How to produce precipitation-weighted annual average isoscapes in IsoriX?

The IsoriX core team

2017-06-16

Welcome to **IsoriX**, in this vignette we present the steps required for you to build an annual average isoscape weighted by the amounth of monthly precipitation.

Before starting

Please read the vignette ***Workflow***, if you haven't done so. You can access it by simply typing:

```
vignette("Workflow", package="IsoriX")
```

Note that this vignette, like the one introducing the workflow takes a lot of time to run. It has thus not been compiled by CRAN but by us. We have included the sources of this vignette as a text file which you can find in your computer. This file is in the folder `IsoriX/doc` within the folder where you usually install your packages (the latter being usually the one that shows up when you type `.libPaths()[1]` in R).

Again, due to constraints on how big this document could be, we had to reduce a lot the resolution of the figures. If you run it on your computer you will see that in fact figures produced by **IsoriX** are great!

Before starting, don't forget to load our package:

```
library(IsoriX)
```

Step 1 - Select the isoscape data

Start by selecting the GNIP data needed for you to build an isoscape. In this example, we will consider all the data available in `GNIPdata` within an extent of latitude and longitude that covers roughly Europe. The difference with what we did in the vignette ***Workflow*** is that here the function `queryGNIP` is called with the argument `split.by = "month"`, we lead to data aggregated across years (as before), but not across months.

```
## load all the GNIP data
data(GNIPdata)

## select the relevant data
GNIPdataEU12 <- queryGNIP(data = GNIPdata,
                          split.by = "month",
                          long.min = -30,
                          long.max = 60,
                          lat.min = 30,
                          lat.max = 70)
```

The dataset we created contains many more rows than the one used during the ***Workflow*** as we now have up to twelve different rows per location (i.e. one per month if records are available for all twelve months) instead of the single one.

```
## [1] 3325    8
```

stationID	isoscape.value	var.isoscape.value	n.isoscape.value	lat	long	elev	month
1003500	-58.84000	201.65673	16	54.52	9.54	43	1
1003500	-59.50438	321.59208	16	54.52	9.54	43	2
1003500	-62.11812	387.91506	16	54.52	9.54	43	3
1003500	-52.11375	221.34881	16	54.52	9.54	43	4
1003500	-44.27063	101.23491	16	54.52	9.54	43	5
1003500	-48.82529	103.74713	17	54.52	9.54	43	6
1003500	-44.78176	60.07234	17	54.52	9.54	43	7
1003500	-48.80647	148.66621	17	54.52	9.54	43	8
1003500	-43.71941	250.31979	17	54.52	9.54	43	9
1003500	-49.05118	33.67141	17	54.52	9.54	43	10
1003500	-55.50412	356.77401	17	54.52	9.54	43	11
1003500	-61.47000	177.85881	17	54.52	9.54	43	12
1005500	-67.61333	291.63838	15	54.52	11.06	3	1
1005500	-66.14375	450.59996	16	54.52	11.06	3	2
1005500	-57.75625	141.20129	16	54.52	11.06	3	3

Step 2 - Fit the geostatistical models

We will now fit not one pair of models as during the ***Workflow*** but twelve pairs of models. We indeed want to fit one mean model and one residual dispersion model for each of the twelve months of a year. Each of the twelve pairs of models are technically fitted independently, but to save you the manual labor of calling twelve times the function `isofit`, we have created the function `isomultifit` that does that for you. This latter function also combines all fitted models in one object of class `multiisofit` which other functions will recognize. As `isofit`, `isomultifit` can fit several model structures, but we will restrict the demonstration to a single example.

```
Europefit12 <- isomultifit(iso.data = GNIPdataEU12,
                           split.by = "month",
                           mean.model.fix  = list(elev = TRUE, lat.abs = TRUE),
                           mean.model.rand = list("uncorr" = TRUE),
                           disp.model.rand = list("uncorr" = TRUE))
```

We check that all models are there:

```
names(Europefit12$multi.fits)
```

```
##  [1] "month_1"  "month_2"  "month_3"  "month_4"  "month_5"  "month_6"
##  [7] "month_7"  "month_8"  "month_9"  "month_10" "month_11" "month_12"
```

You could then look at the output of a given model (here, January) by simply typing `Europefit12$multi.fits$month_1`.

Step 3 - Prepare the elevation raster

As for the ***Workflow***, we prepare the elevation raster from the tif file we downloaded (see ***Workflow*** for details):

```
library(raster)
elevationraster <- raster("../vignette_workflow/gmted2010_30mn.tif")

elev <- relevate(elevation.raster = elevationraster,
```

```
                 isofit = Europefit12,
                 aggregation.factor = 10)
```

```
## class       : RasterLayer
## dimensions  : 467, 1014, 473538  (nrow, ncol, ncell)
## resolution  : 0.08333333, 0.08333333  (x, y)
## extent      : -27.34181, 57.15819, 30.04153, 68.95819  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +datum=WGS84 +no_defs +ellps=WGS84 +towgs84=0,0,0
## data source : /private/var/folders/r4/nmf9vqyj7pz968sk2vrtmbvm0000gn/T/Rtmpj2MgiS/raster/r_tmp_2017-0
## names       : gmted2010_30mn
## values      : -412, 4318.32  (min, max)
```

Step 4 - Prepare the precipitation rasters

We now need the rasters containing the average precipitation amount for each month of the eyar and each location from the elevation raster. We start by dowloading such file (mind that the file is ca. 1Gb and takes a while to download):

```
getprecip()
```

```
## the file wc2.0_30s_prec.zip is already present in /media/alex/Data/Dropbox/Boulot/Mes_projets_de_rech
```

```
## [1] the file seems OK (md5sums do match)
```

We then resize the RasterBrick obtained to the size of the elevation raster:

```
precipitations <- prepcipitate(elevation.raster = elev)
```

Step 5 - Build the isoscape

To build the precipitation-weighted annual average isoscapes steming from `Europefit12` we need to use the function `isomultiscape`. This function is a wrapper to `isoscape` handling several models at once. This is also the function to which we need to provide the prepared precipitation data:

```
isoscapes <- isomultiscape(elevation.raster = elev,
                           isofit = Europefit12,
                           weighting = precipitations)
```

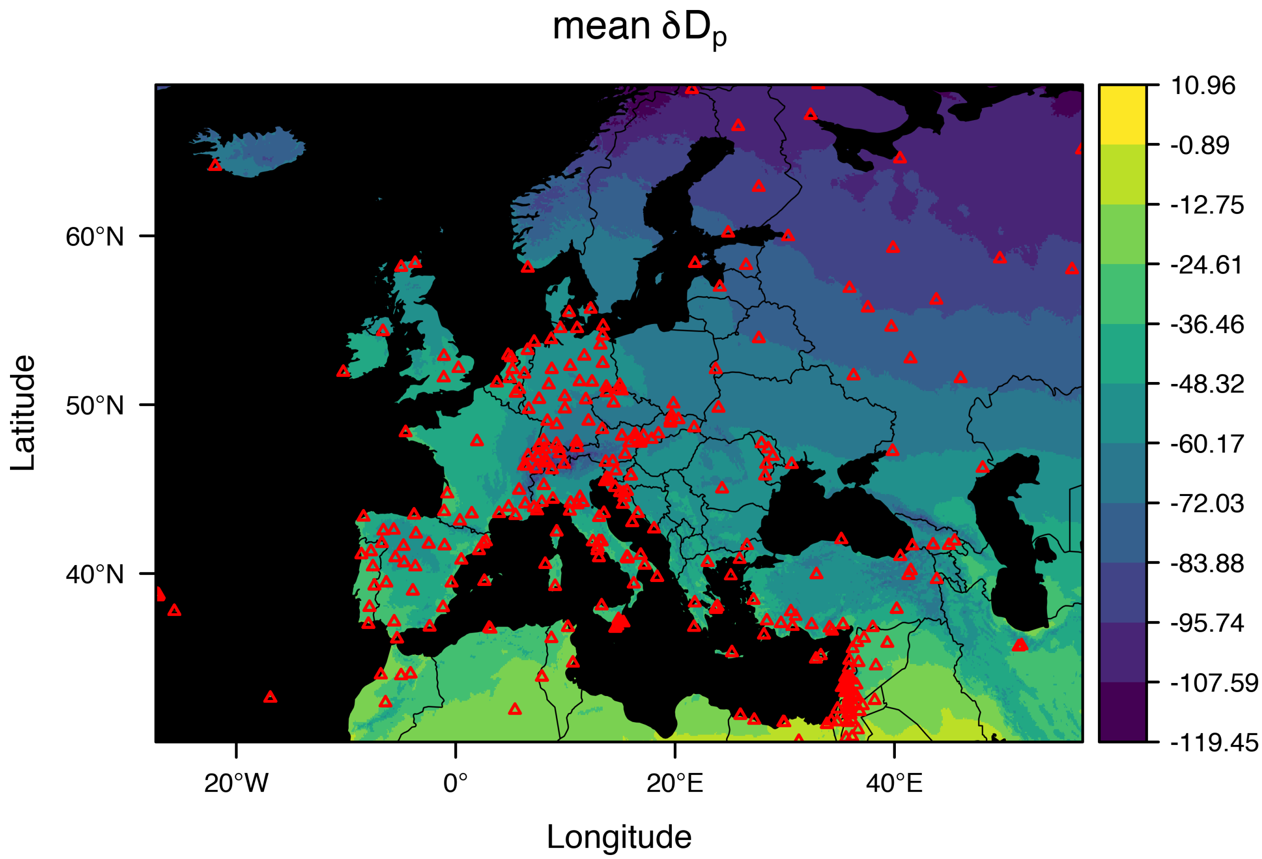
Step 6 - Plotting the isoscapes

We can finally plot the precipitation-weighted annual average isoscape. We first load the objects that are used to decorate the maps:

```
data(countries)
data(oceanmask)
```

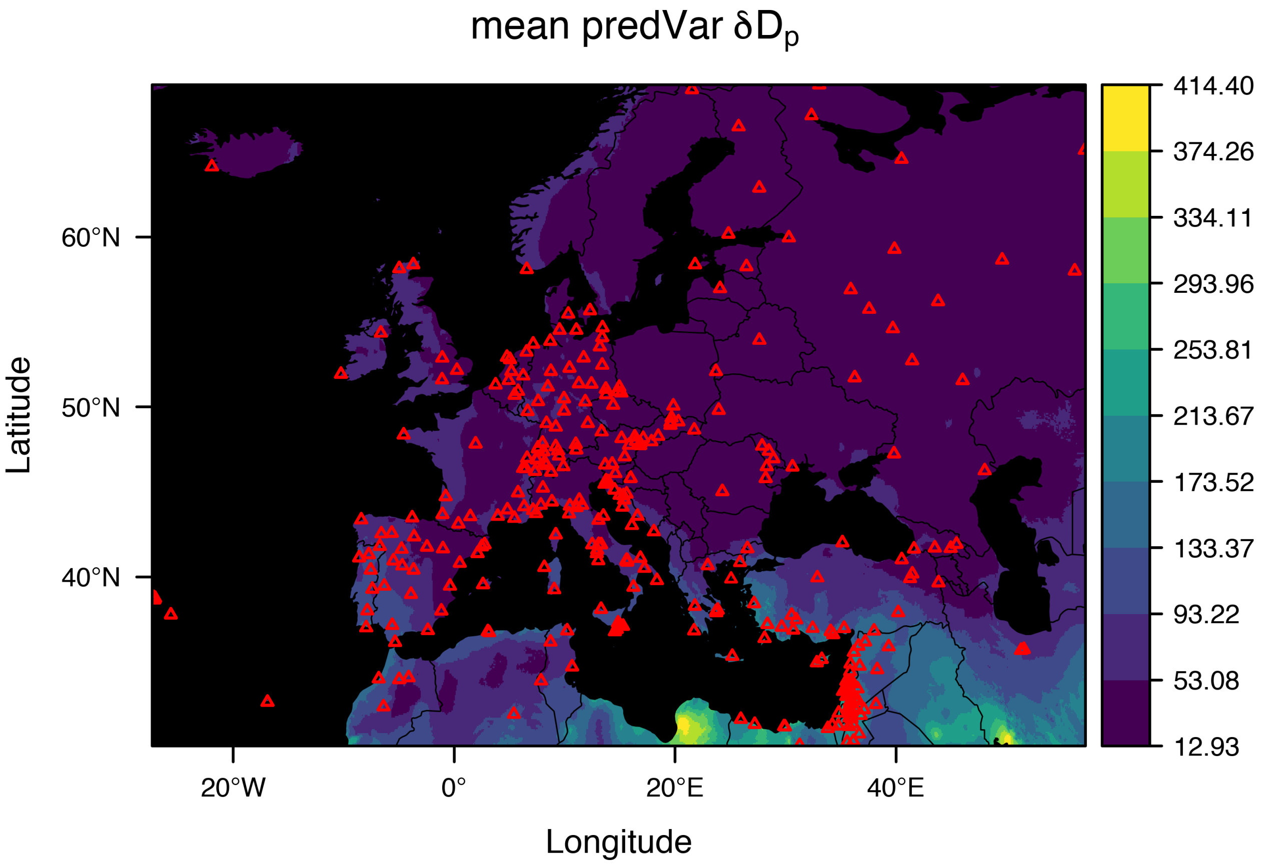
We now produce the plot as we did in the ***Workflow***:

```
plot(x = isoscapes,
     which = "mean",
     borders = list(borders = countries),
     mask = list(mask = oceanmask),
     palette = list(fn = viridisLite::viridis))
```

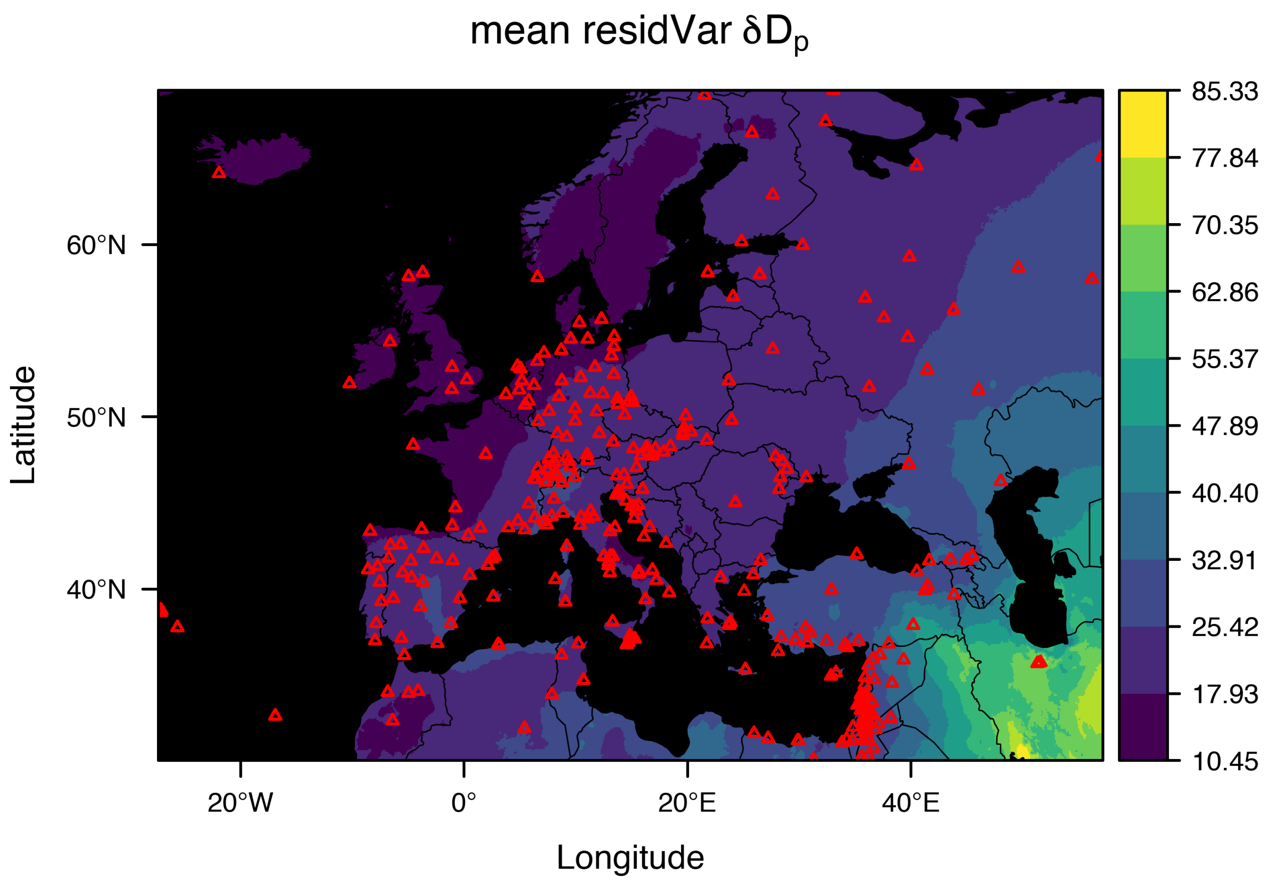


As for any isoscape fitted with **IsoriX** we can also plot the isoscape for the prediction variance, the one for the residual variance, and the one for the response variance:

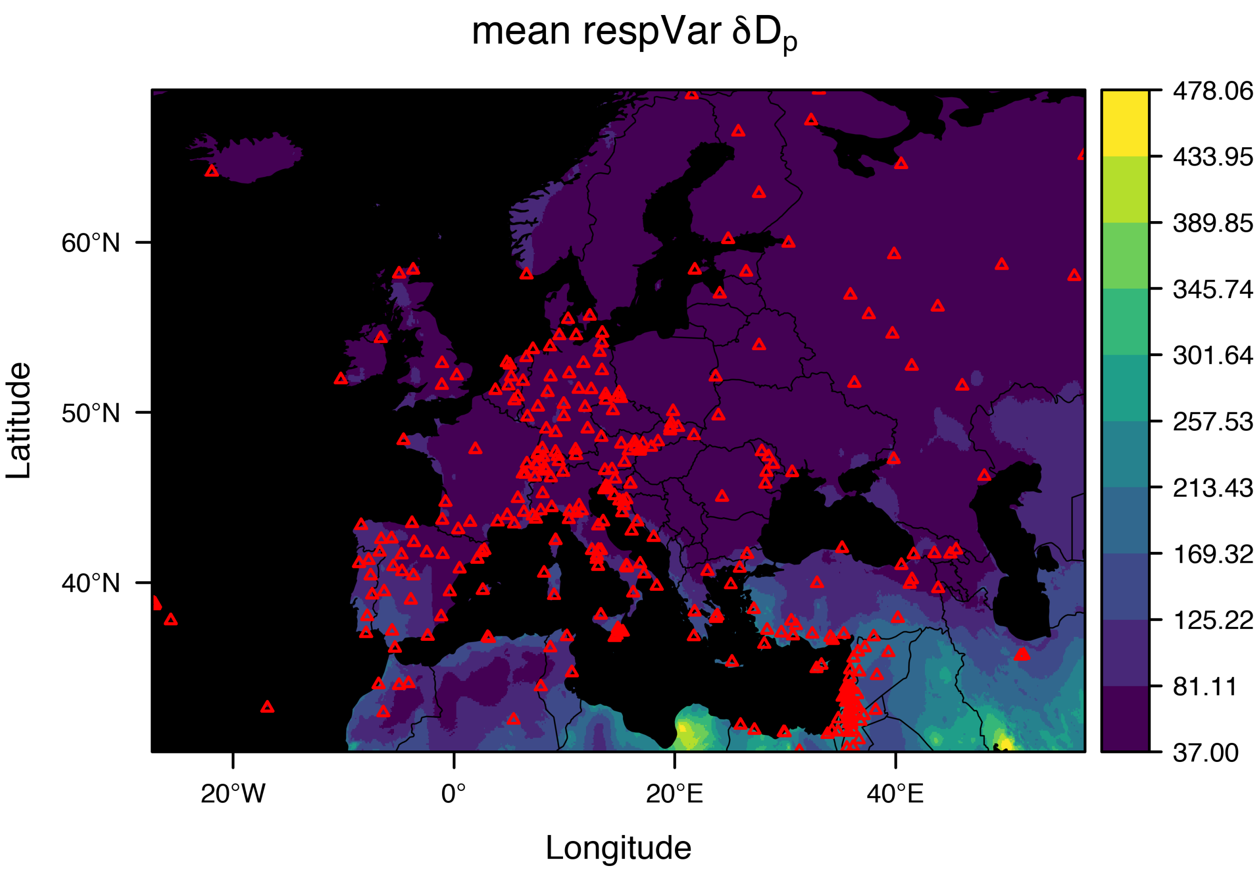
```
plot(x = isoscapes,
     which = "mean.predVar",
     borders = list(borders = countries),
     mask = list(mask = oceanmask),
     palette = list(fn = viridisLite::viridis))
```




```r
plot(x = isoscapes,
     which = "mean.residVar",
     borders = list(borders = countries),
     mask = list(mask = oceanmask),
     palette = list(fn = viridisLite::viridis))
```




```r
plot(x = isoscapes,
     which = "mean.respVar",
     borders = list(borders = countries),
     mask = list(mask = oceanmask),
     palette = list(fn = viridisLite::viridis))
```



Does the isoscapes differ from the one not accounting for precitation?

Above two differences were introduced compared to the simple approached followed during the ***Workflow***. First models were fitted by month, second they were weighted by predicipitation amounts before the aggregation. You can simply study the influence of such additional steps by comparing the isoscapes produced by different workflows.

We will here compare the isoscape for point predictions produced during the ***Workflow*** to the one produced above. To do so, we assume here that you have run the ***Workflow*** and stored the isoscape as the following R object called `isoscape`:

```
isoscape
```

```
## ### stack containing the isoscapeclass       : RasterStack
## dimensions  : 467, 1014, 473538, 8  (nrow, ncol, ncell, nlayers)
## resolution  : 0.08333333, 0.08333333  (x, y)
## extent      : -27.34181, 57.15819, 30.04153, 68.95819  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=longlat +datum=WGS84 +ellps=WGS84 +towgs84=0,0,0
## names       :         mean, mean.predVar, mean.residVar, mean.respVar,         disp, disp.predVar, di
## min values  :  -122.457713,   386.879669,     66.769762,   478.986872,    66.769762,     1.544308,
## max values  :    0.3543686,  542.6219411,  1583.0291725, 2084.0358481, 1583.0291725,    2.3636798,
##
##
## ### first 5 locations of the dataset     coordinates values
## 1   (6.57, 58.1)  -9999
## 2 (21.53, 68.68)  -9999
## 3 (25.75, 66.49)  -9999
```

```
## 4 (27.62, 62.89)  -9999
## 5 (24.83, 60.18)  -9999
## NULL
```

We now compute the difference between the two isoscapes (mind that the two isoscapes must have same resolution and extent to do that directly, which is the case here):

```
isoscape.diff <- isoscape  ## We create a new object of class isoscape
isoscape.diff$isoscape <- isoscape$isoscape - isoscapes$isoscape  ## We replace the isoscape by
                                                                  ## the difference in isoscapes
```

You could plot the point predictions as before, but we choose to add one small step to adjust the title:

```
plotdiff <- plot(x = isoscape.diff,
                 which = "mean",
                 borders = list(borders = countries),
                 mask = list(mask = oceanmask),
                 palette = list(range = c(-13, 13), step = 2,
                                n.labels = Inf, fn = viridisLite::viridis),
                 plot = FALSE)  ## We create the plot
plotdiff$main <- "Difference in" ~ delta * D[p] ~ "(simple - weighted by monthly precipitation amounts)"
plotdiff  ## We display the plot
```