Chapter 6: Prediction of seasonal runoff in ungauged basins - a EU example

Alberto Viglione

1 Introduction

This Tutorial has been developed by Alberto Viglione to illustrate the regional prediction of seasonal runoff in ungauged basins (see, Weingartner et al., 2013). The idea is to predict the seasonality of runoff (measured by the Pardé coefficients) in Europe based on a dataset of 763 catchments in EU, where we have some basic information on runoff, precipitation, temperature and solar radiation.

First of all load the library:

```
library(PUBexamples)
```

Then the data:

```
help(data4chapter5and6)
```

```
data(data4chapter5and6)
head(CatchmentsEU, 15)
```

```
      code                            station                              river     lon     lat   elev   area country
1  6114500                   PONTE DE IONCAIS                   PONTE DE IONCAIS -7.5200 40.6100 330.00  606.0      PT
2  6115500                    M.DA GAMITINHA                     M.DA GAMITINHA -8.4000 38.0700  28.00 2721.0      PT
3  6118010                SAINT-JEAN-BREVELAY                              CLAIE -2.7033 47.8248  99.98  137.0      FR
4  6118015                  LOYAT (PONT D129)                               YVEL -2.3688 47.9938  88.26  315.0      FR
5  6118020      GRAND-FOUGERAY (LA BERNADAISE)                   RUISSEAUX D ARON -1.6908 47.7122  68.00  118.0      FR
6  6118030          PAIMPONT (PONT DU SECRET)                                AFF -2.1438 47.9816 119.01   30.2      FR
7  6118060                             GUENIN                               EVEL -2.9752 47.8999  95.16  316.0      FR
8  6118070                BANNALEC (PONT MEYA)                           STER-GOZ -3.7522 47.9068  85.08   69.7      FR
9  6118150                          TREZILIDE                            GUILLEC -4.0770 48.6150  91.10   43.0      FR
10 6118165                                RAI                              RISLE  0.5806 48.7479 255.45  149.0      FR
11 6118175                         PLOUGONVEN                             JARLOT -3.8005 48.5656  73.86   44.0      FR
12 6118205               SAINT-OUEN-LA-ROUERIE                           LOISANCE -1.4363 48.4270 103.34   81.5      FR
13 6118210          MONFORT-SUR-MEU (LABBAYE)                                MEU -1.9448 48.1275  83.50  468.0      FR
14 6119010            BERENX (PONT DE BERENX)            BERENX (PONT DE BERENX) -0.8537 43.5078 100.44 2575.0      FR
15 6119020 OLORON-SAINTE-MARIE (OLORON-STE-CROIX) OLORON-SAINTE-MARIE (OLORON-STE-CROIX) -0.5971 43.1877 277.79  488.0      FR
```

```
head(meanQmon, 15)    # mean monthly discharge (m3/s)
```

```
          Jan   Feb   Mar    Apr    May    Jun   Jul   Aug   Sep   Oct   Nov   Dec
6114500 18.26 21.75 16.11  10.46   6.67   3.18  1.03  0.29  0.32  2.06  8.13 15.31
6115500 16.95 22.18 17.91   6.02   3.05   1.03  0.38  0.53  1.49  2.03  6.50 14.21
6118010  3.65  3.43  2.65   2.10   1.60   1.02  0.54  0.33  0.34  0.66  1.39  2.56
6118015  5.48  5.18  3.78   2.42   1.86   0.88  0.42  0.17  0.20  0.71  1.64  3.52
6118020  1.87  1.65  1.15   0.73   0.55   0.18  0.07  0.02  0.04  0.21  0.52  1.27
6118030  0.57  0.54  0.43   0.31   0.25   0.09  0.02  0.01  0.02  0.07  0.18  0.37
6118060  8.25  7.92  5.74   3.74   2.49   1.32  0.57  0.30  0.36  1.05  2.58  5.64
6118070  3.12  3.13  2.28   1.74   1.24   0.83  0.58  0.41  0.40  0.67  1.39  2.33
6118150  1.23  1.28  1.04   0.82   0.62   0.43  0.33  0.27  0.27  0.37  0.58  0.94
6118165  2.34  2.32  1.94   1.42   1.14   0.83  0.73  0.61  0.61  0.90  1.36  2.36
6118175  1.29  1.40  1.14   0.91   0.66   0.44  0.32  0.25  0.24  0.34  0.59  0.98
6118205  1.26  1.29  1.12   0.90   0.77   0.56  0.44  0.36  0.38  0.55  0.78  1.03
6118210  7.35  7.30  5.25   3.39   2.86   1.28  0.82  0.34  0.36  0.99  2.12  4.67
6119010 90.01 89.76 85.34 101.70 123.10 115.00 68.80 42.01 40.87 55.45 76.49 94.18
6119020 17.49 17.53 19.64  27.65  38.77  29.70 14.37  6.96  8.42 13.83 19.95 19.71
```

```
head(meanPmon, 15)    # mean monthly catchment precipitation (mm/d)
```

```
         Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
6114500 4.15 3.73 2.52 2.64 2.18 1.20 0.35 0.32 1.31 3.41 3.93 4.05
6115500 2.50 2.40 1.54 1.61 1.10 0.44 0.04 0.09 0.67 2.15 2.64 3.05
6118010 2.99 2.58 2.14 1.87 1.99 1.39 1.24 1.33 2.00 2.79 3.16 3.09
6118015 2.71 2.34 1.98 1.77 1.97 1.42 1.26 1.31 1.92 2.60 2.96 2.81
6118020 2.34 2.05 1.78 1.61 1.96 1.41 1.29 1.26 1.84 2.34 2.58 2.38
6118030 2.56 2.22 1.89 1.72 1.97 1.44 1.26 1.30 1.88 2.51 2.86 2.66
6118060 3.16 2.71 2.24 1.95 2.02 1.42 1.27 1.39 2.08 2.90 3.30 3.26
6118070 3.96 3.37 2.74 2.38 2.23 1.60 1.47 1.67 2.40 3.45 3.88 4.03
6118150 4.07 3.46 2.77 2.39 2.19 1.61 1.51 1.73   NA 3.46   NA 4.14
6118165 2.32 2.06 1.93 1.68 1.99 1.69 1.68 1.46   NA 2.31   NA 2.51
6118175 3.72 3.17 2.56 2.18 2.05 1.54 1.42 1.61 2.26 3.23   NA 3.86
6118205 2.33 2.05 1.78 1.61 1.92 1.45 1.31 1.27 1.82 2.34   NA 2.45
6118210 2.52 2.18 1.87 1.70 1.97 1.43 1.27 1.29 1.87 2.48   NA 2.61
6119010 3.52 3.23 2.91 3.72 3.58 2.85 1.82 2.19 2.77 3.54 4.00 3.69
6119020 3.79 3.41 3.00 3.82 3.70 2.99 1.83 2.17 2.89 3.93 4.36 3.98
```

```
head(meanTmon, 15)    # mean monthly catchment temperature (deg C)
```

```
        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
6114500  6.16  7.16  9.29 10.84 14.05 18.40 21.45 21.45 18.80 14.18  9.49  6.76
6115500 10.66 11.47 13.26 14.71 17.44 20.92 23.60 23.86 21.97 18.40 14.18 11.38
6118010  6.23  6.43  8.07  9.90 12.94 15.78 17.74 17.94 16.17 13.16  9.42  6.97
6118015  5.79  6.09  7.87  9.78 12.94 15.87 17.90 18.02 16.06 12.88  8.96  6.45
6118020  5.40  5.94  7.99 10.11 13.45 16.58 18.67 18.66 16.35 12.85  8.57  5.91
6118030  5.63  5.97  7.81  9.77 12.97 15.94 17.99 18.08 16.03 12.78  8.79  6.26
6118060  6.36  6.49  8.04  9.82 12.80 15.59 17.52 17.77 16.08 13.17  9.53  7.13
6118070  6.63  6.61  7.89  9.41 12.16 14.78 16.65 16.96 15.54 12.91  9.58  7.45
6118150  6.82  6.80  8.02  9.45 12.12 14.68 16.54 16.83 15.43 12.87  9.61  7.60
6118165  3.64  4.25  6.51  8.84 12.29 15.35 17.50 17.57 15.02 11.46  6.99  4.17
6118175  6.58  6.55  7.85  9.38 12.14 14.75 16.66 16.95 15.49 12.86  9.52  7.39
6118205  5.17  5.61  7.60  9.66 12.95 15.99 18.08 18.14 15.92 12.53  8.38  5.75
6118210  5.62  5.98  7.84  9.81 13.03 16.01 18.06 18.14 16.07 12.79  8.79  6.24
6119010  2.59  3.29  5.22  7.15 10.89 14.55 17.36 17.35 14.55 10.66  5.82  3.19
6119020  0.02  0.58  2.37  4.15  8.05 12.00 15.20 15.19 12.08  7.95  3.17  0.66
```

```
head(meanSImon, 15)  # mean monthly catchment SI ratio
```

```
        Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   Nov   Dec
6114500 0.213 0.240 0.273 0.306 0.333 0.346 0.340 0.317 0.285 0.252 0.221 0.206
6115500 0.221 0.247 0.273 0.303 0.323 0.338 0.332 0.312 0.285 0.257 0.230 0.213
6118010 0.200 0.232 0.269 0.312 0.344 0.362 0.354 0.324 0.287 0.248 0.210 0.191
6118015 0.197 0.231 0.269 0.312 0.347 0.368 0.357 0.325 0.286 0.246 0.210 0.185
6118020 0.201 0.232 0.269 0.312 0.343 0.361 0.353 0.324 0.287 0.249 0.211 0.191
6118030 0.202 0.232 0.269 0.312 0.345 0.363 0.354 0.324 0.286 0.248 0.211 0.186
6118060 0.200 0.231 0.269 0.312 0.345 0.365 0.355 0.324 0.286 0.248 0.211 0.186
6118070 0.200 0.231 0.268 0.312 0.347 0.366 0.356 0.324 0.286 0.247 0.210 0.185
6118150 0.194 0.231 0.269 0.314 0.350 0.369 0.360 0.324 0.287 0.247 0.204 0.185
6118165 0.195 0.231 0.270 0.314 0.349 0.368 0.359 0.326 0.288 0.245 0.205 0.185
6118175 0.193 0.230 0.269 0.314 0.350 0.369 0.360 0.325 0.287 0.247 0.206 0.183
6118205 0.195 0.232 0.269 0.314 0.349 0.367 0.359 0.325 0.287 0.246 0.207 0.185
6118210 0.195 0.231 0.269 0.313 0.346 0.367 0.358 0.325 0.287 0.246 0.211 0.185
6119010 0.196 0.234 0.273 0.314 0.345 0.362 0.354 0.326 0.290 0.248 0.207 0.184
6119020 0.189 0.230 0.274 0.318 0.350 0.367 0.359 0.331 0.292 0.246 0.202 0.176
```

In the exercise on annual runoff prediction in ungauged basins, some useful variables where derived from the data:

```
MAQ <- apply(meanQmon, 1, mean)  # m3/s
A <- CatchmentsEU$area  # km2
MAR <- 365.25*24*3.6*MAQ/A  # mm/yr
MAP <- 365.25*apply(meanPmon, 1, mean, na.rm=TRUE)  # mm/yr
MAT <- apply(meanTmon, 1, mean, na.rm=TRUE)  # degC
meanEPmon <- -1.55 + 0.96*(8.128 + 0.457*meanTmon)*meanSImon
 meanEPmon[meanEPmon < 0] <- 0                       # mean monthly potential evapotranspiration (mm/d)
PET <- 365.25*apply(meanEPmon, 1, mean, na.rm=TRUE)  # mm/yr


PETovP <- PET/MAP   # aridity index
MARovMAP <- MAR/MAP  # runoff ratio
ETovP <- (MAP - MAR)/MAP  # actual evaporation over precipitation
```

This exercise is instead about seasonality of runoff, which can be quantified for each station through the Pardé coefficients (i.e., the mean monthly runoff divided by the mean annual runoff):

```
PkQ <- meanQmon/matrix(MAQ, nrow=dim(meanQmon)[1], ncol=12, byrow=FALSE)
 head(PkQ, 15)
```

```
              Jan       Feb      Mar       Apr       May       Jun        Jul        Aug        Sep       Oct       Nov      Dec
6114500 2.1156706 2.5200348 1.866564 1.2119340 0.7728107 0.3684465 0.11933958 0.03360046 0.03707637 0.2386792 0.9419716 1.773873
6115500 2.2041612 2.8842653 2.328999 0.7828349 0.3966190 0.1339402 0.04941482 0.06892068 0.19375813 0.2639792 0.8452536 1.847854
6118010 2.1608288 2.0305871 1.568821 1.2432166 0.9472126 0.6038481 0.31968426 0.19536260 0.20128268 0.3907252 0.8228910 1.515540
6118015 2.5041889 2.3670982 1.727342 1.1058644 0.8499619 0.4021325 0.19192688 0.07768469 0.09139375 0.3244478 0.7494288 1.608530
6118020 2.7167070 2.3970944 1.670702 1.0605327 0.7990315 0.2615012 0.10169492 0.02905569 0.05811138 0.3050847 0.7554479 1.845036
6118030 2.3916084 2.2657343 1.804196 1.3006993 1.0489510 0.3776224 0.08391608 0.04195804 0.08391608 0.2937063 0.7552448 1.552448
6118060 2.4774775 2.3783784 1.723724 1.1231231 0.7477477 0.3963964 0.17117117 0.09009009 0.10810811 0.3153153 0.7747748 1.693694
6118070 2.0662252 2.0728477 1.509934 1.1523179 0.8211921 0.5496689 0.38410596 0.27152318 0.26490066 0.4437086 0.9205298 1.543046
6118150 1.8044010 1.8777506 1.525672 1.2029340 0.9095355 0.6308068 0.48410758 0.39608802 0.39608802 0.5427873 0.8508557 1.378973
6118165 1.6956522 1.6811594 1.405797 1.0289855 0.8260870 0.6014493 0.52898551 0.44202899 0.44202899 0.6521739 0.9855072 1.710145
6118175 1.8084112 1.9626168 1.598131 1.2757009 0.9252336 0.6168224 0.44859813 0.35046729 0.33644860 0.4766355 0.8271028 1.373832
6118205 1.6016949 1.6398305 1.423729 1.1440678 0.9788136 0.7118644 0.55932203 0.45762712 0.48305085 0.6991525 0.9915254 1.309322
6118210 2.4013068 2.3849714 1.715219 1.1075415 0.9343861 0.4181868 0.26790090 0.11108086 0.11761503 0.3234413 0.6926218 1.525728
6119010 1.0991239 1.0960711 1.042098 1.2418720 1.5031902 1.4042800 0.84012577 0.51298959 0.49906890 0.6771072 0.9340294 1.150044
6119020 0.8968464 0.8988975 1.007093 1.4178275 1.9880352 1.5229468 0.73686010 0.35689257 0.43175797 0.7091702 1.0229895 1.010683
```

```
 summary(PkQ)
```

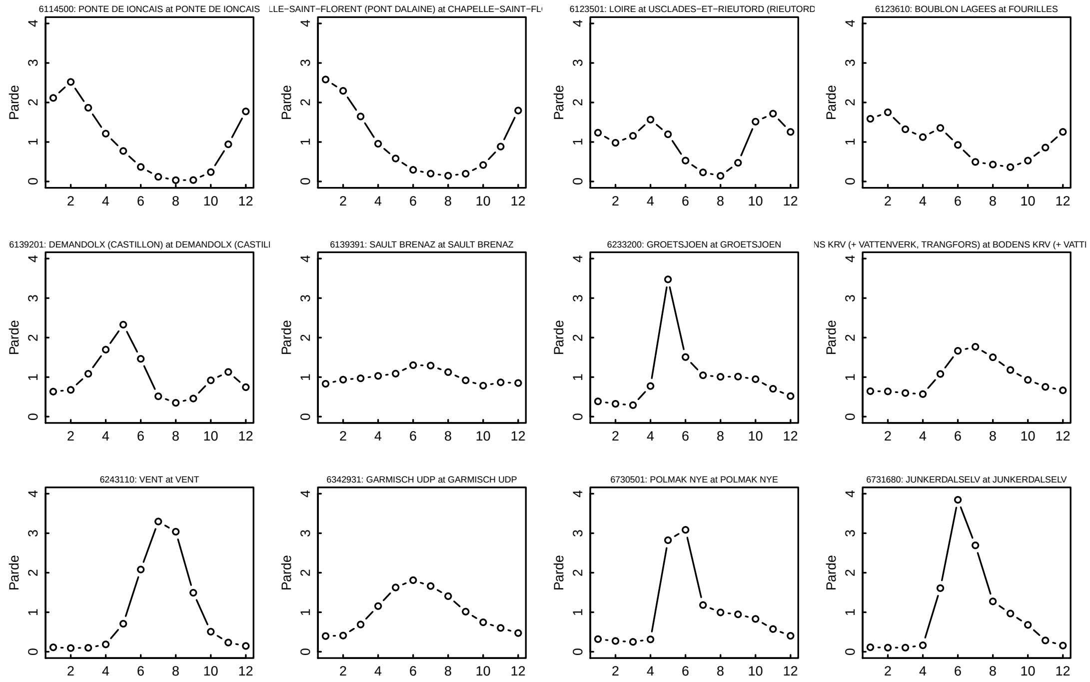
```
      Jan                Feb                Mar              Apr                May                Jun                Jul                Aug                Sep                Oct          
 Min.   :0.02707   Min.   :0.02331   Min.   :0.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.1339   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.07339  
 1st Qu.:0.74262   1st Qu.:0.79000   1st Qu.:1.006   1st Qu.:0.9378   1st Qu.:0.7596   1st Qu.:0.5284   1st Qu.:0.3992   1st Qu.:0.3856   1st Qu.:0.4426   1st Qu.:0.63067  
 Median :1.33287   Median :1.31806   Median :1.260   Median :1.1559   Median :1.0236   Median :0.7517   Median :0.5840   Median :0.5679   Median :0.6311   Median :0.75503  
 Mean   :1.24073   Mean   :1.21522   Mean   :1.194   Mean   :1.1590   Mean   :1.1533   Mean   :0.9913   Mean   :0.7786   Mean   :0.6677   Mean   :0.6526   Mean   :0.79524  
 3rd Qu.:1.70512   3rd Qu.:1.58963   3rd Qu.:1.492   3rd Qu.:1.3524   3rd Qu.:1.4069   3rd Qu.:1.2920   3rd Qu.:1.0705   3rd Qu.:0.9028   3rd Qu.:0.8520   3rd Qu.:0.92600  
 Max.   :3.78947   Max.   :2.97248   Max.   :2.881   Max.   :3.4711   Max.   :4.5092   Max.   :3.8880   Max.   :3.3553   Max.   :3.2591   Max.   :1.7792   Max.   :1.60628  
      Nov              Dec        
 Min.   :0.1098   Min.   :0.0000  
 1st Qu.:0.7739   1st Qu.:0.8081  
 Median :0.9282   Median :1.2349  
 Mean   :0.9919   Mean   :1.1606  
 3rd Qu.:1.2216   3rd Qu.:1.5471  
 Max.   :4.4211   Max.   :2.0738  
```

```
 summary(apply(PkQ, 1, mean))
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
      1       1       1       1       1       1 
```

To visualise the Pardé coefficients you can do (some examples plotted hereafter):

```
for (i in 1:length(CatchmentsEU$code)) {
 plot(c(1:12), PkQ[i,], type="b", xlab="", ylab="Parde",
      main=paste(CatchmentsEU$code[i], ": ", CatchmentsEU$river[i], " at ", CatchmentsEU$station[i], sep=""),
      cex.main=1, font.main=1, ylim=c(0,4))
 readline(i)
}
```



There are similarities and differences. It would be nice to have a way to describe these courves in a compact way and plot them on a map. Propose possibilities.

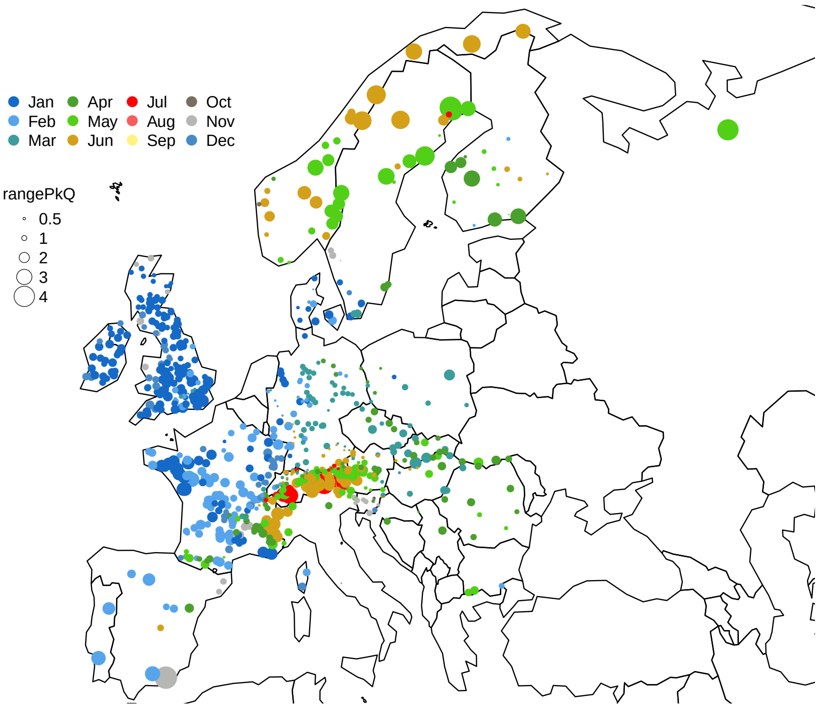
To me the characteristics of runoff seasonality that should be captured are (1) its amplitude (difference between months, range of Pardé) and (2) phase (when does the maximum/minimum occurs)? The simplest way, that I can think of, is to plot on a map points whose color correspond to the month of maximum runoff and whose size is proportional to the range of the Pardé coefficients:

```
monMaxPkQ <- apply(PkQ, 1, FUN=which.max)
rangePkQ <- apply(PkQ, 1, FUN=function(x){max(x) - min(x)})
 summary(rangePkQ)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.08084 1.04900 1.34900 1.43500 1.67600 4.42100
```

which plot like that:

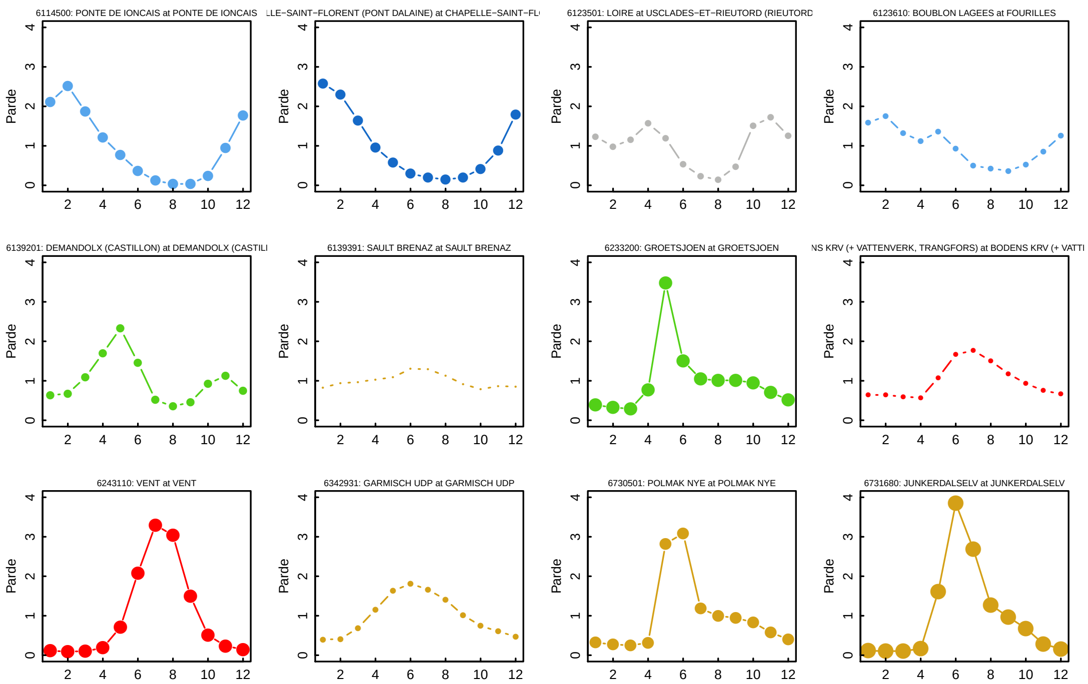
```
library(rworldmap)
newMap <- getMap(resolution="coarse")  # you can use resolution="low", which is better


# for colors
colori.stagioni <- colorRampPalette(c("#1569C7","#56A5EC","#3B9C9C","#4AA02C","#52D017","#D4A017",
                                      "#FF0000","#F75D59","#FFF380","#786D5F","#B6B6B4","#488AC7"))
# for avoiding overlap
ordina <- order(rangePkQ, decreasing=TRUE)
# you want to try also with
#ordina <- order(rangePkQ, decreasing=FALSE)
plot(newMap, xlim=range(CatchmentsEU$lon), ylim=range(CatchmentsEU$lat))
 points(CatchmentsEU$lon[ordina], CatchmentsEU$lat[ordina], pch=16,
        cex=.7*rangePkQ[ordina],
        col=colori.stagioni(12)[monMaxPkQ[ordina]])
legend("topleft", legend=c(.5,1,2,3,4), title="rangePkQ", pch=1, pt.lwd=.5,
       pt.cex=.7*c(.5,1,2,3,4), box.col="white", bg="white", inset=c(0,0.25))
legend("topleft", legend=month.abb, pch=16, pt.cex=1.5,
       col=colori.stagioni(12),
       ncol=4, box.col="white", bg="white", inset=c(0,0.11))
```



Well, some nice spatial patterns appear and may be useful for interpretation. What do you see?

The previous plots can be coloured now:

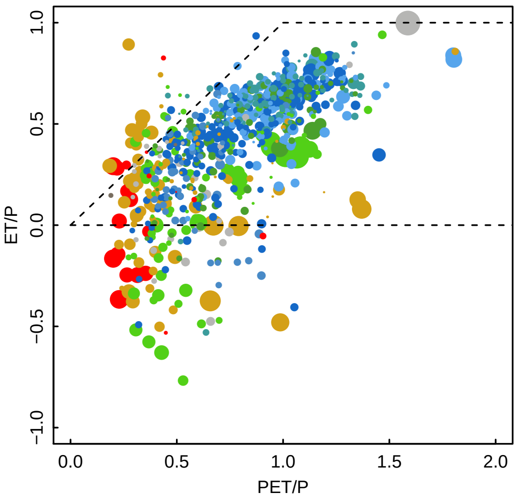
```
for (i in 1:length(CatchmentsEU$code)) {
 plot(c(1:12), PkQ[i,], type="b", xlab="", ylab="Parde",
      main=paste(CatchmentsEU$code[i], ": ", CatchmentsEU$river[i], " at ", CatchmentsEU$station[i], sep=""),
      cex.main=1, font.main=1, ylim=c(0,4),
      cex=.7*rangePkQ[i],
      pch=16, col=colori.stagioni(12)[monMaxPkQ[i]])
 readline(i)
}
```



The plotted points do not capture well the bimodalities, that's a compromise.

One question related to Chapter 5 is: does this way of plotting points help to interpret Budyko?

```
#ordina <- order(rangePkQ, decreasing=TRUE)
plot(PETovP[ordina], ETovP[ordina],
     xlim=c(0,2), ylim=c(-1,1), xlab="PET/P", ylab="ET/P", pch=16,
     col=colori.stagioni(12)[monMaxPkQ[ordina]],
     cex=.7*rangePkQ[ordina])
 segments(x0=c(0,1), x1=c(1,4), y0=c(0,1), y1=c(1,1), lty=2)
 segments(x0=0, x1=4, y0=0, lty=2)
```



Mmmm... maybe there is something. Maybe estimating seasonality could help to estimate annual runoff too!

2 Spatial proximity as explanatory variable

Since we see that there is quite a nice spatial coherence in the seasonality, spatial proximity should be a good way to predict Pardé in ungauged basins. Let's try the nearest neighbor approach.

We have latitude and longitude of the gauging stations, let's use them.

```
head(CatchmentsEU, 15)
```

```
      code                          station                               river     lon     lat   elev   area country
1  6114500                PONTE DE IONCAIS                    PONTE DE IONCAIS -7.5200 40.6100 330.00  606.0      PT
2  6115500                  M.DA GAMITINHA                      M.DA GAMITINHA -8.4000 38.0700  28.00 2721.0      PT
3  6118010             SAINT-JEAN-BREVELAY                               CLAIE -2.7033 47.8248  99.98  137.0      FR
4  6118015               LOYAT (PONT D129)                                YVEL -2.3688 47.9938  88.26  315.0      FR
5  6118020    GRAND-FOUGERAY (LA BERNADAISE)                  RUISSEAUX D ARON -1.6908 47.7122  68.00  118.0      FR
6  6118030        PAIMPONT (PONT DU SECRET)                                 AFF -2.1438 47.9816 119.01   30.2      FR
7  6118060                           GUENIN                                EVEL -2.9752 47.8999  95.16  316.0      FR
8  6118070              BANNALEC (PONT MEYA)                           STER-GOZ -3.7522 47.9068  85.08   69.7      FR
9  6118150                        TREZILIDE                             GUILLEC -4.0770 48.6150  91.10   43.0      FR
10 6118165                              RAI                               RISLE  0.5806 48.7479 255.45  149.0      FR
11 6118175                       PLOUGONVEN                              JARLOT -3.8005 48.5656  73.86   44.0      FR
12 6118205            SAINT-OUEN-LA-ROUERIE                            LOISANCE -1.4363 48.4270 103.34   81.5      FR
13 6118210        MONFORT-SUR-MEU (LABBAYE)                                 MEU -1.9448 48.1275  83.50  468.0      FR
14 6119010           BERENX (PONT DE BERENX)             BERENX (PONT DE BERENX) -0.8537 43.5078 100.44 2575.0      FR
15 6119020 OLORON-SAINTE-MARIE (OLORON-STE-CROIX) OLORON-SAINTE-MARIE (OLORON-STE-CROIX) -0.5971 43.1877 277.79  488.0      FR
```

On the internet I came around this function that roughly calculate the euclidean distance in kilometers between two points given in longitude and latitude values. It might be not too precise due to the inaccurate estimate of the earth radius (`R`). Juraj Parajka can tell you how to do it better :-)

```
# Calculate distance in kilometers between two points
earth.dist <- function (long1, lat1, long2, lat2) {
 rad <- pi/180
 a1 <- lat1 * rad
 a2 <- long1 * rad
 b1 <- lat2 * rad
 b2 <- long2 * rad
 dlon <- b2 - a2
 dlat <- b1 - a1
 a <- (sin(dlat/2))^2 + cos(a1) * cos(b1) * (sin(dlon/2))^2
 c <- 2 * atan2(sqrt(a), sqrt(1 - a))
 R <- 6378.145
 d <- R * c
 return(d)
}
```

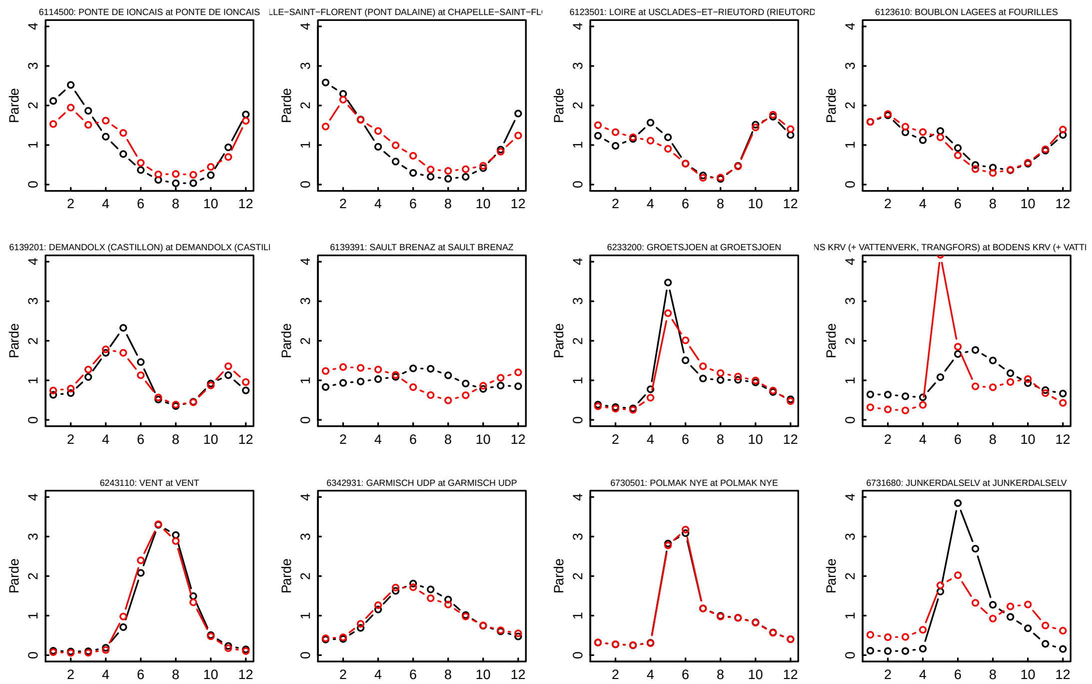
the function can be used as follows:

```
nn <- length(CatchmentsEU$code)
spatNN <- data.frame(matrix(ncol=2, nrow=nn, dimnames=list(CatchmentsEU$code, c("code","dist"))))
# THIS TAKES SOME SECONDS
for (i in 1:nn) {
 mindist=Inf
 dist=mindist
 for (j in 1:nn) {
  if (j != i) {
   # calculate distance
   distanza <- earth.dist(CatchmentsEU$lon[i], CatchmentsEU$lat[i], CatchmentsEU$lon[j], CatchmentsEU$lat[j])
   if (distanza < mindist) {
    finrow <- j
    mindist <- distanza
   }
  }
 }
 # write on output data.frame
 spatNN[i, "dist"] <- mindist
 spatNN[i, "code"] <- CatchmentsEU$code[finrow]
}


head(spatNN, 20)
```

```
              dist    code
6114500 254.257854 6212500
6115500 291.746091 6114500
6118010  21.989072 6118060
6118015  16.849268 6118030
6118020  31.854681 6123170
6118030  16.849268 6118015
6118060  21.989072 6118010
6118070  58.096816 6118060
6118150  21.123102 6118175
6118165  62.066731 6122150
6118175  21.123102 6118150
6118205  50.318846 6118210
6118210  21.974031 6118030
6119010  29.122297 6119040
6119020   1.463051 6119030
6119030   1.463051 6119020
6119040  22.290258 6119030
6119050  38.200024 6119040
6119110  19.894205 6119120
6119120  19.894205 6119110
```

Check it visually for each catchment:

```
for (i in 1:length(CatchmentsEU$code)) {
 plot(c(1:12), PkQ[i,], type="b", xlab="", ylab="Parde",
      main=paste(CatchmentsEU$code[i], ": ", CatchmentsEU$river[i], " at ", CatchmentsEU$station[i], sep=""),
      cex.main=1, font.main=1, ylim=c(0,4))
 donor <- which(CatchmentsEU$code == spatNN$code[i])
 lines(c(1:12), PkQ[donor,], type="b", col=2)
 readline(i)
}
```



It is not too bad, but what about using a compact error measure to plot the goodness of our method? A possible one is the widely used Nash-Sutcliffe coefficient, which is also used in the PUB book assessment of Chapter 6 (Blöschl et al., 2013, page 130).

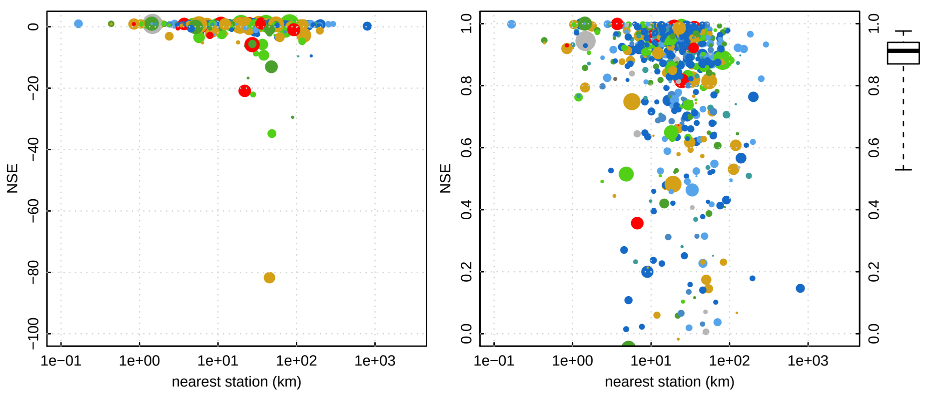
```
calcNSE <- function (est, obs) {
 # est = estimated Parde (vector of 12 values)
 # obs = observed Parde (vector of 12 values)
 est <- as.numeric(est)
 obs <- as.numeric(obs)

 # Nash efficiency ()
 mobs <- mean(obs)
 NSE <- 1 - sum((est - obs)^2)/sum((obs - mobs)^2)
 return(NSE)
}
```

Let's use it for all catchments:

```
NSEs <- rep(NA, length(CatchmentsEU$code))
for (i in 1:length(CatchmentsEU$code)) {
 donor <- which(CatchmentsEU$code == spatNN$code[i])
 NSEs[i] <- calcNSE(PkQ[donor,], PkQ[i,])
}
spatNN$NSE <- NSEs
```

And let's plot it as a function of distance between neighbors and with a boxplot that can be compared with the assessment figures in the PUB book.

```
boxplot20 <- function(m, ...){
 # m has to be a data.frame or list
 bp <- boxplot(m, plot=FALSE)
 bp$stats <- sapply(m, function(x)
                  quantile(x, c(0.2,0.4, 0.5, 0.6, 0.8), na.rm=TRUE))
 bxp(bp, outline=FALSE, ...)
}
layout(matrix(1:3, nrow=1), widths=c(5,5,1))
DD <- spatNN$dist
DD[DD == 0] <- NA
ordina <- order(rangePkQ, decreasing=TRUE)
plot(DD[ordina], spatNN$NSE[ordina],
     xlab="nearest station (km)", ylab="NSE",
     log="x", xlim=c(1e-1, 3e3), ylim=c(-100, 1),
     cex=.7*rangePkQ,
     pch=16, col=colori.stagioni(12)[monMaxPkQ])
 grid()
plot(DD[ordina], spatNN$NSE[ordina],
     xlab="nearest station (km)", ylab="NSE",
     log="x", xlim=c(1e-1, 3e3), ylim=c(0, 1),
     cex=.7*rangePkQ,
     pch=16, col=colori.stagioni(12)[monMaxPkQ])
 grid()
 axis(4)
par(mar=c(3,0,2,0)+0.03)
boxplot20(as.data.frame(spatNN$NSE),
          ylim=c(0, 1), axes=FALSE)
```



Comparing it to Fig. 6.28 at page 130 of the book, the result is nice.

3 What about the climate seasonality?

So far we have used spatial proximity as similarity measure. Since we have information on mean montly precipitation and potential evaporation, it is reasonable to think that they should relate to the seasonality of runoff. We can think of expressing the seasonality of precipitation and potential evaporation also through Pardé:

```
PkP <- meanPmon/matrix(MAP/365.25, nrow=dim(meanPmon)[1], ncol=12, byrow=FALSE)
PkEP <- meanEPmon/matrix(PET/365.25, nrow=dim(meanEPmon)[1], ncol=12, byrow=FALSE)
```

and plot it along with the runoff Pardé:

```
for (i in 1:length(CatchmentsEU$code)) {
 plot(c(1:12), PkQ[i,], type="b", xlab="", ylab="Parde",
      main=paste(CatchmentsEU$code[i], ": ", CatchmentsEU$river[i], " at ", CatchmentsEU$station[i], sep=""),
      cex.main=1, font.main=1, ylim=c(0,4))
  lines(c(1:12), PkEP[i,], type="b", pch=2, col="red")
  lines(c(1:12), PkP[i,], type="b", pch=6, col="blue")
 readline(i)
}
```

Following Milly (1994) and Woods (2003), the climate seasonality index, S can be defined as

$$S = |\delta_P - \delta_E R|$$

where R is the aridity index and δ_P and δ_E are half of the amplitudes of the seasonal cycle as expressed in Pardé, with the convention of being positive if the maximum is in summer and negative if it is in winter (i.e., δ_E is always positive in the northern emisphere).

If we calculate month of maximum and ranges as before:

```
monMaxPkP <- apply(PkP, 1, FUN=which.max)
rangePkP <- apply(PkP, 1, FUN=function(x){max(x, na.rm=TRUE) - min(x, na.rm=TRUE)})
rangePkEP <- apply(PkEP, 1, FUN=function(x){max(x) - min(x)})
```

then

```
deltaE <- rangePkEP/2
sw <- cut(monMaxPkP, breaks=c(1,4,10,12), include.lowest=TRUE)  # summer-winter
 levels(sw) <- c("winter", "sommer", "winter")                  # summer-winter
deltaP <- c(-1,1)[as.numeric(sw)]*rangePkP/2
seasS <- abs(deltaP - deltaE*PETovP)
```

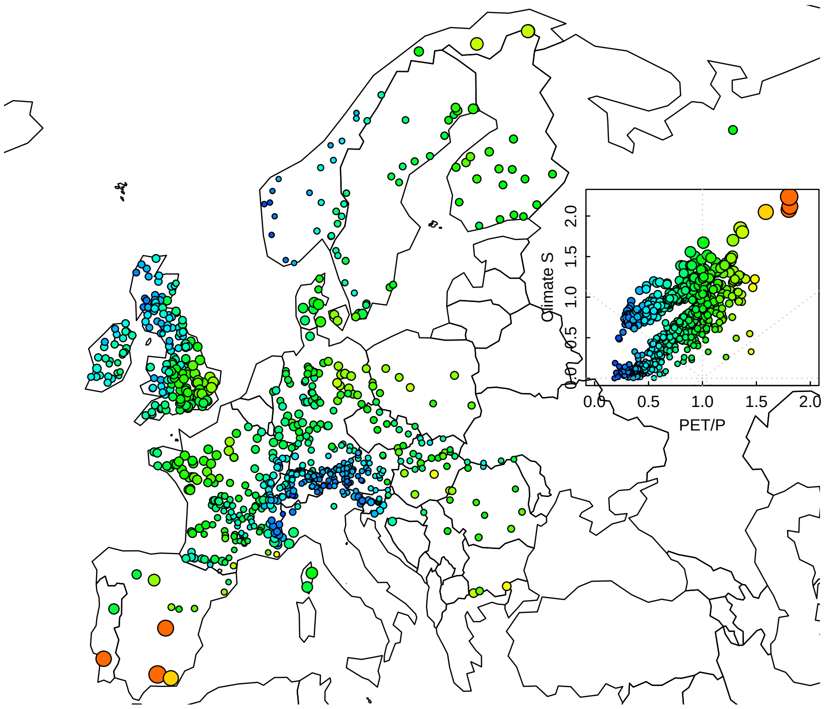
How do we use these information for understanding the climate seasonality and plot it on a map?

The following lines are taken from Woods (2003): "When $R > 1$, mean potential evaporation exceeds mean rainfall (i.e., a relatively dry climate), whereas $R < 1$ indicates a relatively wet climate. Small values of S indicate that the balance between rainfall and potential evaporation does not change much during the seasonal cycle. So for example, if $R < 1$ and $S < 1 - R$, then we have a wet climate in which rain exceeds potential evaporation both in the long term, and in each season. One might say that the duration of the wet season is the whole seasonal cycle. If instead $R > 1$ and $S < R - 1$, then we have a dry climate with potential evaporation exceeding rain throughout the year, and one might say there is no wet season. Large values of S (greater than the larger of $(R - 1)$ and $(1 - R)$ indicate that a switch occurs from a rainfall surplus in the wet season (the part of the seasonal cycle when $P > E_P$) to a rainfall deficit for the rest of the seasonal cycle (the dry season, when $P < E_P$)."

I would like to plot these information on a map. Why not using the aridity index for point colors and the magnitude of the climatic seasonality as point size?

```
library(Hmisc)  # for multiple plots
# colors related to aridity index
colori.aridity <- rev(rainbow(20, start=0, end=.65, alpha=1))
plot(newMap, xlim=range(CatchmentsEU$lon), ylim=range(CatchmentsEU$lat))
 points(CatchmentsEU$lon, CatchmentsEU$lat, pch=21,
        cex=0.5 + exp(seasS)/5,
        bg=colori.aridity[round(10*PETovP)])
subplot(   # needs package 'Hmisc'
 {plot(PETovP, seasS, pch=21, xlim=c(0, 2),
       xlab="PET/P", ylab="Climate S",
       bg=colori.aridity[round(10*PETovP)],
```

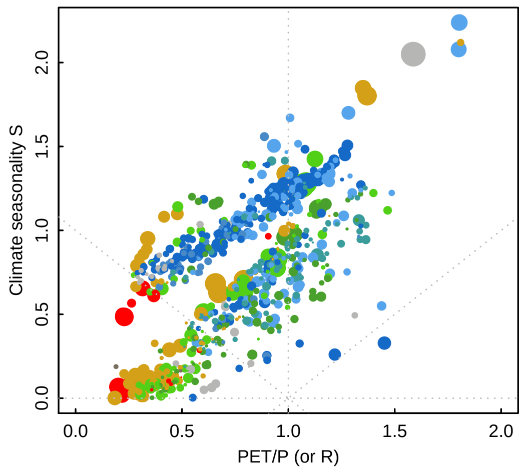
```
       cex=0.5 + exp(seasS)/5)
  abline(v=1, col="grey", lty=3)
  abline(h=0, col="grey", lty=3)
  abline(1, -1, col="grey", lty=3)
  abline(-1, 1, col="grey", lty=3)},
 c(33,53), c(52,62)
)
```



Can we interpret the map and the graph?

Let's try to plot S and R vs the seasonality of runoff in a graph:

```
#ordina <- order(rangePkQ, decreasing=TRUE)
plot(PETovP[ordina], seasS[ordina], pch=16, xlim=c(0, 2),
     xlab="PET/P (or R)", ylab="Climate seasonality S",
     col=colori.stagioni(12)[monMaxPkQ[ordina]],
     cex=.7*rangePkQ[ordina])
 abline(v=1, col="grey", lty=3)
 abline(h=0, col="grey", lty=3)
 abline(1, -1, col="grey", lty=3)
 abline(-1, 1, col="grey", lty=3)
```



It is interestingly showing two clusters of points, that can be divided approximatively by the one to one line (that's courious). How do we interpret them?

If you think this is an artefact of our analysis, can you propose another measure?

4 Climate/catchment similarity as explanatory variable

That's analogous as spatial proximity but now the proximity is searched in the space of climate/catchment characteristics.

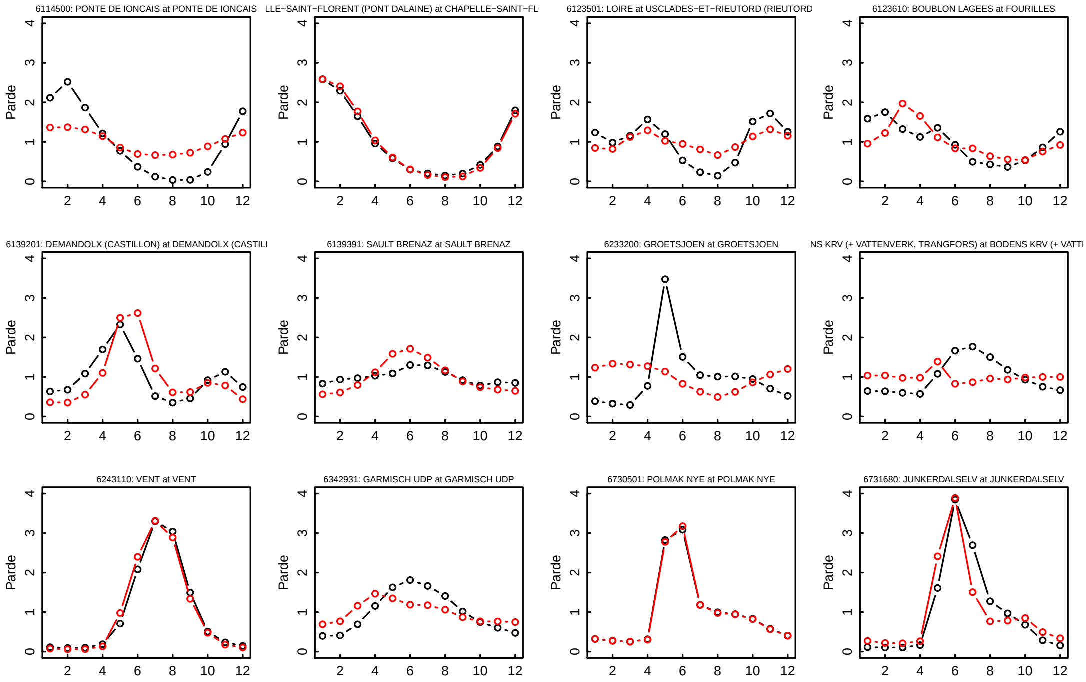
In the aridity - climate seasonality space, there was also a spatial coherence in terms of runoff seasonality (1st figure in Section 3). Therefore why not trying the nearest neighbor in that space?

```
# I do it differently from before, I use the command dist (see help(dist))
climDistances <- dist(cbind(PETovP, seasS), method="euclidean", diag=FALSE, upper=TRUE)
 # I did not rescale the two variables because their magnitude is comparable
climDistances <- as.matrix(climDistances)
 diag(climDistances) <- 1000
finrow <- apply(climDistances, 1, which.min)
mindist <- apply(climDistances, 1, min)
climNN <- data.frame(dist=mindist, code=CatchmentsEU$code[finrow])
 head(climNN, 20)  # climatic nearest neighbor
```

```
               dist    code
6114500 0.070448333 6934250
6115500 0.042963491 6216800
6118010 0.010777044 6607711
6118015 0.008068030 6607800
6118020 0.005306711 6122150
6118030 0.002642327 6123200
6118060 0.007226626 6607705
6118070 0.007378366 6605700
6118150 0.017436606 6125250
6118165 0.009533171 6119300
6118175 0.038418831 6604625
6118205 0.004737708 6605640
6118210 0.010916202 6123190
6119010 0.009625814 6608100
6119020 0.028236425 6604860
6119030 0.045870816 6604660
6119040 0.012299348 6604670
6119050 0.009330283 6604670
6119110 0.009995289 6136151
6119120 0.002512259 6607230
```

Check it visually:

```
for (i in 1:length(CatchmentsEU$code)) {
 plot(c(1:12), PkQ[i,], type="b", xlab="", ylab="Parde",
      main=paste(CatchmentsEU$code[i], ": ", CatchmentsEU$river[i], " at ", CatchmentsEU$station[i], sep=""),
      cex.main=1, font.main=1, ylim=c(0,4))
 donor <- which(CatchmentsEU$code == climNN$code[i])
 lines(c(1:12), PkQ[donor,], type="b", col=2)
 readline(i)
}
```



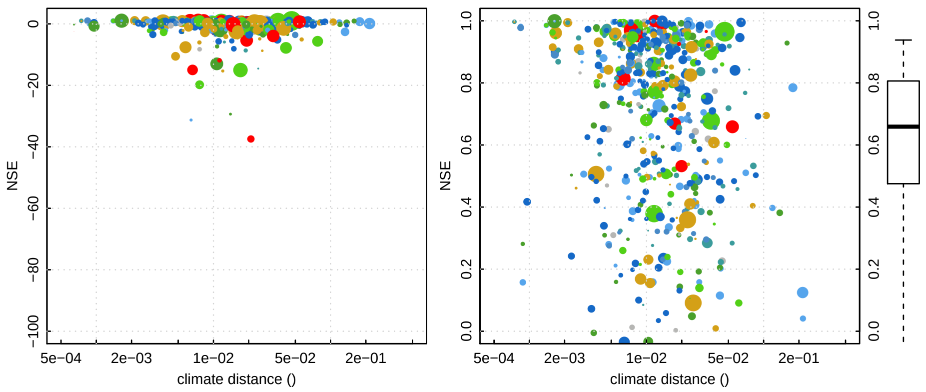
Looks less good that the nearest neighbor in geographical space.

What about Nash-Sutcliffe?

```
NSEs <- rep(NA, length(CatchmentsEU$code))
for (i in 1:length(CatchmentsEU$code)) {
 donor <- which(CatchmentsEU$code == climNN$code[i])
 NSEs[i] <- calcNSE(PkQ[donor,], PkQ[i,])
}
climNN$NSE <- NSEs
```

which plots:

```
layout(matrix(1:3, nrow=1), widths=c(5,5,1))
DD <- climNN$dist
DD[DD == 0] <- NA
ordina <- order(rangePkQ, decreasing=TRUE)
plot(DD[ordina], climNN$NSE[ordina],
     xlab="climate distance ()", ylab="NSE",
     log="x", xlim=c(5e-4, 5e-1), ylim=c(-100, 1),
     cex=.7*rangePkQ,
     pch=16, col=colori.stagioni(12)[monMaxPkQ])
 grid()
plot(DD[ordina], climNN$NSE[ordina],
     xlab="climate distance ()", ylab="NSE",
     log="x", xlim=c(5e-4, 5e-1), ylim=c(0, 1),
     cex=.7*rangePkQ,
     pch=16, col=colori.stagioni(12)[monMaxPkQ])
 grid()
 axis(4)
par(mar=c(3,0,2,0)+0.03)
boxplot20(as.data.frame(climNN$NSE),
          ylim=c(0, 1), axes=FALSE)
```



Yes, spatial proximity was better.

What about adding catchment characteristics?

What about combining spatial location and climate/catchment characteristics?

5 Classify the seasonal runoff Pardé coefficients into regime types

Here we reason as geographers, we want to give names to the seasonality of runoff, to identify regime types.

We can try a classification based on runoff seasonality using the `kmeans` procedure:

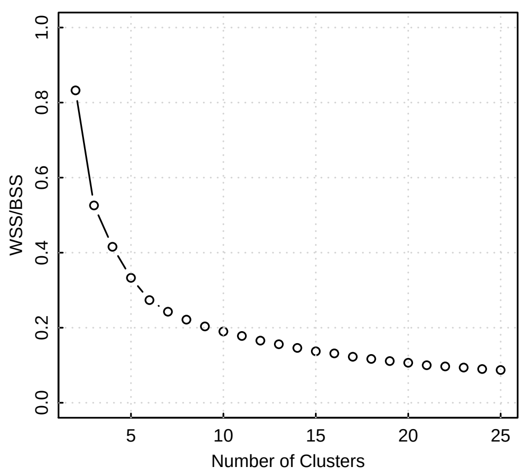
```
help(kmeans)
```

Let's calculate the ration of the within and between variances for different number of clusters (se to know approximatively how many clusters to look for):

```
WSSovBSS <- NULL    # within sum of squares over between sum of squares
for (i in 1:24) {
 dummy <- kmeans(PkQ, centers=i+1, nstart=100, iter.max=100)
 WSSovBSS[i] <- dummy$tot.withinss/dummy$betweenss
}
```

and plot it:

```
plot((1:length(WSSovBSS))+1, WSSovBSS, type="b", xlab="Number of Clusters", ylim=c(0,1),
 ylab="WSS/BSS")
 grid()
```

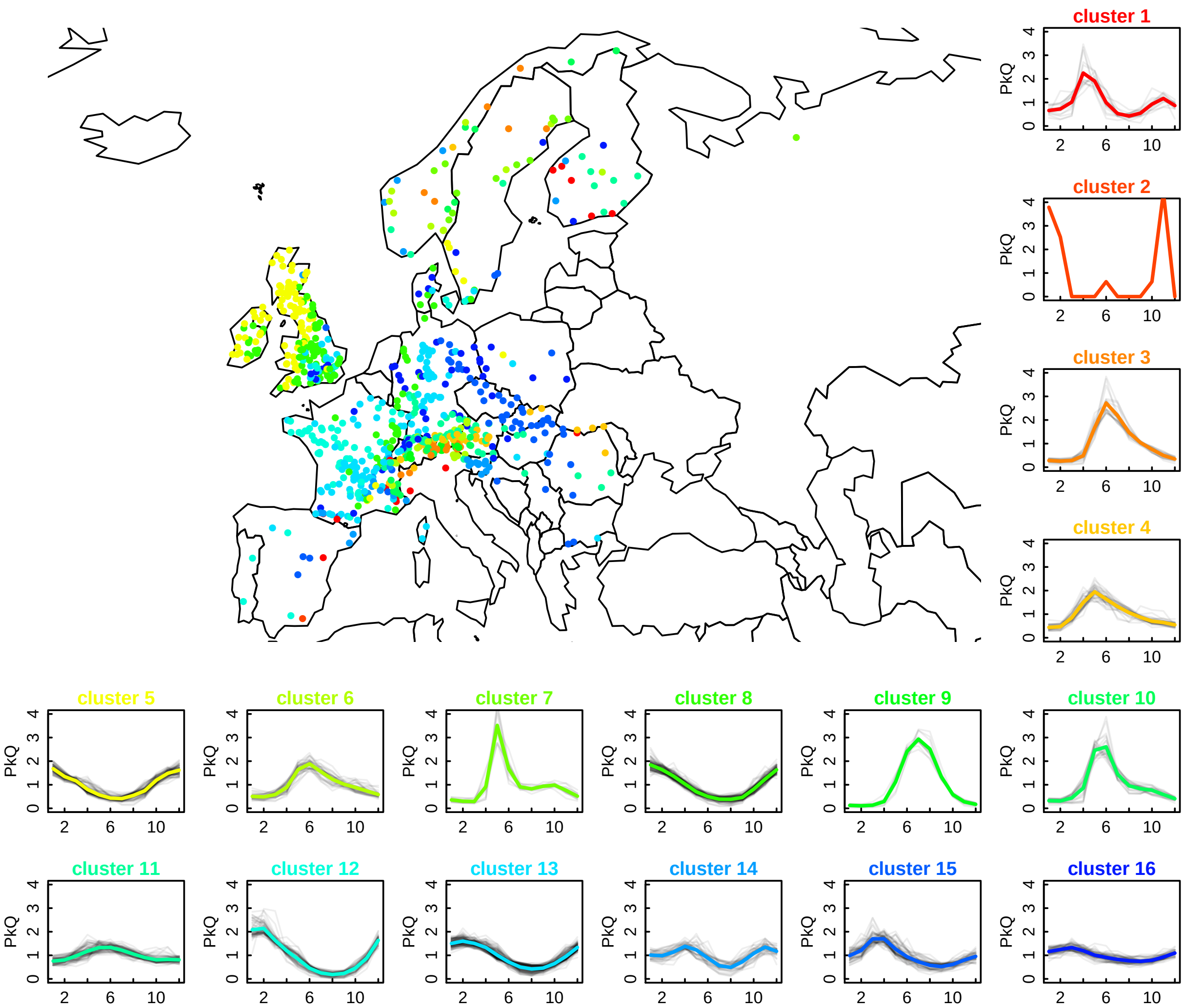


Ok, after 10 clusters we are below 0.2. Let's try to group first in more than 10 clusters:

```
nclasters=16
# K-Means Cluster Analysis
fit <- kmeans(PkQ, nclasters, nstart=100)
# save cluster assignment
fit.cluster <- fit$cluster
 names(fit.cluster) <- CatchmentsEU$code
# get cluster means
KmenasEU <- aggregate(PkQ, by=list(fit.cluster), FUN=mean)[,-1]
 rownames(KmenasEU) <- paste("cluster", 1:nclasters, sep="")
```

and plot them in a fancy way:

```
layout(matrix(c(1,1,1,1,1,2,
                1,1,1,1,1,3,
                1,1,1,1,1,4,
                1,1,1,1,1,5,
                6:17), ncol=6, byrow=TRUE))
par(mar=c(2.3,2.3,1.3,1)+0.03, mgp=c(1.5,0.3,0), tcl=.2, xaxs="r", yaxs="r")
 plot(newMap, xlim=range(CatchmentsEU$lon, na.rm=TRUE), ylim=range(CatchmentsEU$lat, na.rm=TRUE))
  points(CatchmentsEU$lon, CatchmentsEU$lat, pch=16,
         col=rainbow(nclasters, start=0, end=.65, alpha=1)[fit.cluster],
         cex=1)
 for (j in 1:nclasters) {
  plot(c(1,12), c(0,4), type="n", xlab="", ylab="PkQ", main=paste("cluster", j),
       col.main=rainbow(nclasters, start=0, end=.65, alpha=1)[j])
  dummy <- PkQ[fit.cluster == j,]
  for (i in 1:dim(dummy)[1]) {
   lines(seq(1,12), dummy[i,], col="#00000011")
  }
  lines(seq(1,12), KmenasEU[j,], lwd=2, col=rainbow(nclasters, start=0, end=.65, alpha=1)[j])
 }
```



Are all these groups different enough between themselves?

Personally I think that 6 clusters would be enough. In my case I would identify:

- strong peak in summer (like for example cluster 9 or cluster 3);
- strong peak in spring (like for example cluster 7);
- spring maximum (like for example cluster 4 or cluster 6);
- winter maximum (like for example cluster 5 or cluster 8);
- bimodal (like for example cluster 1 or cluster 14);
- weak seasonality (like for example cluster 11 or cluster 16).

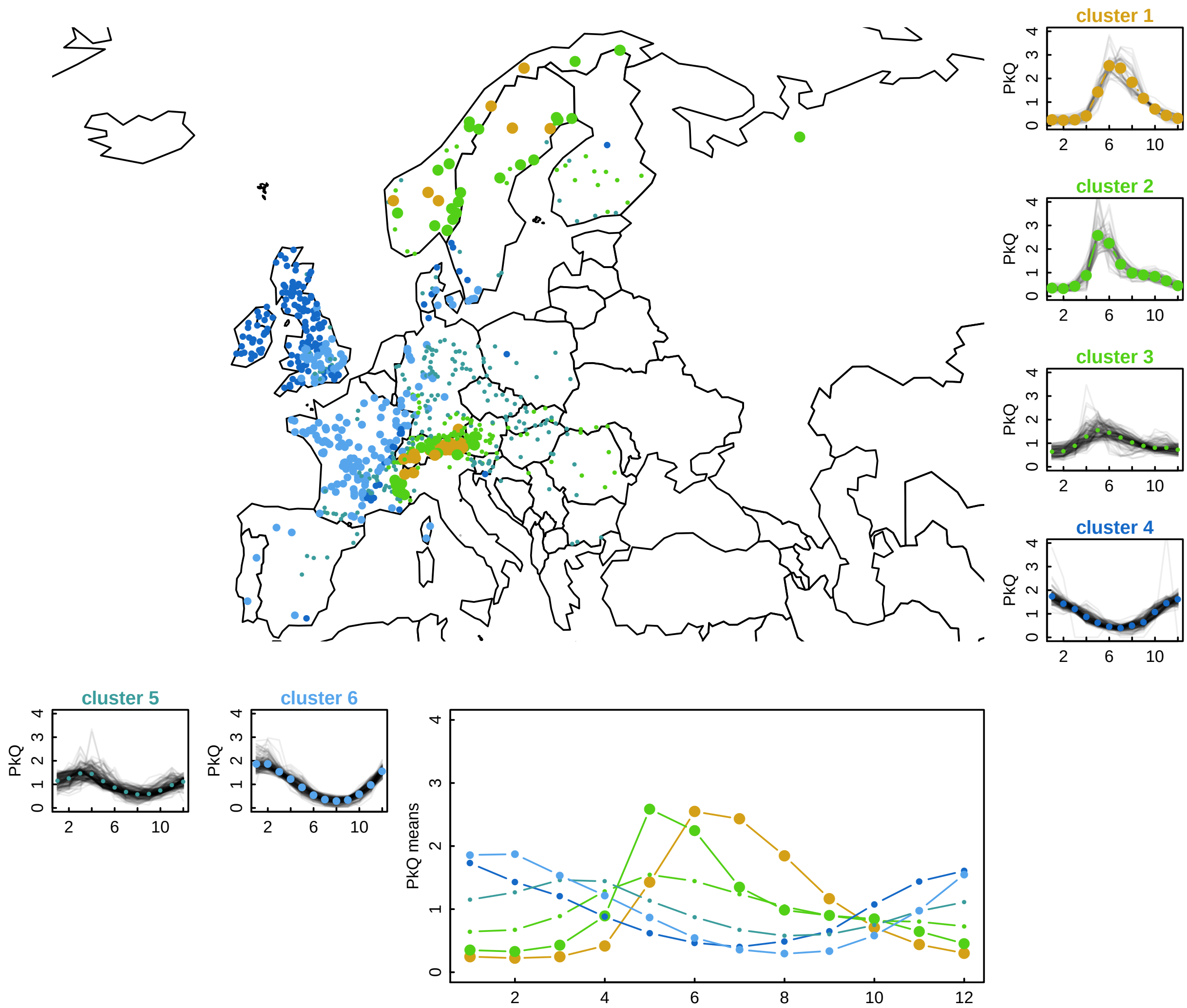
(notice that you will have different clusters, at least in terms of ordering).

Let's `kmeans` do the job again but where we choose the initial cluster centers:

```
centri <- rbind(fit$centers[9,],  # strong peak in summer
                fit$centers[7,],  # strong peak in spring
                fit$centers[4,],  # spring maximum
                fit$centers[5,],  # winter maximum
                fit$centers[14,], # bimodal
                fit$centers[16,]) # weak seasonality



nclasters=dim(centri)[1]
# K-Means Cluster Analysis
fit <- kmeans(PkQ, centers=centri, nstart=100)
# save cluster assignment
fit.cluster <- fit$cluster
 names(fit.cluster) <- CatchmentsEU$code
# get cluster means
KmenasEU <- aggregate(PkQ, by=list(fit.cluster), FUN=mean)[,-1]
 rownames(KmenasEU) <- paste("cluster", 1:nclasters, sep="")
```

Which can be plotted using the same code used for the previous figure. Let's try to plot it nicely using colors and sizes proposed in Section 1.

```
monMaxKmenasEU <- apply(KmenasEU, 1, FUN=which.max)
 monMaxKmenasEU

cluster1 cluster2 cluster3 cluster4 cluster5 cluster6
       6        5        5        1        3        2

rangeKmenasEU <- apply(KmenasEU, 1, FUN=function(x){max(x) - min(x)})
 rangeKmenasEU

 cluster1  cluster2  cluster3  cluster4  cluster5  cluster6
2.3243210 2.2594389 0.9016973 1.3277493 0.8857319 1.5791053



layout(matrix(c(1,1,1,1,1,2,
                1,1,1,1,1,3,
                1,1,1,1,1,4,
                1,1,1,1,1,5,
                6,7,8,8,8,0,
                0,0,8,8,8,0), ncol=6, byrow=TRUE))
par(mar=c(2.3,2.3,1.3,1)+0.03, mgp=c(1.5,0.3,0), tcl=.2, xaxs="r", yaxs="r")
 plot(newMap, xlim=range(CatchmentsEU$lon, na.rm=TRUE), ylim=range(CatchmentsEU$lat, na.rm=TRUE))
  points(CatchmentsEU$lon, CatchmentsEU$lat, pch=16,
         cex=.7*rangeKmenasEU[fit.cluster],
         col=colori.stagioni(12)[monMaxKmenasEU[fit.cluster]])
 for (j in 1:nclasters) {
  plot(c(1,12), c(0,4), type="n", xlab="", ylab="PkQ", main=paste("cluster", j),
       col.main=colori.stagioni(12)[monMaxKmenasEU[j]])
  dummy <- PkQ[fit.cluster == j,]
  for (i in 1:dim(dummy)[1]) {
   lines(seq(1,12), dummy[i,], col="#00000011")
  }
  lines(seq(1,12), fit$centers[j,], type="b", pch=16,
        cex=.7*rangeKmenasEU[j],
        col=colori.stagioni(12)[monMaxKmenasEU[j]])
 }
plot(c(1,12), c(0,4), type="n", xlab="", ylab="PkQ means")
for (j in 1:nclasters) {
 lines(seq(1,12), KmenasEU[j,],  type="b", pch=16,
       cex=.7*rangeKmenasEU[j],
       col=colori.stagioni(12)[monMaxKmenasEU[j]])
}
```



Therefore, when maximising the between variance and minimising the within variance for 6 clusters I get the following:

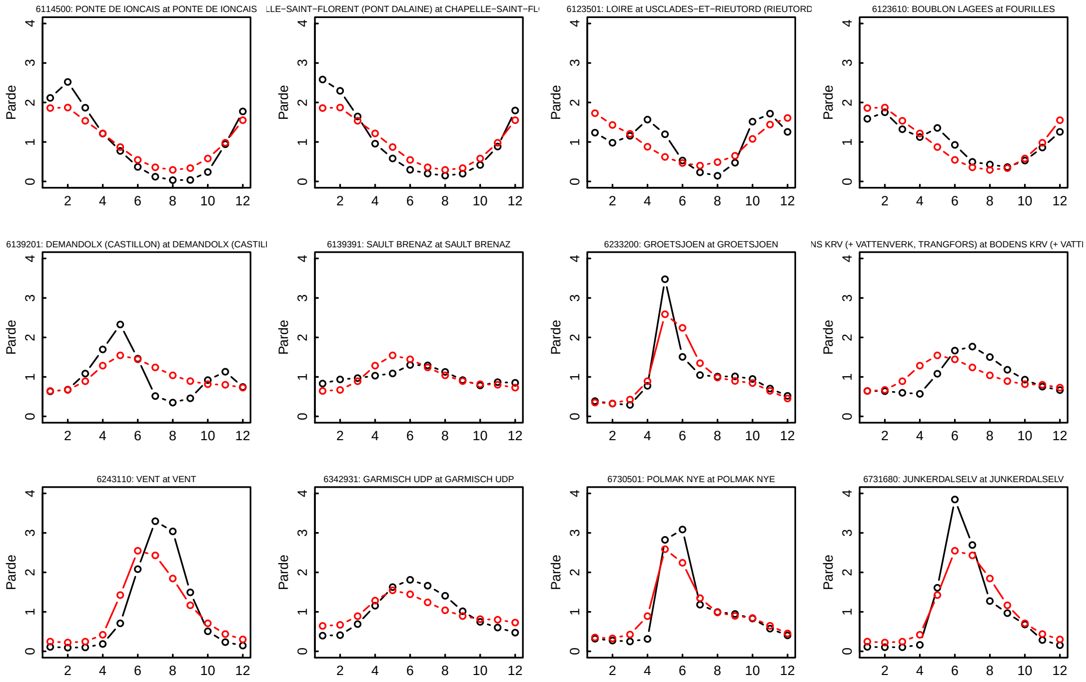
- cluster 1: peak in summer;
- cluster 2: peak in spring;
- cluster 3: late-spring maximum;
- cluster 4: late-fall maximum;
- cluster 5: early-spring maximum;
- cluster 6: winter maximum.

(notice that you will have different clusters, at least in terms of ordering).

What would be the error if I use the mean cluster Pardé as estimate in ungauged basins? (notice that this is not estimation in ungauged basins though).

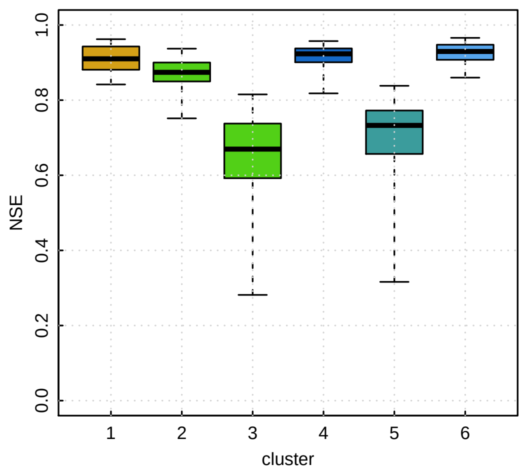
Check it visually:

```
for (i in 1:length(CatchmentsEU$code)) {
 plot(c(1:12), PkQ[i,], type="b", xlab="", ylab="Parde",
      main=paste(CatchmentsEU$code[i], ": ", CatchmentsEU$river[i], " at ", CatchmentsEU$station[i], sep=""),
      cex.main=1, font.main=1, ylim=c(0,4))
 donorCluster <- fit.cluster[i]
 lines(c(1:12), KmenasEU[donorCluster,], type="b", col=2)
 readline(i)
}
```



What about Nash-Sutcliffe?

```
classifications <- data.frame(fit.cluster=fit.cluster)
 rownames(classifications) <- CatchmentsEU$code
NSEs <- rep(NA, length(CatchmentsEU$code))
for (i in 1:length(CatchmentsEU$code)) {
 donorCluster <- fit.cluster[i]
 NSEs[i] <- calcNSE(KmenasEU[donorCluster,], PkQ[i,])
}
classifications$NSE0 <- NSEs  # NSE0 because is not PUB


boxplot20(split(classifications$NSE0, classifications$fit.cluster),
          ylim=c(0,1), xlab="cluster", ylab="NSE",
          boxfill=colori.stagioni(12)[monMaxKmenasEU])
 grid()
```



Apart clusters 3 and 5, the result is nice.

```
quantile(classifications$NSE0)
```

```
          0%         25%         50%         75%        100%
-121.2799179   0.7083537   0.8448460   0.9315927   0.9912498
```

But what about prediction in ungauged basins?

6 Allocate ungauged basins to the regime types

Looking at the map with the clusters, again the spatial coherence is striking. The most natural way is to use the spatial proximity again, e.g., through the nearest neighbor:

```
NSEs <- rep(NA, length(CatchmentsEU$code))
donorCluster <- rep(NA, length(CatchmentsEU$code))
for (i in 1:length(CatchmentsEU$code)) {
 donor <- which(CatchmentsEU$code == spatNN$code[i])
 donorCluster[i] <- fit.cluster[donor]
 NSEs[i] <- calcNSE(KmenasEU[donorCluster[i],], PkQ[i,])
}
classifications$reg.cluster <- donorCluster
classifications$NSE <- NSEs  # now is PUB
 head(classifications, 20)
```

```
        fit.cluster      NSE0 reg.cluster       NSE
6114500           6 0.8817538           6 0.8817538
6115500           6 0.7654732           6 0.7654732
6118010           6 0.9592077           6 0.9592077
6118015           6 0.8805138           6 0.8805138
6118020           6 0.8407020           6 0.8407020
6118030           6 0.8876026           6 0.8876026
6118060           6 0.8825192           6 0.8825192
6118070           6 0.9754464           6 0.9754464
6118150           6 0.9724077           6 0.9724077
6118165           6 0.9232526           6 0.9232526
6118175           6 0.9726351           6 0.9726351
6118205           6 0.8305336           6 0.8305336
6118210           6 0.8879485           6 0.8879485
6119010           5 0.3703362           5 0.3703362
6119020           3 0.3909119           5 0.2904111
6119030           5 0.4879748           3 0.1499807
6119040           5 0.7721574           5 0.7721574
6119050           6 0.7842414           5 0.7260961
6119110           5 0.8494703           5 0.8494703
6119120           5 0.8914300           5 0.8914300
```

Look at how many times the clusters are rightly assigned:

```
table(classifications$fit.cluster, classifications$reg.cluster)
```

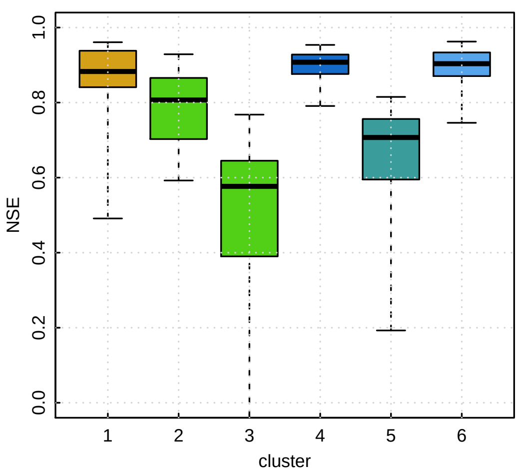
```
      1   2   3   4   5   6
  1  24   8   4   0   1   0
  2   6  37   8   0   0   0
  3   3  10  94   0  28   3
  4   0   0   2 131   9  17
  5   2   0  37  11 136  21
  6   0   0   2  17  19 133
```

```
round(prop.table(table(classifications$fit.cluster, classifications$reg.cluster), 1)*100)
```

```
     1  2  3  4  5  6
  1 65 22 11  0  3  0
  2 12 73 16  0  0  0
  3  2  7 68  0 20  2
  4  0  0  1 82  6 11
  5  1  0 18  5 66 10
  6  0  0  1 10 11 78
```

Not a too bad allocation (see the diagonals).

```
boxplot20(split(classifications$NSE, classifications$reg.cluster),
          ylim=c(0,1), xlab="cluster", ylab="NSE",
          boxfill=colori.stagioni(12)[monMaxKmenasEU])
 grid()
```



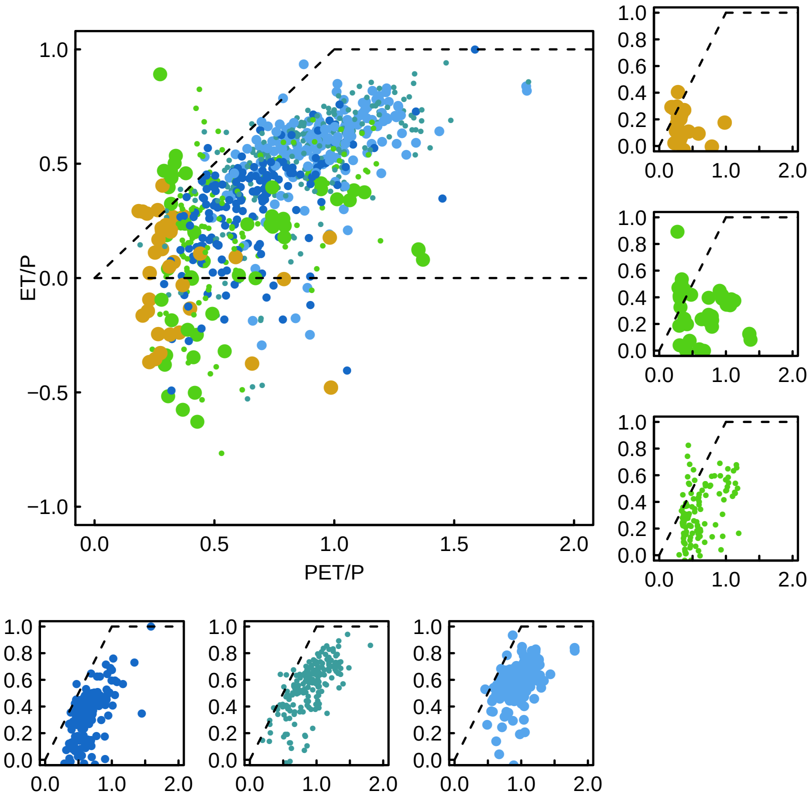
```
quantile(classifications$NSE)
```

```
           0%          25%          50%          75%         100%
-148.0032555    0.5343466    0.7952048    0.9215807    0.9912498
```

7 Can seasonality be useful for prediction of annual runoff in ungauged basins?

I just plot Budyko stratifying the points based on their seasonal regime (the 6 clusters discussed before).

```
layout(matrix(c(1,1,1,2,
                1,1,1,3,
                1,1,1,4,
                5,6,7,0), ncol=4, byrow=TRUE))
plot(PETovP, ETovP,
     xlim=c(0,2), ylim=c(-1,1), xlab="PET/P", ylab="ET/P", pch=16,
     col=colori.stagioni(12)[monMaxKmenasEU[classifications$fit.cluster]],
     cex=.7*rangeKmenasEU[classifications$fit.cluster])
 segments(x0=c(0,1), x1=c(1,4), y0=c(0,1), y1=c(1,1), lty=2)
 segments(x0=0, x1=4, y0=0, lty=2)
par(mar=c(1.5,1.5,1,1)+0.03)
for (i in 1:nclasters) {
 plot(PETovP[classifications$fit.cluster == i], ETovP[classifications$fit.cluster == i],
     xlim=c(0,2), ylim=c(0,1), xlab="", ylab="", pch=16,
     col=colori.stagioni(12)[monMaxKmenasEU[i]],
     cex=.7*rangeKmenasEU[i])
 segments(x0=c(0,1), x1=c(1,4), y0=c(0,1), y1=c(1,1), lty=2)
}
```



Now use the regionalised seasonal regime:

```
layout(matrix(c(1,1,1,2,
                1,1,1,3,
                1,1,1,4,
                5,6,7,0), ncol=4, byrow=TRUE))
plot(PETovP, ETovP,
     xlim=c(0,2), ylim=c(-1,1), xlab="PET/P", ylab="ET/P", pch=16,
     col=colori.stagioni(12)[monMaxKmenasEU[classifications$reg.cluster]],
     cex=.7*rangeKmenasEU[classifications$reg.cluster])
```

```
 segments(x0=c(0,1), x1=c(1,4), y0=c(0,1), y1=c(1,1), lty=2)
 segments(x0=0, x1=4, y0=0, lty=2)
par(mar=c(1.5,1.5,1,1)+0.03)
for (i in 1:nclusters) {
 plot(PETovP[classifications$reg.cluster == i], ETovP[classifications$reg.cluster == i],
     xlim=c(0,2), ylim=c(0,1), xlab="", ylab="", pch=16,
     col=colori.stagioni(12)[monMaxKmenasEU[i]],
     cex=.7*rangeKmenasEU[i])
 segments(x0=c(0,1), x1=c(1,4), y0=c(0,1), y1=c(1,1), lty=2)
}
```

well well well, not too bad. How can we use the groups to get better estimates of annual runoff?

8 Compare to the PUB book assessment

In the Level 2 Assessment of the PUB book (Blöschl et al., 2013) in Chapter 6 the Nash-Sutcliffe efficiency, the normalised error and the absolute normalised error in the estimation of the Pardé range is calculated. Here I report the NSE results for the nearest neighbor regionalisation made in Section 2.

```
NSEnn <- rep(NA, length(CatchmentsEU$code))
NEnn <- NSEnn
for (i in 1:length(CatchmentsEU$code)) {
 donor <- which(CatchmentsEU$code == spatNN$code[i])
 PkQdon <- PkQ[donor,]  # donor
 PkQrec <- PkQ[i,]      # rec
 NSEnn[i] <- calcNSE(PkQdon, PkQrec)
 rngPkQdon <- max(PkQdon) - min(PkQdon)
 rngPkQrec <- max(PkQrec) - min(PkQrec)
 NEnn[i] <- (rngPkQdon - rngPkQrec)/rngPkQrec
}
tabella <- data.frame(CatchmentsEU[,c("code", "area", "elev")], temp=MAT, aridity=PETovP,
                      NEnn=round(NEnn, 3), ANEnn=abs(round(NEnn, 3)), NSEnn=round(NSEnn, 3))
 head(tabella, 15)
```

```
      code   area   elev     temp  aridity   NEnn ANEnn NSEnn
1  6114500  606.0 330.00 13.16917 0.9323429 -0.317 0.317 0.823
2  6115500 2721.0  28.00 16.82083 1.8007552 -0.123 0.123 0.933
3  6118010  137.0  99.98 11.72917 0.9634795  0.215 0.215 0.921
4  6118015  315.0  88.26 11.55083 1.0160202 -0.032 0.032 0.985
5  6118020  118.0  68.00 11.70667 1.1278134 -0.050 0.050 0.991
6  6118030   30.2 119.01 11.50167 1.0447785  0.033 0.033 0.985
7  6118060  316.0  95.16 11.69167 0.9209745 -0.177 0.177 0.949
8  6118070   69.7  85.08 11.38083 0.7516084  0.320 0.320 0.897
9  6118150   43.0  91.10 11.39750 0.7619614  0.098 0.098 0.991
10 6118165  149.0 255.45 10.29917 1.0137340  0.597 0.597 0.639
11 6118175   44.0  73.86 11.34333 0.8283659 -0.089 0.089 0.992
12 6118205   81.5 103.34 11.31500 1.1380359  0.937 0.937 0.006
13 6118210  468.0  83.50 11.53167 1.1017651  0.026 0.026 0.984
14 6119010 2575.0 100.44  9.38500 0.5952235  0.221 0.221 0.491
15 6119020  488.0 277.79  6.78500 0.4776441 -0.132 0.132 0.944
```

```
aridity_class <- cut(tabella$aridity, breaks=c(-Inf,0.4,0.6,0.8,1,2,Inf))
temp_class <- cut(tabella$temp,  breaks=c(-Inf,3,6,8,10,12,Inf))
elev_class <- cut(tabella$elev, breaks=c(0,300,600,900,1200,1500,Inf))
area_class <- cut(tabella$area, breaks=c(0,50,100,500,1000,5000,Inf))
```

```
add_bxp <- function(performance="ANE", variable="area", classes, table) {
 # to add boxplot in a nice way
 perf <- table[, performance]
 boxplot20(split(perf, classes), add=TRUE, axes=FALSE)
}
```

Fig 6.28 at page 130 of the book:

```
layout(matrix(1:16, nrow=4, byrow=TRUE))
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NSE",
                     characteristic="Aridity", ylim=c(0,1),
                     main="Regression")
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NSE",
                     characteristic="Aridity", ylim=c(0,1),
                     main="Spatial_proximity")
 add_bxp(performance="NSEnn", variable="aridity", classes=aridity_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NSE",
                     characteristic="Aridity", ylim=c(0,1),
                     main="Geostatistics")
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NSE",
                     characteristic="Aridity", ylim=c(0,1),
                     main="Process_based")
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NSE",
                     characteristic="MAT", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NSE",
```

```
                     characteristic="MAT", ylim=c(0,1))
  add_bxp(performance="NSEnn", variable="temp", classes=temp_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NSE",
                     characteristic="MAT", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NSE",
                     characteristic="MAT", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NSE",
                     characteristic="Elevation", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NSE",
                     characteristic="Elevation", ylim=c(0,1))
  add_bxp(performance="NSEnn", variable="elev", classes=elev_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NSE",
                     characteristic="Elevation", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NSE",
                     characteristic="Elevation", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NSE",
                     characteristic="Area", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NSE",
                     characteristic="Area", ylim=c(0,1))
  add_bxp(performance="NSEnn", variable="area", classes=area_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NSE",
                     characteristic="Area", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NSE",
                     characteristic="Area", ylim=c(0,1))
```

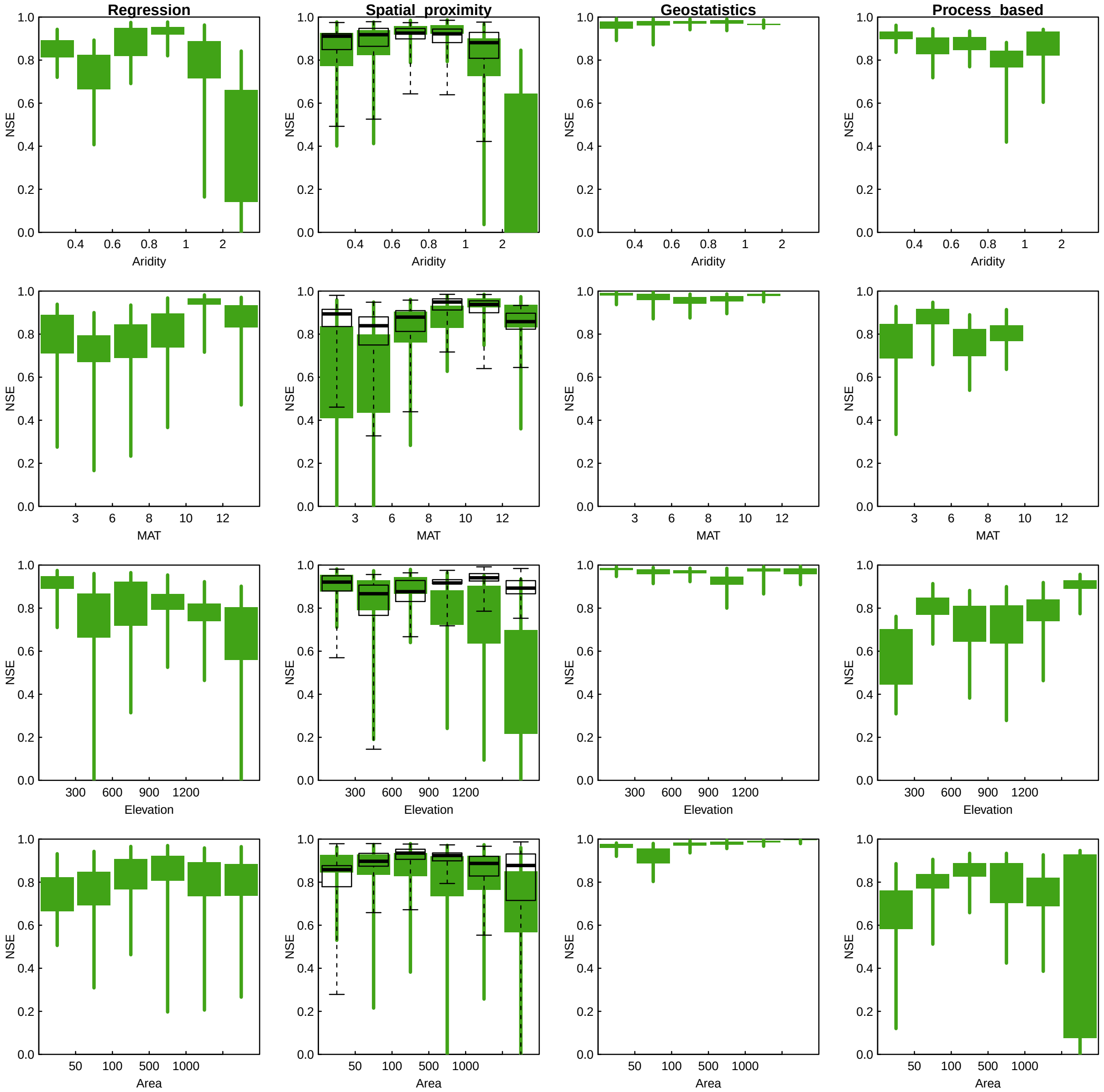


Fig 6.29 at page 131 of the book:

```
layout(matrix(1:16, nrow=4, byrow=TRUE))
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="ANE",
```

```
                     characteristic="Aridity", ylim=c(0.5,0),
                     main="Regression")
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="ANE",
                     characteristic="Aridity", ylim=c(0.5,0),
                     main="Spatial_proximity")
 add_bxp(performance="ANEnn", variable="aridity", classes=aridity_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="ANE",
                     characteristic="Aridity", ylim=c(0.5,0),
                     main="Geostatistics")
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="ANE",
                     characteristic="Aridity", ylim=c(0.5,0),
                     main="Process_based")
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="ANE",
                     characteristic="MAT", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="ANE",
                     characteristic="MAT", ylim=c(0.5,0))
 add_bxp(performance="ANEnn", variable="temp", classes=temp_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="ANE",
                     characteristic="MAT", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="ANE",
                     characteristic="MAT", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="ANE",
                     characteristic="Elevation", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="ANE",
                     characteristic="Elevation", ylim=c(0.5,0))
 add_bxp(performance="ANEnn", variable="elev", classes=elev_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="ANE",
                     characteristic="Elevation", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="ANE",
                     characteristic="Elevation", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="ANE",
                     characteristic="Area", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="ANE",
                     characteristic="Area", ylim=c(0.5,0))
 add_bxp(performance="ANEnn", variable="area", classes=area_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="ANE",
                     characteristic="Area", ylim=c(0.5,0))
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="ANE",
                     characteristic="Area", ylim=c(0.5,0))
```

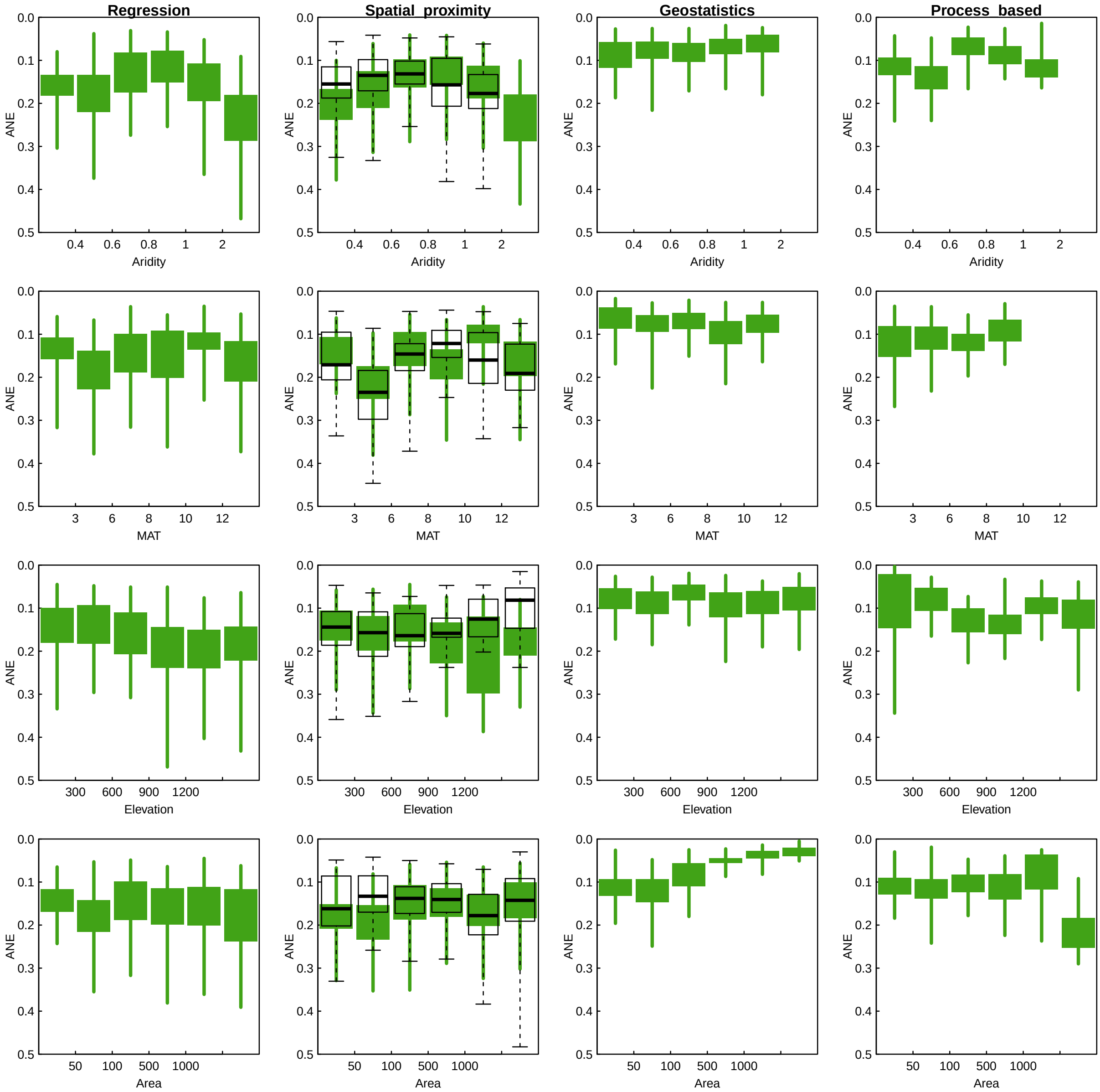
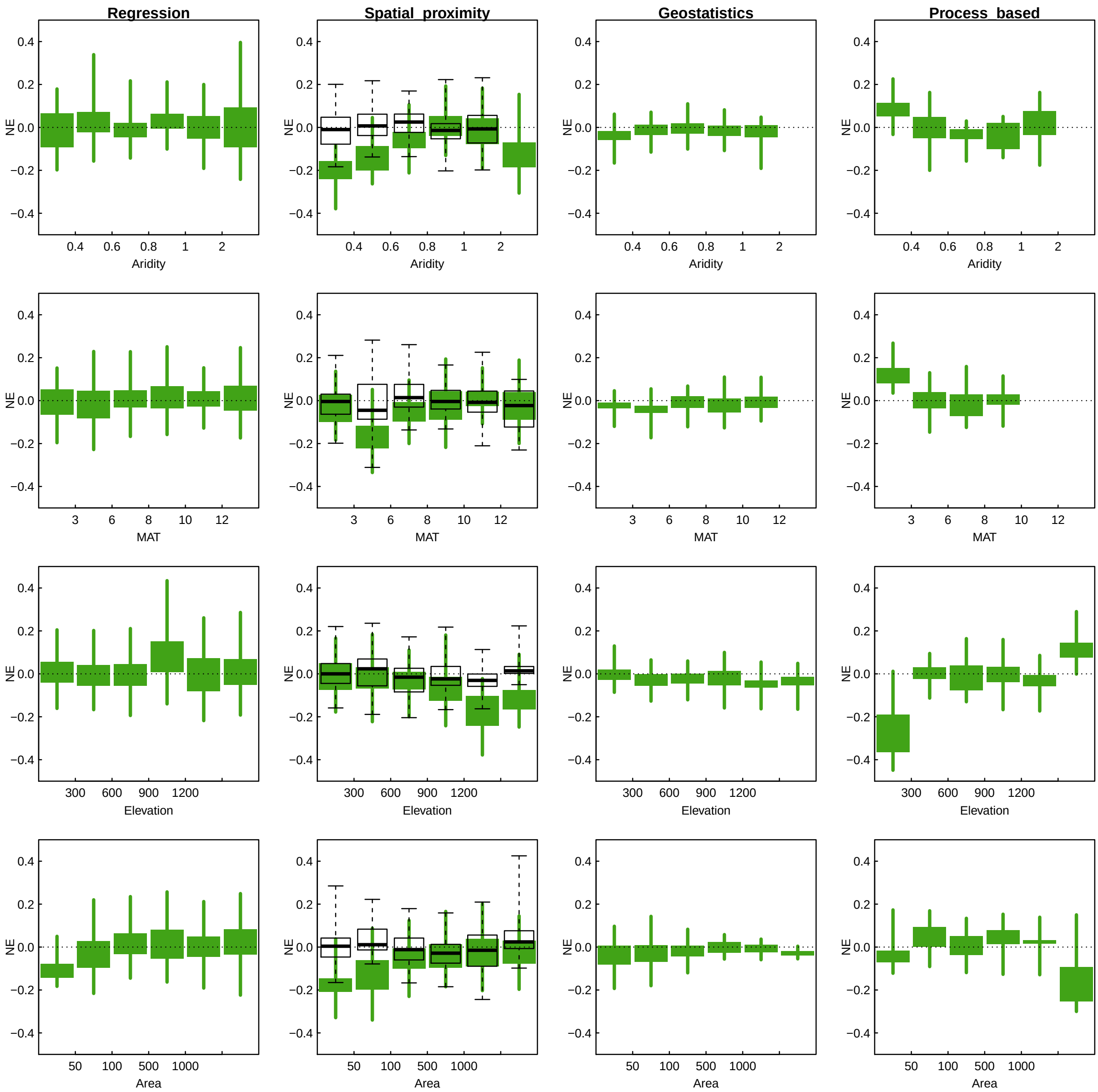


Fig 6.30 at page 132 of the book:

```
layout(matrix(1:16, nrow=4, byrow=TRUE))
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NE",
                     characteristic="Aridity", ylim=c(-.5,.5),
                     main="Regression"); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NE",
                     characteristic="Aridity", ylim=c(-.5,.5),
                     main="Spatial_proximity"); abline(h=0, lty=3)
 add_bxp(performance="NEnn", variable="aridity", classes=aridity_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NE",
                     characteristic="Aridity", ylim=c(-.5,.5),
                     main="Geostatistics"); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NE",
                     characteristic="Aridity", ylim=c(-.5,.5),
                     main="Process_based"); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NE",
                     characteristic="MAT", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NE",
                     characteristic="MAT", ylim=c(-.5,.5)); abline(h=0, lty=3)
 add_bxp(performance="NEnn", variable="temp", classes=temp_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NE",
                     characteristic="MAT", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NE",
                     characteristic="MAT", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NE",
                     characteristic="Elevation", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NE",
                     characteristic="Elevation", ylim=c(-.5,.5)); abline(h=0, lty=3)
 add_bxp(performance="NEnn", variable="elev", classes=elev_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NE",
```

```
                     characteristic="Elevation", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NE",
                     characteristic="Elevation", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Regression", performance="NE",
                     characteristic="Area", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Spatial_proximity", performance="NE",
                     characteristic="Area", ylim=c(-.5,.5)); abline(h=0, lty=3)
 add_bxp(performance="NEnn", variable="area", classes=area_class, table=tabella)
plotPUBfiguresLevel2(chapter=6, method="Geostatistics", performance="NE",
                     characteristic="Area", ylim=c(-.5,.5)); abline(h=0, lty=3)
plotPUBfiguresLevel2(chapter=6, method="Process_based", performance="NE",
                     characteristic="Area", ylim=c(-.5,.5)); abline(h=0, lty=3)
```



References

Blöschl, G., Sivapalan, M., Wagener, T., Viglione, A. and Savenije, H. (2013) *Runoff Prediction in Ungauged Basins: Synthesis Across Processes, Places and Scales*, University Press, Cambridge, 484 pages, ISBN:9781107028180.

Milly P.C.D. Climate, soil water storage and the average annual water balance, *Water Resources Research* **30**, 2143–56, doi:10.1029/94WR00586.

Weingartner, R., G. Blöschl, D. M. Hannah, D. G. Marks, J. Parajka, C. S. Pearson, M. Rogger, J. L. Salinas, E. Sauquet, R. Srikanthan, S. E. Thompson and A. Viglione (2013) Prediction of seasonal runoff in ungauged basins. In *Runoff Prediction in Ungauged Basins: Synthesis Across Processes, Places and Scales*, University Press, Cambridge, 102-134, ISBN:9781107028180.

Woods, R. (2003) The relative roles of climate, soil, vegetation and topography in determining seasonal and long-term catchment dynamics, *Advances in Water Resources*, **26**, 295–309, doi:10.1016/S0309-1708(02)00164-1.