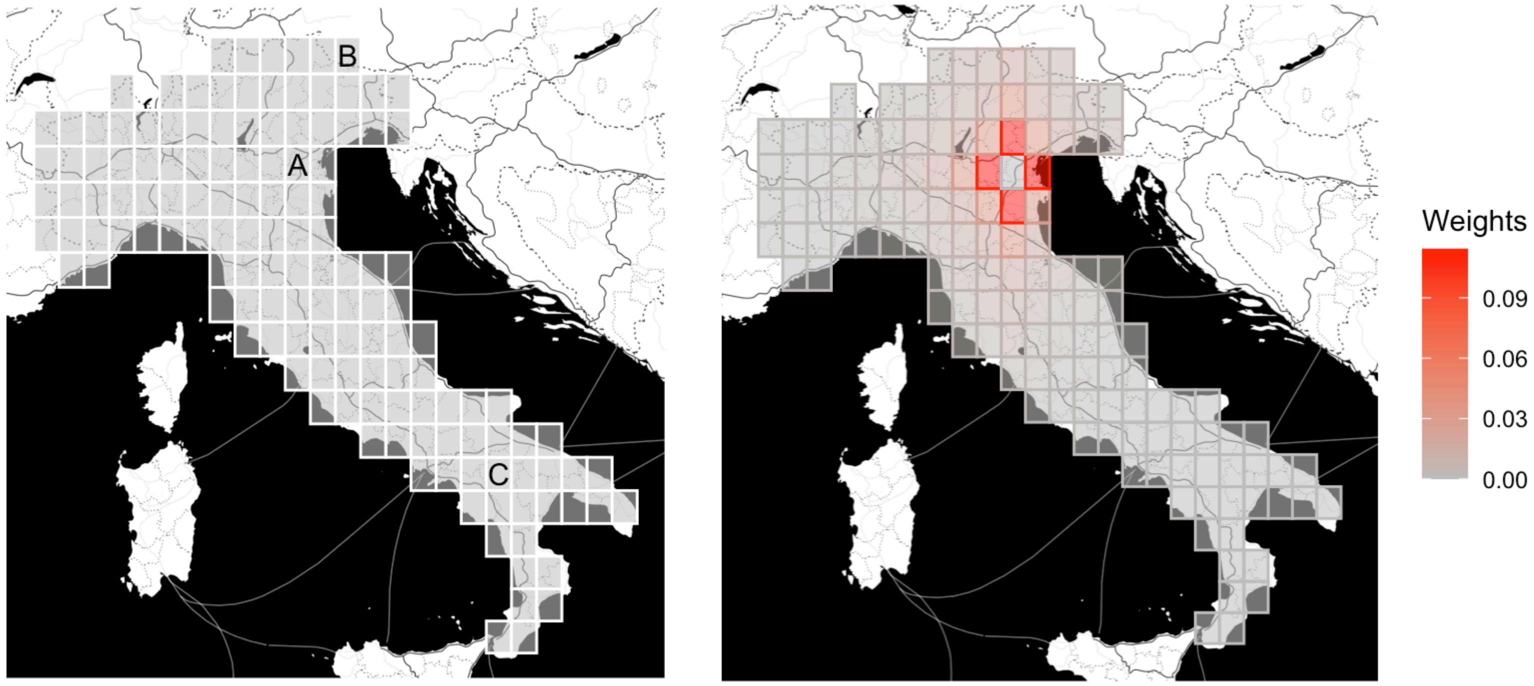
```r
                              test_df_pet$index == index_pairs[reg_2] |
                              test_df_pet$index == index_pairs[reg_3],]) +
  geom_smooth(aes(x = month, y = PRE, color = index)) + scale_color_manual(
    name = "Climate",
    labels = c("Mediterranean", "Transitional", "Alpine"),
    values = c("274,384", "271,384", "262,392")
  ) + labs(x = "Months")

grid.arrange(plot_1_pre, plot_2_pre, ncol = 2, widths = c(0.9, 1.3))
```



The following plots the region under consideration (Continental Italy) and how we divide it to exemplify some concepts of spatial analysis together with the weighting scheme we use.

```r
# Prepare a plot showcasing the region under consideration
plot_df<-create_plot_df(1,reg_latitudes, reg_long_min, reg_long_max, lats, lons, SPEI, SMoist, NDVI)
reg_plot <- qmplot(
  y = lat_plot,
  x = lon_plot,
  data = plot_df,
  geom = "blank",
  maptype = "toner-background",
  zoom = 7
) +
  geom_rect(
    aes(
      ymin = lat_min,
```

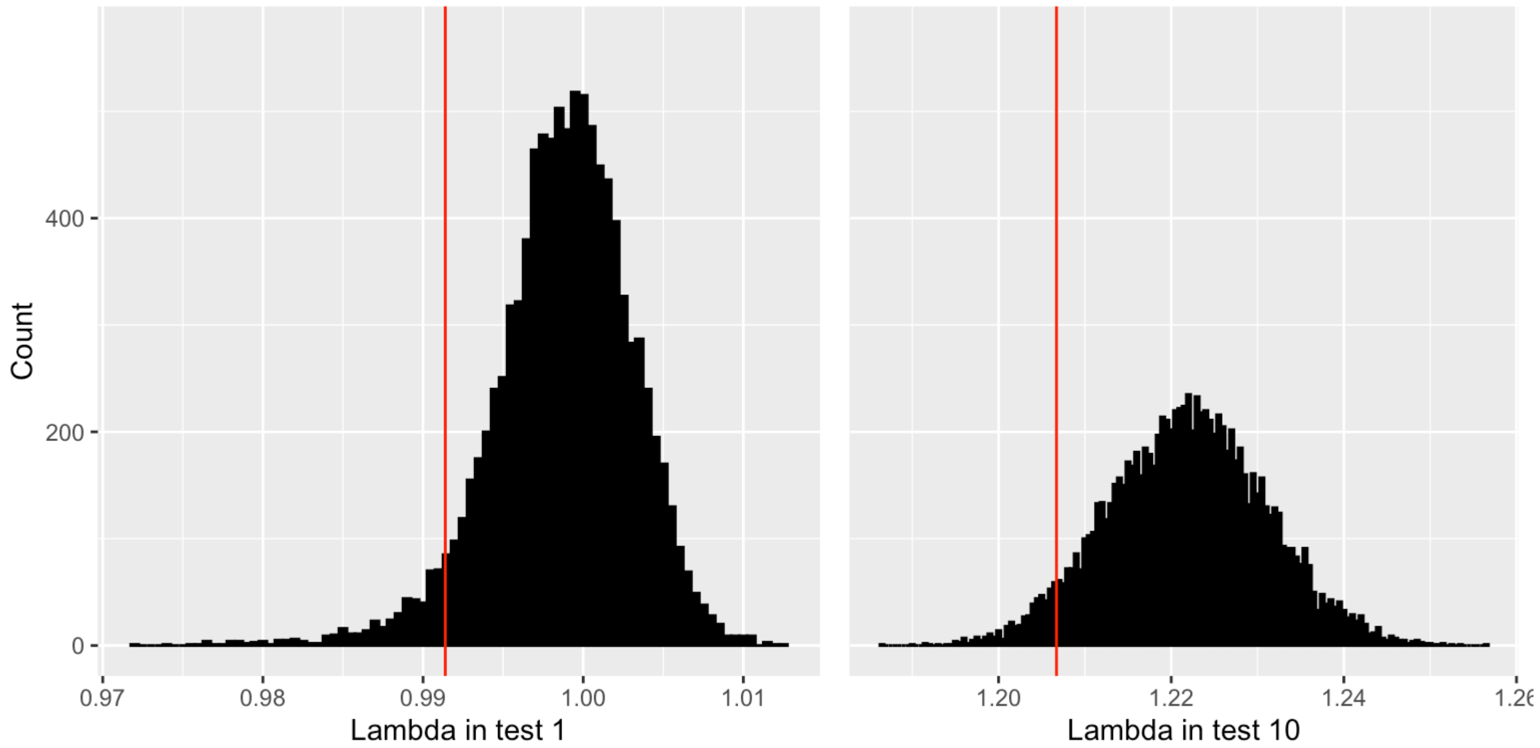
```r
      ymax = lat_max,
      xmin = lon_min,
      xmax = lon_max,
      alpha = 0.01,
      color = "white",
      fill = "grey"
    )
  ) + scale_alpha_continuous(guide = "none") +
  scale_color_discrete(guide = "none", type = "white") +
  scale_fill_discrete(guide = "none", type = "grey") +
  annotate(
    "text",
    x = 12.75,
    y = 46.75,
    label = "B",
    size = 4,
    color = "black"
  ) +
  annotate(
    "text",
    x = 11.75,
    y = 45.25,
    label = "A",
    size = 4,
    color = "black"
  ) +
  annotate(
    "text",
    x = 15.75,
    y = 40.75,
    label = "C",
    size = 4,
    color = "black"
  )

# Prepare a plot showcasing the weighting scheme
plot_weights <- create_plot_df_weights(weights[reg_2,])
weight_plot <- create_plot(plot_weights, var = "Weights")
reg_plot <-
  reg_plot + theme(plot.margin = unit(c(0, 0.75, 0, 0), "cm"))
grid.arrange(reg_plot,
             weight_plot,
             ncol = 2,
             widths = c(0.87, 1))
```



The following plots the distribution of \(\lambda\) under the first and tenth unit root test

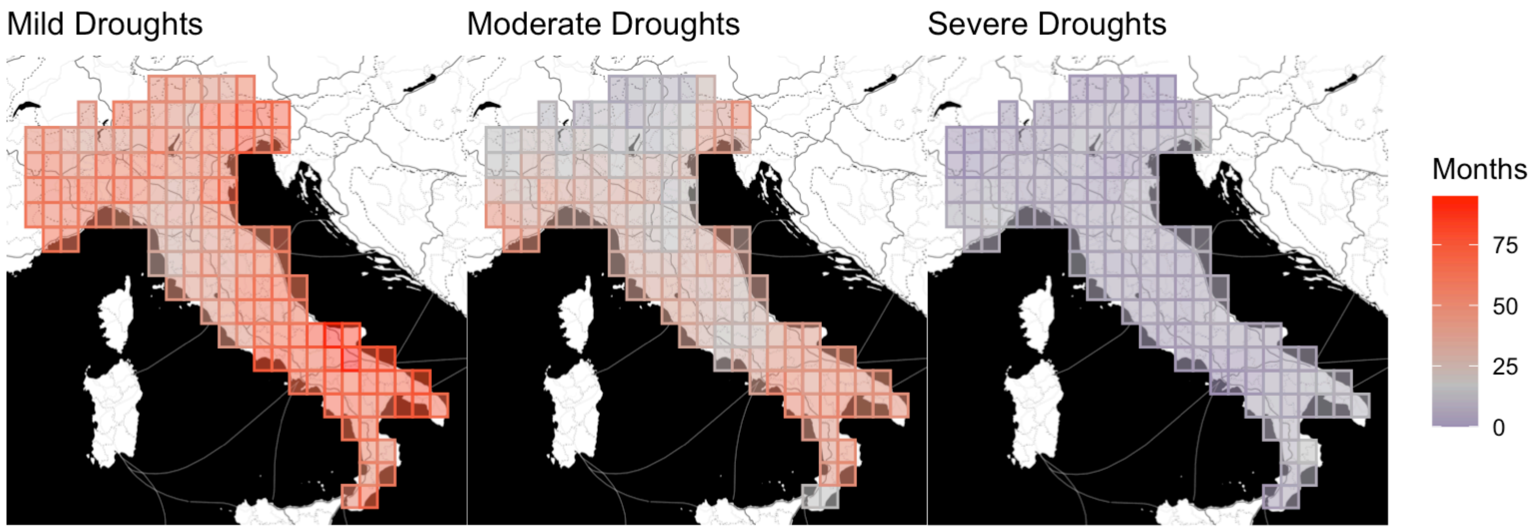
```
lambda_first_test <- tibble(x = lambda_mc[1:10000, ])
lambda_tenth_test <- tibble(x = lambda_mc[90001:100000, ])

hist1 <-
  ggplot(data = lambda_first_test) + geom_histogram(
    aes(x = x),
    binwidth = 0.0005,
    color = "black",
    fill = "black"
  ) + labs(x = "Lambda in test 1", y = "Count") +
  geom_vline(aes(xintercept = quantile(x, probs = 0.05)), color = "red") +
  scale_y_continuous(limits = c(0, 570))

hist2 <-
  ggplot(data = lambda_tenth_test) + geom_histogram(
    aes(x = x),
    binwidth = 0.0005,
    color = "black",
    fill = "black"
  ) + labs(x = "Lambda in test 10", y = NULL) +
  scale_y_continuous(limits = c(0, 570), guide = "none") +
  geom_vline(aes(xintercept = quantile(x, probs = 0.05)), color = "red")
```

```r
grid.arrange(hist1, hist2, ncol = 2, widths = c(1.2, 1))
```



The following plots the number of droughts of each type in the 402 months under consideration.

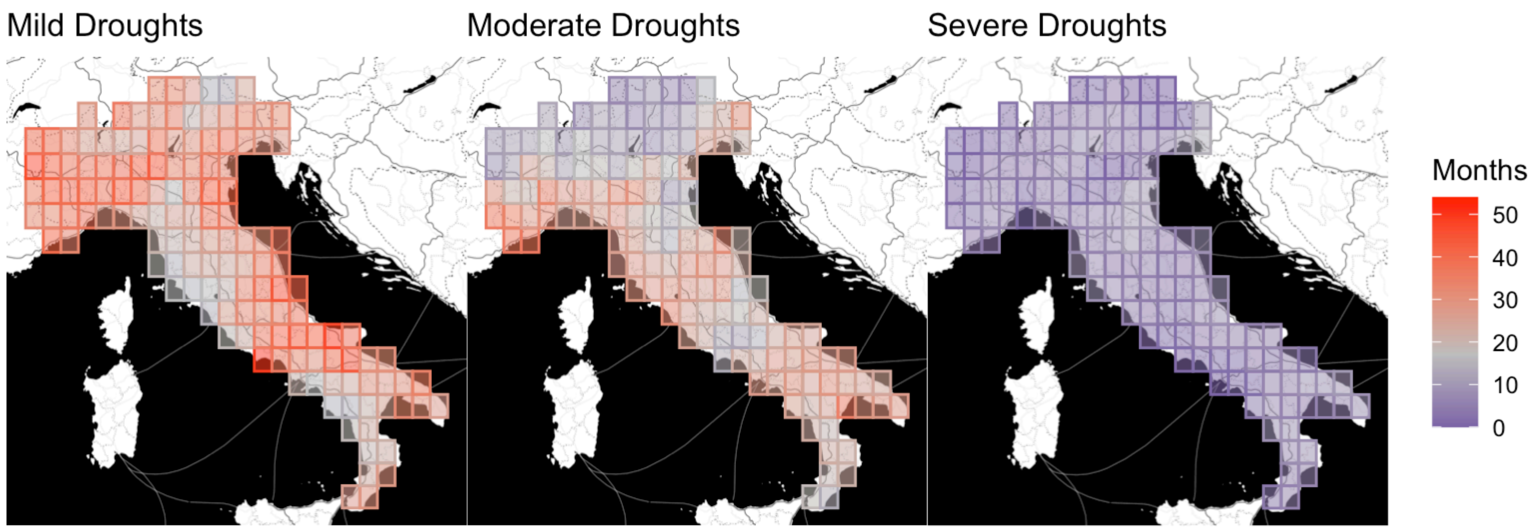
```r
plots_droughts <-
  long_term_prediction_plot(
    test_df_spei[1:402,], plot="initial"
  )
```

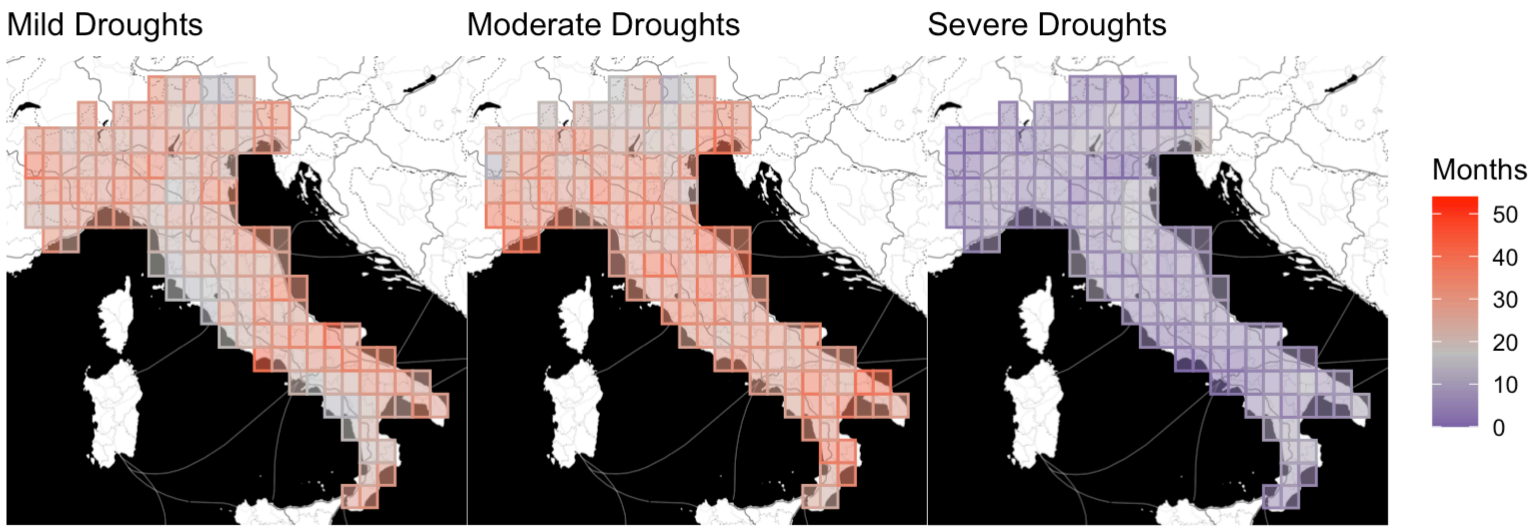


The following plots the predicted number of droughts in the second 201 periods of the period under consideration (below) against the actual number of them in the data (above). Note that we use a model fit only on the first 201 months.

```
# Prepare the data
test_df_201<- test_df[((139 * 199) + 1):(139 * 201), ]
test_df_201$month <- c(rep(1, times = 139), rep(2, times = 139))
test_df_201_spei <- test_df_spei[200:201, ]
test_df_201_ndvi <- test_df_ndvi[200:201, ]
test_df_201_smoist <- test_df_smoist[200:201, ]

# Fit a model on the first 201 periods
spvar_df_201 <- spvar_df[1:(201 * 139), ]
spvar_fit_201 <-
  systemfit(
    c(
      eqspei = spei ~ 0 + . - spei - ndvi - smoist,
      eqndvi = ndvi ~ 0 + . - spei - ndvi - smoist,
      eqsmoist = smoist ~ 0 + . - spei - ndvi - smoist
    ),
    data = spvar_df_201,
    method = "SUR"
  )

# Create the plot
plots_droughts_201 <-
  long_term_prediction_plot(
    test_df_spei[201:402, ],
    test_df_201,
    test_df_201_spei,
    test_df_201_smoist,
    test_df_201_ndvi,
```

```
    t_pred = 201,
    spvar_fit = spvar_fit_201
  )
```

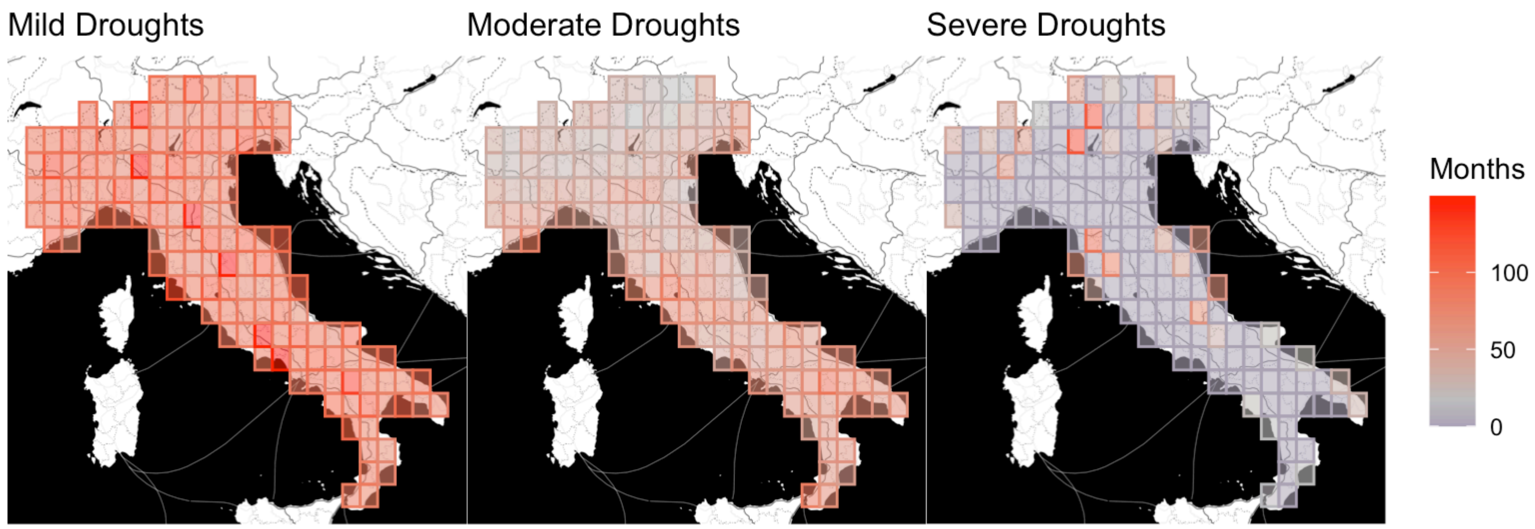




The following plots the predicted number of droughts in the next 402 periods (the 402 months after those covered by the dataset, i.e. starting from January 2014).

```
# Create a table with the data of the last 2 years
test_df_last_year <- create_test_df(
  reg_latitudes,
  reg_long_min,
  reg_long_max,
  lats,
  lons,
```

```
  SPEI_last_year,
  SMoist_last_year,
  NDVI_last_year
)

# Subset the data
test_df_last_year <- test_df_last_year[1:278, ]

#Create a separate df for each variable
test_df_last_year_spei <-
  data.frame(split(test_df_last_year$SPEI, as.factor(test_df_last_year$index)))
test_df_last_year_ndvi <-
  data.frame(split(test_df_last_year$NDVI, as.factor(test_df_last_year$index)))
test_df_last_year_smoist <-
  data.frame(split(
    test_df_last_year$SMoist,
    as.factor(test_df_last_year$index)
  ))


plots_droughts_future <-
  long_term_prediction_plot(
    test_df_spei[201:402, ],
    test_df_last_year,
    test_df_last_year_spei,
    test_df_last_year_smoist,
    test_df_last_year_ndvi,
    t_pred = 402,
    spvar_fit = spvar_fit,
    plot = "future"
  )
```


Mild Droughts
Moderate Droughts
Severe Droughts
Months
100
50
0