Chapter 7: Prediction of flow duration curves in ungauged basins - an Italian example

Attilio Castellarin

1 Introduction

This Tutorial has been developed by Attilio Castellarin to illustrate the construction and regional prediction of Flow Duration Curves (FDC, see e.g., Vogel and Fennessey, 1994, 1995). A detailed literature review is available in Castellarin et al. (2013).

First of all load the library:

```
library(PUBexamples)
```

Then the data:

```
data(data4chapter7)
Descriptors
```

```
  Code        River                  Gauge   Area Elev  MAT    MAP   PET
1  801       Burano                   Foci  124.1 1702 12.0 1217.9 707.4
2  901  Candigliano             Acqualagna  613.8 1702 12.1 1160.1 707.4
3  902  Candigliano               Piobbico  186.7 1526 12.7 1163.1 735.4
4 1002      Metauro Barco di Bellaguardia 1043.6 1702 12.4 1120.0 721.1
5 1004      Metauro               Calmazzo  375.9 1384 12.7 1131.1 732.3
6 1701        Bosso                  Cagli  126.1 1526 12.2 1272.5 712.2
7 2101     Biscuvio               Piobbico   95.2 1526 13.0 1116.5 745.2
8 2201      Sentino            S. Vittore  263.6 1702 13.4  918.1 760.2
```

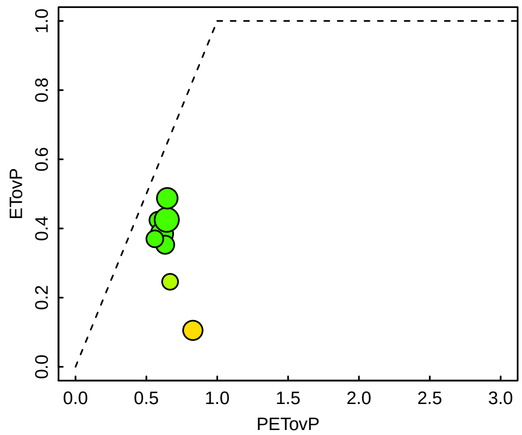
```
dailyQ[1:10, 1:20]
```

```
   Code Year  day1  day2  day3  day4  day5  day6  day7  day8  day9 day10 day11 day12 day13 day14 day15 day16 day17 day18
1   801 1924  3.39  3.39  3.77  7.98  7.19  5.73  5.17  8.14 11.34 11.04  9.67  8.52  7.38  6.61  5.88  5.12  4.53  3.63
2   801 1925  0.81  0.90  0.82  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.80  0.71  0.71  0.71
3   801 1927  7.11  6.14  5.67  6.99 14.43 12.04  8.00 17.43 17.34 10.22  7.20  6.27  6.20  6.18  5.88  6.14 10.16  8.53
4   801 1928  5.73  4.62  3.83  3.38  3.17  3.17  3.04  2.85  2.55  2.43  2.28  2.08  1.98  1.89  1.83  1.83  1.83  1.81
5   801 1929 17.90 24.80 13.30 11.50  8.26  6.19  5.28  4.70  4.20  3.80  3.49  3.21  3.13  3.06  2.85  2.32  2.19  2.04
6   801 1930  3.68  3.02  2.61  2.30  2.25  1.92  1.80  1.51  1.34  1.34  1.41  3.80  3.18  2.60  2.39  4.01 13.40  7.17
7   801 1931  6.99 11.40  6.75  5.31  4.42  4.17  3.79  3.50  2.95  2.77  2.41  2.25  2.19  2.19  2.19  2.19  2.19  2.17
8   901 1924 18.45 18.20 15.41 16.50 25.12 25.97 22.69 21.07 37.05 45.01 31.45 25.37 22.35 19.70 17.95 16.51 16.20 15.45
9   901 1925  7.40  6.92  6.70  6.39  6.35  6.04  6.48  6.08  5.91  5.65  5.43  5.17  4.95  4.95  4.08  5.13  5.21  5.13
10  901 1926 14.17 12.77 17.30 19.49 24.73 18.17 15.10 13.23 11.52 10.31  9.81  9.10  8.66  8.43 42.58 36.40 37.49 31.37
```

Budyko:

```
# mean annual runoff
MAR <- sapply(split(dailyQ[,-c(1,2)], dailyQ$Code), FUN=function(x){mean(as.matrix(x))}) # m3/s
MAR <- 365.25*24*3.6*MAR[-9]/Descriptors$Area
PETovP <- Descriptors$PET/Descriptors$MAP
ETovP <- (Descriptors$MAP - MAR)/Descriptors$MAP
```

```
colori <- rev(rainbow(10, start=0, end=.65, alpha=1))
plot(PETovP, ETovP, xlim=c(0,3), ylim=c(0,1), pch=21,
     bg=colori[round(10*PETovP)],
     cex=log10(Descriptors$Area))
 segments(x0=c(0,1), y0=c(0,1), x1=c(1,4), y1=c(1,1), lty=2)
```



2 Empirical Flow-Duration Curves

Using the available daily streamflow data, construct and represent on a semi-logarithmic plot the following empirical curves:

a) Period-of-Record Flow-Duration Curve (POR-FDC)

b) Annual FDC for a typical hydrologic year (median AFDC's)

c) Percentile AFDC's associated with a non-exceedance probability of 0.1 e 0.9 (one-in-ten-years dry and humid hydrological years, respectively)

The following code extracts data and produce plots on:

- Empirical Period-of-Record Flow-Duration Curve;
- Empirical Median Annual Flow Duration Curve;
- Percentile Annual Flow Duration Curve;

for the Target Site.

```
N <- 365
Target.Code <- 1701 # Target Site
```

The database contains complete series of daily streamflows (no missing values) in which data for Feb. 29th on leap years have been dropped (365 values per year):

```
dailyQ[1:10, 1:20]
```

```
   Code Year  day1  day2  day3  day4  day5  day6  day7  day8  day9 day10 day11 day12 day13 day14 day15 day16 day17 day18
1   801 1924  3.39  3.39  3.77  7.98  7.19  5.73  5.17  8.14 11.34 11.04  9.67  8.52  7.38  6.61  5.88  5.12  4.53  3.63
2   801 1925  0.81  0.90  0.82  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.81  0.80  0.71  0.71  0.71
3   801 1927  7.11  6.14  5.67  6.99 14.43 12.04  8.00 17.43 17.34 10.22  7.20  6.27  6.20  6.18  5.88  6.14 10.16  8.53
4   801 1928  5.73  4.62  3.83  3.38  3.17  3.17  3.04  2.85  2.55  2.43  2.28  2.08  1.98  1.89  1.83  1.83  1.83  1.81
5   801 1929 17.90 24.80 13.30 11.50  8.26  6.19  5.28  4.70  4.20  3.80  3.49  3.21  3.13  3.06  2.85  2.32  2.19  2.04
6   801 1930  3.68  3.02  2.61  2.30  2.25  1.92  1.80  1.51  1.34  1.34  1.41  3.80  3.18  2.60  2.39  4.01 13.40  7.17
7   801 1931  6.99 11.40  6.75  5.31  4.42  4.17  3.79  3.50  2.95  2.77  2.41  2.25  2.19  2.19  2.19  2.19  2.19  2.17
8   901 1924 18.45 18.20 15.41 16.50 25.12 25.97 22.69 21.07 37.05 45.01 31.45 25.37 22.35 19.70 17.95 16.51 16.20 15.45
9   901 1925  7.40  6.92  6.70  6.39  6.35  6.04  6.48  6.08  5.91  5.65  5.43  5.17  4.95  4.95  4.08  5.13  5.21  5.13
10  901 1926 14.17 12.77 17.30 19.49 24.73 18.17 15.10 13.23 11.52 10.31  9.81  9.10  8.66  8.43 42.58 36.40 37.49 31.37
```

```
M <- dailyQ[which(dailyQ[,1] == Target.Code),]  # Select the Target site data from the database
Anni <- unique(M[,2]) # Identify years (in Italian "Anni") with data
Nanni <- length(Anni) # no. of years
QMG <- matrix(as.matrix(M[,3:367]), Nanni, N) # Initialize matrix for storing Annual-FDC
# (Construction of empirical AFDC's)
# Reorganize obs. values ordering them in descending order
QMG <- -t(apply(-QMG, 1, sort)) # Sort each row in descending order
```

Compute percentiles AFDC's:

```
pvalues <- c(0.1,0.5,0.9); Np <- length(pvalues)
QPRC <- matrix(0, Np, N) # Initialization
# Computation
for (iD in 1:N) QPRC[,iD] <- as.vector(quantile(QMG[,iD], pvalues)) #Computation
```

Duration for AFDCs:

```
D_AFDC <- 1:N/(N + 1)
```

Reorganize obs values into a single vector (Construction of empirical Period of Record POR-FDC):

```
FDC_obs <- -sort(-c(QMG)) # Period-of-Record Flow Duration Curve
```

Duration for POR-FDCs:

```
N_POR <- length(FDC_obs)             # record length for POR_FDC
D_FDC <- 1:N_POR/(N_POR + 1)
```

Plot the results:

```
yy <- c(min(FDC_obs), max(FDC_obs)) # Axes limits
yt <- c(0.01,0.1,1,10,100,1000)  # Tick marks
```

Figure.1: Empirical POR-FDC's:

```
plot(D_FDC, FDC_obs, type="l", lty="dotted", col="black", lwd=3,
     log="y", yaxt="n", ylim=yy,
     main="Empirical Period-of-Record Flow-Duration Curves", cex.main=1, font.main=1,
     xlab="Duration", ylab=expression(paste("Discharge (",m^3,"/s)")))
 axis(2, at=yt)
 grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
      lwd=par("lwd"), equilogs=TRUE)
legend("topright", inset=.05, legend="POR-FDC", lty="dotted", lwd=3, col="black", bty="n")
```

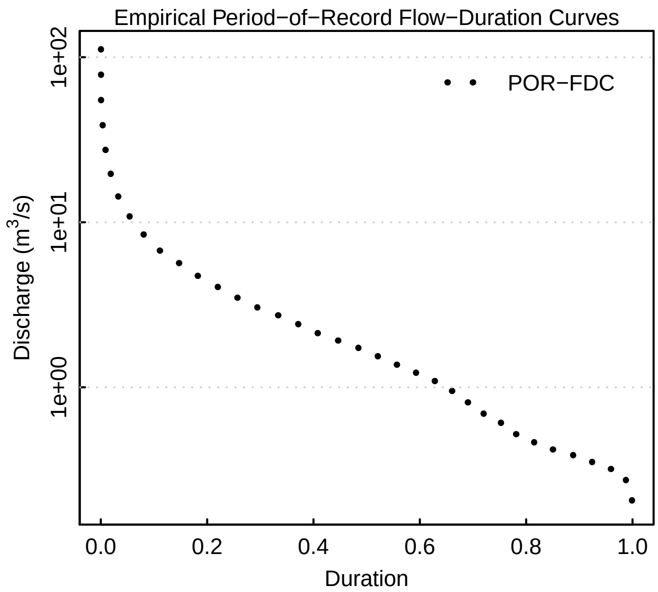


Figure.2: Empirical Median Annual FDC's:

```
# In the background: empirical AFDC's (gray)
plot(D_AFDC, QMG[1,], type="l", lty=1, col=rgb(.3,.3,.3),
     log="y", yaxt="n", ylim=yy,
     main="Empirical Annual Flow-Duration Curves", cex.main=1, font.main=1,
     xlab="Duration", ylab=expression(paste("Discharge (",m^3,"/s)")))
for (ire in 2:Nanni) lines(D_AFDC, QMG[ire,], lty=1, col=rgb(.3,.3,.3))
lines(D_AFDC, QPRC[2,], lty=1, col="black", lwd=3)
 axis(2, at=yt)
 grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
      lwd=par("lwd"), equilogs=TRUE)
legend("topright", inset=.05, legend=c("Empirical AFDC","Median AFDC"),
       bty="n", lwd=c(.75,3), col=c(rgb(.3,.3,.3), "black"))
```

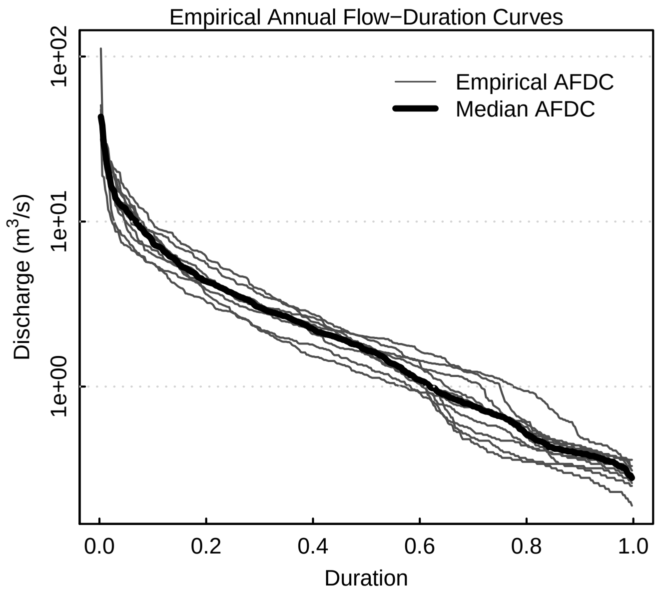


Figure.3: Empirical Percentile Annual FDC's:

```
# In the background: empirical AFDC's (gray)
plot(D_AFDC, QMG[1,], type="l", lty=1, col=rgb(.3,.3,.3),
     log="y", yaxt="n", ylim=yy,
     main="Empirical Annual Flow-Duration Curves", cex.main=1, font.main=1,
     xlab="Duration", ylab=expression(paste("Discharge (",m^3,"/s)")))
for (ire in 2:Nanni) lines(D_AFDC, QMG[ire,], lty=1, col=rgb(.3,.3,.3))
lines(D_AFDC, QPRC[1,], lty=1, col="red", lwd=3)
lines(D_AFDC, QPRC[3,], lty=1, col="blue", lwd=3)
 axis(2, at=yt)
 grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
      lwd=par("lwd"), equilogs=TRUE)
legend("topright", inset=.05, legend=c("Empirical AFDC","Percentile AFDC (dry year)","Percentile AFDC (wet year)"),
       bty="n", lwd=c(.75,3), col=c(rgb(.3,.3,.3), "red", "blue"))
```

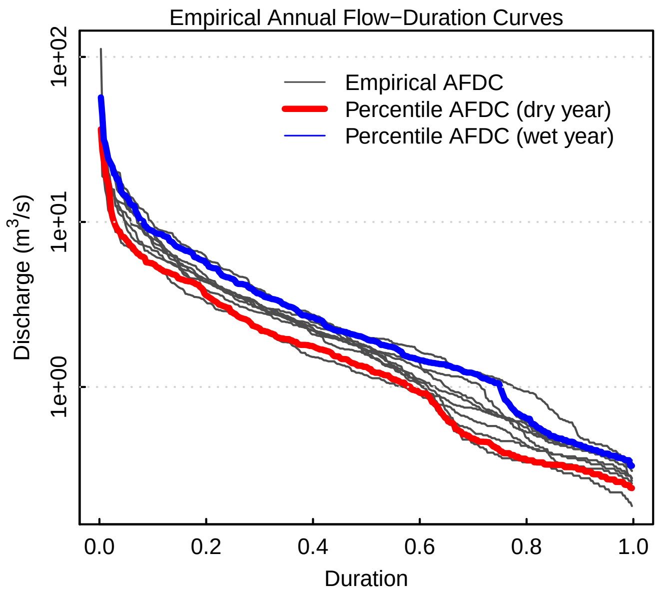
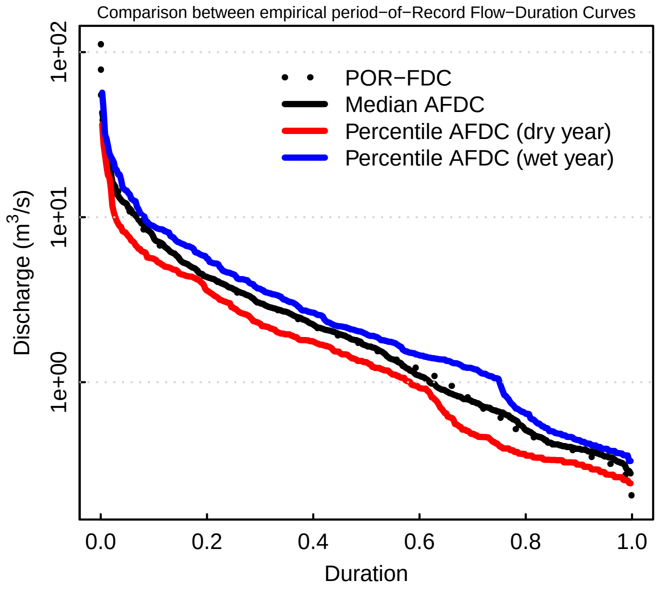


Figure.4: Comparison:

```
plot(D_FDC, FDC_obs, type="l", lty="dotted", col="black", lwd=3,
     log="y", yaxt="n", ylim=yy,
     main="Comparison between empirical period-of-Record Flow-Duration Curves", cex.main=.8, font.main=1,
     xlab="Duration", ylab=expression(paste("Discharge (",m^3,"/s)")))
lines(D_AFDC, QPRC[2,], lty=1, col="black", lwd=3)
lines(D_AFDC, QPRC[1,], lty=1, col="red", lwd=3)
lines(D_AFDC, QPRC[3,], lty=1, col="blue", lwd=3)
```

```
axis(2, at=yt)
grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
     lwd=par("lwd"), equilogs=TRUE)
legend("topright", inset=.05,
       legend=c("POR-FDC","Median AFDC","Percentile AFDC (dry year)","Percentile AFDC (wet year)"),
       lty=c("dotted","solid","solid","solid"), lwd=c(3,3,3,3), col=c("black","black","red","blue"), bty="n")
```



3 Regional model of Median AFDC

Considering the study region illustrated above, identify a regional model for predicting Median AFDC's in ungauged sites. To develop the model (1) adopt the graphical regional procedure and (2) assume that the *target site is ungauged* (i.e., discard all hydrometric information for this site).

The regional model to be developed consists of two components:

i) dimensionless Median AFDC, which reports the ratio between daily streamflows and long-term mean annual flow as a function of duration and is *valid for the Target Site (Region of Influence)*;

ii) a multiregression model that enables one to predict the long-term annual mean μ in ungauged sites as a function of relevant physiographic and climatic catchment descriptors, of the form:

$$\hat{\mu} = A_0 \omega_1^{A_1} \omega_2^{A_2} \dots \omega_n^{A_n} \cdot \varepsilon$$

where ω_i, with $i = 1, 2, \dots n$, are the explanatory variables (i.e., catchment descriptors) of the model and ε is the error term.

The adjusted Nash-Sutcliffe Efficiency measure, NSE_{adj}, may be used to guide the selection of the most suitable multiregression model:

$$NSE_{adj} = 1 - \left(\frac{N-1}{N-(p+1)} \right) (1 - NSE)$$

where

$$NSE = 1 - \frac{\sum_i (x_i - \hat{x}_i)^2}{\sum_i (x_i - \bar{x})^2}$$

and x_i are the empirical values, $\hat{x}_i$ are the estimated values, $\bar{x}$ is the empirical mean value, N is the number of catchments and p is the number of explanatory variables.

3.1 Dimensionless Median AFDC

The following code extracts data and computes the regional dimensionless Flow-Duration curve starting from regional data.

```
Target.Code
```

```
[1] 1701
```

Initialize a Variable to store all AFDC50:

```
Code <- c(801,901,902,1002,1004,1701,2101,2201)
AFDC50 <- matrix(0, length(Code), 365); rownames(AFDC50) <- Code
RegAFDC50 <- matrix(0, length(Code), 365); rownames(RegAFDC50) <- Code
```

Initialize the variable Mean Annual Flow (MAF):

```
MAF <- rep(0, length(Code)); names(MAF) <- Code
```

Calculate AFDC50 and MAF for all sites:

```
for (istaz in 1:length(Code)) #Loop on sites
{
  #Select the Target Site data from the database
  M <- dailyQ[which(dailyQ[,1] == Code[istaz]),]
  # Identifies the years ("Anni" in Italian) with data
  Anni <- unique(M[,2])
  Nanni <- length(Anni) #no. of years

  QMG <- matrix(as.matrix(M[,3:367]), Nanni, N) # Initialize matrix for storing Annual-FDC
  # Reorganize obs values ordering in decsending order
  # (Construction of empirical AFDC's)
  QMG <- -t(apply(-QMG, 1, sort)) # Sort each row in descending order

  # Store in memory AFDC50
  for (iD in 1:N) AFDC50[istaz, iD] <- as.vector(quantile(QMG[,iD], 0.5))
  # Store in memory MAF
  MAF[istaz] <- mean(QMG) #Average value of all observed flows
} #End Loop on sites
```

Dimensionless mean AFDC's:

```
for (istaz in 1:length(Code)) RegAFDC50[istaz,] <- AFDC50[istaz,]/MAF[istaz]
```

Target site dimensionless AFDC:

```
Target.RegAFDC50 <- RegAFDC50[which(Code == Target.Code),]
```

Regional sample without Target site:

```
RegAFDC50 <- RegAFDC50[-which(Code == Target.Code),]
```

ROI approach: Load catchment descriptors:

```
Descriptors
```

```
  Code       River                Gauge   Area Elev  MAT    MAP   PET
1  801      Burano                 Foci  124.1 1702 12.0 1217.9 707.4
2  901 Candigliano           Acqualagna  613.8 1702 12.1 1160.1 707.4
3  902 Candigliano             Piobbico  186.7 1526 12.7 1163.1 735.4
4 1002     Metauro Barco di Bellaguardia 1043.6 1702 12.4 1120.0 721.1
5 1004     Metauro             Calmazzo  375.9 1384 12.7 1131.1 732.3
6 1701       Bosso                Cagli  126.1 1526 12.2 1272.5 712.2
7 2101    Biscuvio             Piobbico   95.2 1526 13.0 1116.5 745.2
8 2201     Sentino           S. Vittore  263.6 1702 13.4  918.1 760.2
```

```
Attributes <- Descriptors[,c("Code","Area","MAT","MAP")]
```

Compute ROI distances with the Target Site:

```
for (icol in 2:4) Attributes[,icol] <- Attributes[,icol]/sd(Attributes[,icol])
```

Standardize by standard deviation:

```
Target.Attributes <- Attributes[which(Attributes$Code == Target.Code),]
Distance <- sqrt((Target.Attributes$MAT - Attributes$MAT)^2 +
                 (Target.Attributes$MAP - Attributes$MAP)^2 +
                 (Target.Attributes$Area - Attributes$Area)^2)
```

Drop site Target.Code and compute weights (weighted inverse distance):

```
exponent <- 3
Weights <- 1/Distance[-which(Code == Target.Code)]^exponent/sum(1/Distance[-which(Code == Target.Code)]^exponent)
```

Regional dimensionless AFDC50 (discarding site Target.Code):

```
Regional_Curve <- rep(0, N) # Initialize variable
for (iD in 1:N) Regional_Curve[iD] <- sum(RegAFDC50[,iD]*Weights)
```

Figure - Regional dimensionless AFDC's:

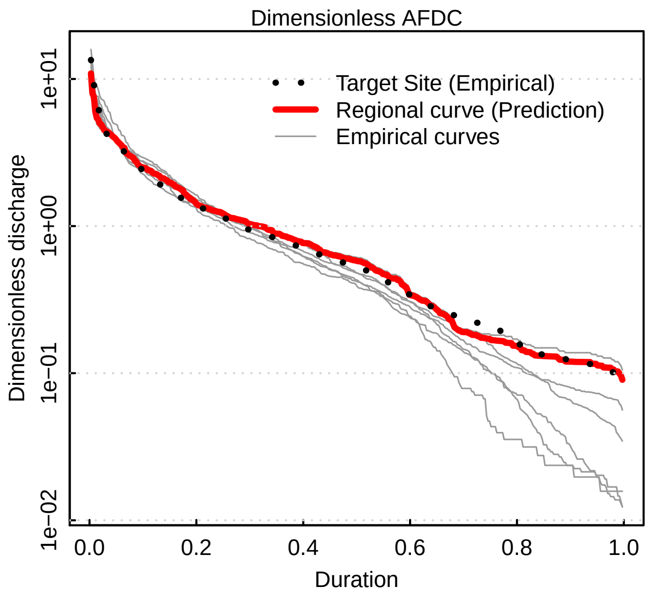
```
yy <- c(min(c(Target.RegAFDC50, RegAFDC50)),
        max(c(Target.RegAFDC50, RegAFDC50))) # Axes limits
yt <- c(0.01,0.1,1,10,100,1000) # Tick marks
```

Duration for AFDCs:

```
D_AFDC <- 1:N/(N + 1)
```

Sites' curves:

```
plot(D_AFDC, RegAFDC50[1,], type="l", lty=1, lwd=.75, col=rgb(.6,.6,.6),
     log="y", yaxt="n", ylim=yy,
     main="Dimensionless AFDC", cex.main=1, font.main=1,
     xlab="Duration", ylab="Dimensionless discharge")
for (isite in 2:(length(RegAFDC50[,1]) - 1)) {
 lines(D_AFDC, RegAFDC50[isite,], lty=1, lwd=.75, col=rgb(.6,.6,.6))
}
# Regional curve
lines(D_AFDC, Regional_Curve, lty=1, lwd=3, col=rgb(1,0,0))
lines(D_AFDC, Target.RegAFDC50, lty="dotted", lwd=3, col="black")
 axis(2,at=yt)
 grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
      lwd=par("lwd"), equilogs=TRUE)
legend("topright", inset=.05, legend=c("Target Site (Empirical)","Regional curve (Prediction)","Empirical curves"),
       lwd=c(3,3,.75), col=c("black","red",rgb(.6,.6,.6)), lty=c("dotted","solid","solid"), bty="n")
```



To save data for later utilizations:

```
# Median AFDC for the site of interest
dummy <- as.vector(AFDC50[which(Code == Target.Code),])
write(dummy, file="Target_AFDC50.txt", ncolumns=1)


# Mean Annual flow values
dummy <- matrix(c(Code, MAF), length(Code), 2)
write(t(dummy), 'Code_MAF.txt', ncolumns=2)


# Regional Curve
write(Regional_Curve, "Regional_Curve.txt")
```

3.2 Regional multiregression model

```
Target.Code
```

```
[1] 1701
```

```
A <- as.data.frame(matrix(c(Code, MAF), length(Code), 2))  # Mean Annual flow values
 colnames(A) <- c("Cod","MAF")   # Columns names
B <- Attributes # Catchment descriptors
 colnames(B) <- c("Cod","A","MAT","MAP") # Columns names
dimB <- dim(B) # Dimensions of B
```

Log-transformation of the data (dependent variable):

```
y <- log(A[,2])
y <- y[which(A$Cod != Target.Code)] # Discard Target Site
```

Log-trasformation of the data (explanatory variable):

```
x <- log(B[2:dimB[2]])
x <- x[which(B$Cod != Target.Code),] # Discard Target Site
```

Stepwise Regression Analysis

EXAMPLE: Model Area (first step):

```
M1.Area <- lm(y ~ x$A)
summary(M1.Area)
```

```
Call:
lm(formula = y ~ x$A)

Residuals:
        1         2         3         4         5         6         7 
-0.093994  0.064685  0.012343  0.008219 -0.186600  0.065112  0.130235 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.99373    0.04529   44.02 1.14e-07 ***
x$A          0.91165    0.05593   16.30 1.59e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1177 on 5 degrees of freedom
Multiple R-squared:  0.9815,	Adjusted R-squared:  0.9778 
F-statistic: 265.6 on 1 and 5 DF,  p-value: 1.586e-05
```

Checking the structure of the list:

```
str(M1.Area)
```

```
List of 12
 $ coefficients : Named num [1:2] 1.994 0.912
  ..- attr(*, "names")= chr [1:2] "(Intercept)" "x$A"
 $ residuals    : Named num [1:7] -0.09399 0.06469 0.01234 0.00822 -0.1866 ...
  ..- attr(*, "names")= chr [1:7] "1" "2" "3" "4" ...
 $ effects      : Named num [1:7] -4.9086 -1.9182 0.0435 -0.0107 -0.1758 ...
  ..- attr(*, "names")= chr [1:7] "(Intercept)" "x$A" "" "" ...
 $ rank         : int 2
 $ fitted.values: Named num [1:7] 1.11 2.57 1.48 3.05 2.12 ...
  ..- attr(*, "names")= chr [1:7] "1" "2" "3" "4" ...
 $ assign       : int [1:2] 0 1
 $ qr           :List of 5
  ..$ qr   : num [1:7, 1:2] -2.646 0.378 0.378 0.378 0.378 ...
  .. ..- attr(*, "dimnames")=List of 2
  .. .. ..$ : chr [1:7] "1" "2" "3" "4" ...
  .. .. ..$ : chr [1:2] "(Intercept)" "x$A"
  .. ..- attr(*, "assign")= int [1:2] 0 1
  ..$ qraux: num [1:2] 1.38 1.48
  ..$ pivot: int [1:2] 1 2
  ..$ tol  : num 1e-07
  ..$ rank : int 2
  ..- attr(*, "class")= chr "qr"
 $ df.residual  : int 5
 $ xlevels      : Named list()
 $ call         : language lm(formula = y ~ x$A)
 $ terms        :Classes 'terms', 'formula' length 3 y ~ x$A
  .. ..- attr(*, "variables")= language list(y, x$A)
  .. ..- attr(*, "factors")= int [1:2, 1] 0 1
  .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. ..$ : chr [1:2] "y" "x$A"
  .. .. .. ..$ : chr "x$A"
  .. ..- attr(*, "term.labels")= chr "x$A"
  .. ..- attr(*, "order")= int 1
  .. ..- attr(*, "intercept")= int 1
  .. ..- attr(*, "response")= int 1
  .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
  .. ..- attr(*, "predvars")= language list(y, x$A)
  .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
  .. .. ..- attr(*, "names")= chr [1:2] "y" "x$A"
 $ model        :'data.frame':	7 obs. of  2 variables:
  ..$ y  : num [1:7] 1.01 2.63 1.49 3.06 1.93 ...
  ..$ x$A: num [1:7] -0.971 0.628 -0.562 1.159 0.137 ...
  ..- attr(*, "terms")=Classes 'terms', 'formula' length 3 y ~ x$A
```

```
  .. .. ..- attr(*, "variables")= language list(y, x$A)
  .. .. ..- attr(*, "factors")= int [1:2, 1] 0 1
  .. .. .. ..- attr(*, "dimnames")=List of 2
  .. .. .. .. ..$ : chr [1:2] "y" "x$A"
  .. .. .. .. ..$ : chr "x$A"
  .. .. ..- attr(*, "term.labels")= chr "x$A"
  .. .. ..- attr(*, "order")= int 1
  .. .. ..- attr(*, "intercept")= int 1
  .. .. ..- attr(*, "response")= int 1
  .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
  .. .. ..- attr(*, "predvars")= language list(y, x$A)
  .. .. ..- attr(*, "dataClasses")= Named chr [1:2] "numeric" "numeric"
  .. .. .. ..- attr(*, "names")= chr [1:2] "y" "x$A"
 - attr(*, "class")= chr "lm"
```

```
M1.Area[[1]][1]
```

```
(Intercept)
   1.993726
```

```
M1.Area$coefficients[1]
```

```
(Intercept)
   1.993726
```

```
summary(M1.Area)[[1]]
```

```
lm(formula = y ~ x$A)
```

EXAMPLE: Model MAP (second step):

```
M1.MAP <- lm(y ~ x$MAP)
summary(M1.MAP)
```

```
Call:
lm(formula = y ~ x$MAP)

Residuals:
       1        2        3        4        5        6        7 
-0.71913  0.83042 -0.30341  1.20972  0.09749 -0.92057 -0.19451 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    5.107      9.181   0.556    0.602
x$MAP         -1.367      3.858  -0.354    0.738

Residual standard error: 0.8552 on 5 degrees of freedom
Multiple R-squared:  0.0245,	Adjusted R-squared:  -0.1706 
F-statistic: 0.1256 on 1 and 5 DF,  p-value: 0.7375
```

EXAMPLE: Model MAT (third step):

```
M1.MAT <- lm(y ~ x$MAT)
summary(M1.MAT)
```

```
Call:
lm(formula = y ~ x$MAT)

Residuals:
      1       2       3       4       5       6       7 
-1.0363  0.6127 -0.3324  1.1374  0.1066 -0.8010  0.3130 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   14.827     28.969   0.512    0.631
x$MAT         -3.971      8.868  -0.448    0.673

Residual standard error: 0.849 on 5 degrees of freedom
Multiple R-squared:  0.03856,	Adjusted R-squared:  -0.1537 
F-statistic: 0.2005 on 1 and 5 DF,  p-value: 0.673
```

Area in the model with one explanatory variable

EXAMPLE: Model with Area and MAP (fourth step):

```
M2.A.MAP <- lm(y ~ x$A + x$MAP)
summary(M2.A.MAP)
```

```
Call:
lm(formula = y ~ x$A + x$MAP)

Residuals:
        1         2         3         4         5         6         7 
-0.038496  0.096397  0.038990  0.019138 -0.174846  0.060202 -0.001385 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.60372    1.16501   3.093   0.0365 *  
x$A          0.90593    0.05161  17.555 6.18e-05 ***
x$MAP       -0.67745    0.48990  -1.383   0.2389    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1082 on 4 degrees of freedom
Multiple R-squared:  0.9875,	Adjusted R-squared:  0.9813 
F-statistic:   158 on 2 and 4 DF,  p-value: 0.0001562
```

EXAMPLE: Model with Area and MAT (fifth step):

```
M2.A.MAT <- lm(y ~ x$A + x$MAT)
summary(M2.A.MAT)
```

```
Call:
lm(formula = y ~ x$A + x$MAT)

Residuals:
        1         2         3         4         5         6         7
-0.021883  0.103826  0.009085  0.009750 -0.199836  0.042815  0.056243

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.01083    4.18700  -0.480    0.656
x$A          0.92591    0.05835  15.867 9.22e-05 ***
x$MAT        1.22668    1.28249   0.956    0.393
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1187 on 4 degrees of freedom
Multiple R-squared:  0.985,	Adjusted R-squared:  0.9774
F-statistic:   131 on 2 and 4 DF,  p-value: 0.0002261
```

STOP - Area is the only descriptor worth including!

Regional Model performance

Empirical values

```
EmpMuQ <- A[,2]
EmpMuQ <- EmpMuQ[which(A$Cod != Target.Code)] # Discard Target site
```

Regional estimates

```
RegMuQ <- exp(M1.Area$coefficients[1] + M1.Area$coefficients[2]*x$A)
```

Adjusted NSE index

```
NSE <- 1 - sum((EmpMuQ - RegMuQ)^2)/sum((EmpMuQ - mean(EmpMuQ))^2)
N <- length(x$A); p <- (length(M1.Area$coefficients) - 1)
NSEadj <- 1 - (N-1)/(N - (p + 1))*(1 - NSE)
print("NSE and Adjusted NSE:")
```

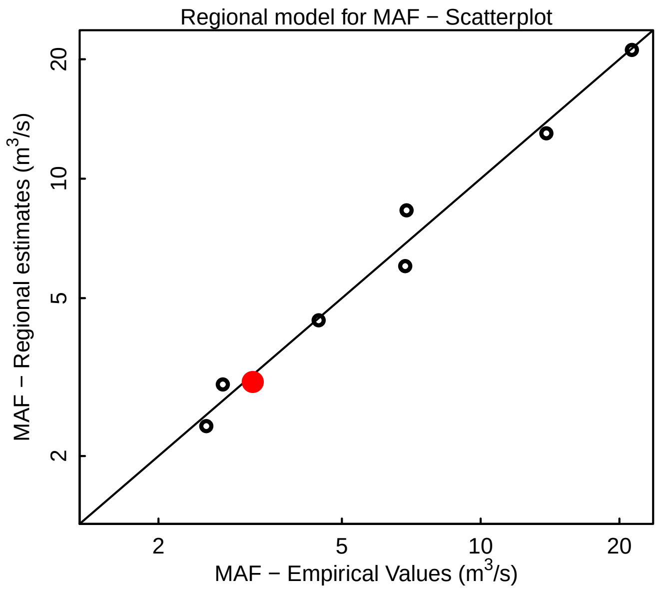
```
[1] "NSE and Adjusted NSE:"
```

```
print(c(NSE, NSEadj))
```

```
[1] 0.9872642 0.9847170
```

Figure - Scatter-plot:

```
plot(EmpMuQ, RegMuQ, type="p", pch=1, lwd=2,
     log="xy",
     xlim=c(1.5, max(A[,2])),
     ylim=c(1.5, max(A[,2])),
     main="Regional model for MAF - Scatterplot", cex.main=1, font.main=1,
     xlab=expression(paste("MAF - Empirical Values (",m^3,"/s)")),
     ylab=expression(paste("MAF - Regional estimates (",m^3,"/s)")))
#Perfect fit
abline(a=0, b=1)
# Plot in RED regional estimate for Target site
# Log-transformed catchment descriptor (Area in this case):
Target.RegMuQ <- exp(M1.Area$coefficients[1] +
                     M1.Area$coefficients[2]*log(B$A[which(B$Cod == Target.Code)]))
# Plot cross-validated estimate
points(A[which(A[,1] == Target.Code),2], Target.RegMuQ, col="red", pch=19, cex=2)
```



Save data for later utilizations:

```
write(Target.RegMuQ, "Target_RegMuQ.txt")
```

4 Reliability of the regional model

Graphically compare the empirical Median AFDC (Section 2 of the tutorial) with its regional prediction (Section 3 of the tutorial) for the Target Site.

Graphical comparison of empirical and regional median AFDC:

```
RegMuQ <- Target.RegMuQ                              # Regional estimate of MAF for the site of interest
RegAFDC <- Regional_Curve                            # Regional dimensionless median AFDC
EmpAFDC <- as.vector(AFDC50[which(Code == Target.Code),])  # Empirical median AFDC for the site of interest
```

Duration:

```
N <- 365
D_AFDC <- 1:N/(N + 1)
```

Figure.1 - Comparison of Median AFDC's:

```
# Plot empirical AFDC
plot(D_AFDC, EmpAFDC, type="l", lty=1, col="black", lwd=2.75,
     log="y", yaxt="n",
     main="Reliability of the regional model",
     sub="Median AFDC",
     xlab="Duration", ylab=expression(paste("Discharge (",m^3,"/s)")))
axis(2, at=c(0.01,0.1,1,10,100,1000))
# Plot regional prediction
lines(D_AFDC, RegAFDC*RegMuQ, lty=1, col="red", lwd=2.75)
 grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
      lwd=par("lwd"), equilogs=TRUE)
legend("topright", inset=.05, legend=c("Empirical","Predicted"),
       bty="n", lwd=c(2.75,2.75), col=c("black","red"))
```

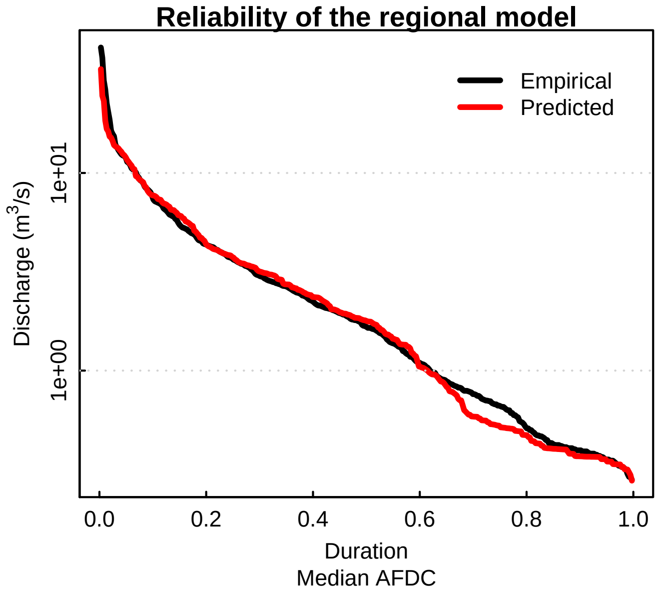
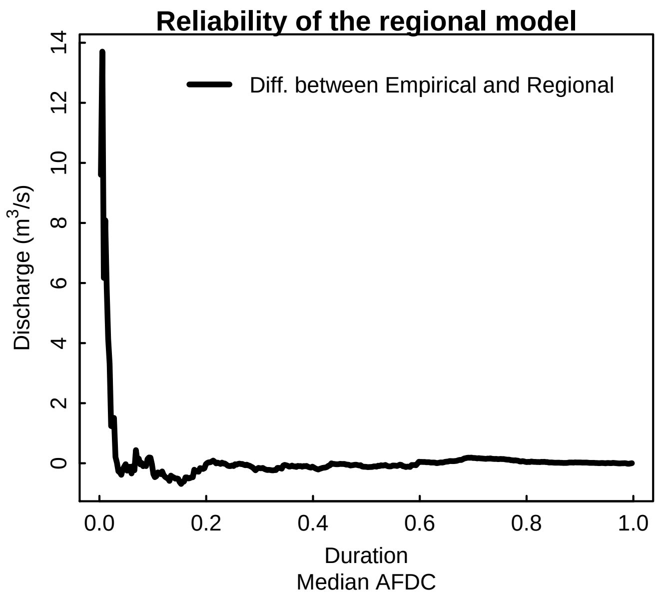


Figure.2 - Residuals:

```
# Plot empirical Difference (Emp - Reg)
plot(D_AFDC, EmpAFDC - RegAFDC*RegMuQ,
     type="l", lty=1, col="black", lwd=2.75,
     main="Reliability of the regional model",
     sub="Median AFDC",
     xlab="Duration", ylab=expression(paste("Discharge (",m^3,"/s)")))
legend("topright", inset=.05, legend="Diff. between Empirical and Regional",
       bty="n", lwd=2.75, col="black")
```



5 Jackknife cross-validation

Try different target sites / flow-duration curves.

The observed median flow duration curves are:

```
AFDC50[,1:10]
```

```
      [,1]    [,2]    [,3]    [,4]    [,5]  [,6]   [,7]   [,8]   [,9]  [,10]
801  28.78  20.250  19.580  15.300  13.600 13.37 12.680 12.650 12.310 11.690
901 180.50 134.000 122.500 112.000 100.045 91.53 88.645 86.210 77.910 75.765
902  57.65  49.500  44.205  34.675  34.125 33.84 30.440 29.495 27.645 25.840
```

```
1002 241.50 204.000 170.000 159.500 154.000 135.00 126.000 114.000 113.000 108.500
1004  83.20  76.100  65.150  52.750  50.630  47.51  42.000  39.885  35.560  34.555
1701  43.15  38.290  29.350  26.520  22.500  20.41  18.595  16.345  15.775  15.350
2101  40.30  32.950  25.730  23.750  22.890  21.49  19.590  18.570  18.350  17.990
2201  79.95  64.655  53.985  51.320  49.590  43.74  41.565  36.250  35.300  34.000
```

```
summary(t(AFDC50))
```

```
     801             901             902              1002             1004             1701             2101             2201
 Min.   : 0.290   Min.   :  0.78   Min.   : 0.055   Min.   :  0.735   Min.   : 0.085   Min.   : 0.280   Min.   : 0.040   Min.   : 0.355
 1st Qu.: 0.510   1st Qu.:  2.17   1st Qu.: 0.455   1st Qu.:  2.835   1st Qu.: 0.720   1st Qu.: 0.665   1st Qu.: 0.120   1st Qu.: 1.190
 Median : 1.690   Median :  6.66   Median : 1.915   Median : 10.245   Median : 2.900   Median : 1.665   Median : 1.000   Median : 3.410
 Mean   : 2.778   Mean   : 14.08   Mean   : 4.543   Mean   : 20.153   Mean   : 6.952   Mean   : 3.194   Mean   : 2.609   Mean   : 6.718
 3rd Qu.: 3.380   3rd Qu.: 17.34   3rd Qu.: 4.900   3rd Qu.: 24.400   3rd Qu.: 8.495   3rd Qu.: 3.635   3rd Qu.: 2.640   3rd Qu.: 7.780
 Max.   :28.780   Max.   :180.50   Max.   :57.650   Max.   :241.500   Max.   :83.200   Max.   :43.150   Max.   :40.300   Max.   :79.950
```

and, normalised with the mean:

```
MAF
```

```
     801       901       902      1002      1004      1701      2101      2201
 2.758438 13.883379  4.451914 21.286991  6.905945  3.203266  2.539797  6.859820
```

```
normAFDC50 <- AFDC50/matrix(MAF, nrow=dim(AFDC50)[1], ncol=dim(AFDC50)[2])
normAFDC50[,1:10]
```

```
         [,1]      [,2]      [,3]     [,4]     [,5]     [,6]     [,7]     [,8]     [,9]    [,10]
801  10.43344  7.341110  7.098219 5.546617 4.930326 4.846945 4.596804 4.585928 4.462670 4.237905
901  13.00116  9.651829  8.823501 8.067201 7.206099 6.592776 6.384973 6.209584 5.611746 5.457245
902  12.94949 11.118812  9.929436 7.788784 7.665242 7.601224 6.837508 6.625240 6.209688 5.804245
1002 11.34496  9.583318  7.986098 7.492839 7.234466 6.341902 5.919108 5.355384 5.308407 5.097010
1004 12.04759 11.019491  9.433901 7.638346 7.331364 6.879580 6.081716 5.775459 5.149187 5.003660
1701 13.47063 11.953426  9.162524 8.279051 7.024082 6.371622 5.805013 5.102605 4.924662 4.791985
2101 15.86741 12.973476 10.130730 9.351140 9.012530 8.461305 7.713214 7.311607 7.224986 7.083243
2201 11.65483  9.425175  7.869740 7.481246 7.229053 6.376261 6.059197 5.284395 5.145908 4.956399
```

```
summary(t(normAFDC50))
```

```
     801              901               902               1002              1004              1701              2101              2201
 Min.   : 0.1051   Min.   : 0.05618   Min.   : 0.01235   Min.   : 0.03453   Min.   : 0.01231   Min.   : 0.08741   Min.   : 0.01575   Min.   : 0.05175
 1st Qu.: 0.1849   1st Qu.: 0.15630   1st Qu.: 0.10220   1st Qu.: 0.13318   1st Qu.: 0.10426   1st Qu.: 0.20760   1st Qu.: 0.04725   1st Qu.: 0.17347
 Median : 0.6127   Median : 0.47971   Median : 0.43015   Median : 0.48128   Median : 0.41993   Median : 0.51978   Median : 0.39373   Median : 0.49710
 Mean   : 1.0070   Mean   : 1.01443   Mean   : 1.02043   Mean   : 0.94672   Mean   : 1.00673   Mean   : 0.99701   Mean   : 1.02735   Mean   : 0.97930
 3rd Qu.: 1.2253   3rd Qu.: 1.24862   3rd Qu.: 1.10065   3rd Qu.: 1.14624   3rd Qu.: 1.23010   3rd Qu.: 1.13478   3rd Qu.: 1.03945   3rd Qu.: 1.13414
 Max.   :10.4334   Max.   :13.00116   Max.   :12.94948   Max.   :11.34496   Max.   :12.04759   Max.   :13.47063   Max.   :15.86741   Max.   :11.65483
```

Redo the above procedure in a loop to estimate the regional ones in cross-validation:

```
predMAFcv <- rep(NA, length(MAF))
 names(predMAFcv) <- names(MAF)
predAFDC50cv <- matrix(NA, nrow=dim(AFDC50)[1], ncol=dim(AFDC50)[2])
 rownames(predAFDC50cv) <- rownames(AFDC50)
prednormAFDC50cv <- predAFDC50cv
for (i in 1:length(Descriptors$Code)) {
 Target.Code <- Descriptors$Code[i]  # Target Site
 # Dimensionless mean AFDC's
 Code <- Descriptors$Code
 RegAFDC50 <- matrix(0, length(Code), 365); rownames(RegAFDC50) <- Code
 for (istaz in 1:length(Code)) RegAFDC50[istaz,] <- AFDC50[istaz,]/MAF[istaz]
 # Target site dimensionless AFDC
 Target.RegAFDC50 <- RegAFDC50[which(Code == Target.Code),]
 # Regional sample without Target site
 RegAFDC50 <- RegAFDC50[-which(Code == Target.Code),]
 # Compute ROI distances with the Target Site
 Target.Attributes <- Attributes[which(Attributes$Code == Target.Code),]
 Distance <- sqrt((Target.Attributes$MAT - Attributes$MAT)^2 +
                  (Target.Attributes$MAP - Attributes$MAP)^2 +
                  (Target.Attributes$A - Attributes$A)^2)
 # Drop site Target.Code and compute weights (weighted inverse distance):
 # (exponent=3)
 Weights <- 1/Distance[-which(Code == Target.Code)]^exponent/sum(1/Distance[-which(Code == Target.Code)]^exponent)
 # Regional dimensionless AFDC50 (discarding site Target.Code):
 Regional_Curve <- rep(0, 365) # Initialize variable
 for (iD in 1:365) Regional_Curve[iD] <- sum(RegAFDC50[,iD]*Weights)
 prednormAFDC50cv[i,] <- Regional_Curve
 # Regional multiregression model for the mean
 y <- log(A[,2])
 y <- y[which(A$Cod != Target.Code)] # Discard Target Site
 x <- log(B[2:dimB[2]])
 x <- x[which(B$Cod != Target.Code),] # Discard Target Site
 M1.Area <- lm(y ~ x$A)  # as before I use Area only
 # Regional Models
 RegMuQ <- exp(M1.Area$coefficients[1] +
               M1.Area$coefficients[2]*log(B$A[which(B$Cod == Target.Code)]))
 predMAFcv[i] <- RegMuQ
 predAFDC50cv[i,] <- RegMuQ*Regional_Curve
}
```

```
predAFDC50cv[,1:10]
```

```
        [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]      [,8]      [,9]     [,10]
801   42.18721  36.95785  29.03551  25.92845  22.48542  20.57715  18.78359  16.83630  16.10389  15.62119
901  153.12897 129.31399 110.36224  93.81100  88.11972  82.17520  75.03473  70.48151  66.98351  64.31846
902   58.33493  50.52541  41.96554  35.80058  34.07762  31.92782  28.71557  27.19922  25.35952  24.70369
1002 266.27602 211.98662 186.84571 163.72207 150.19253 139.58939 130.83302 125.53471 115.34192 111.62949
1004 115.00070  96.63737  84.02224  69.05042  66.84720  64.96062  58.96427  56.70187  53.69427  50.85574
1701  33.54607  24.58548  23.17885  18.42871  16.68320  16.26754  15.28509  15.10530  14.54911  13.84049
2101  29.52861  25.58201  22.47023  18.04231  17.47555  16.92633  15.23147  14.60219  13.54978  12.82953
2201  80.85957  67.80777  56.42431  48.94530  46.66065  43.94919  39.99121  37.98199  36.18902  35.01883
```

```
summary(t(predAFDC50cv))
```

```
      801               901              902                1002               1004              1701               2101               2201
 Min.   : 0.2391   Min.   :  0.575   Min.   : 0.09974   Min.   :  0.9678   Min.   :  0.1859   Min.   : 0.2774   Min.   : 0.04588   Min.   : 0.1585
 1st Qu.: 0.5986   1st Qu.:  1.739   1st Qu.: 0.43393   1st Qu.:  2.9311   1st Qu.:  0.8976   1st Qu.: 0.5263   1st Qu.: 0.26259   1st Qu.: 0.5815
 Median : 1.5920   Median :  6.150   Median : 1.89300   Median :  9.7809   Median :  3.7805   Median : 1.7852   Median : 1.02907   Median : 2.5882
 Mean   : 3.1536   Mean   : 12.616   Mean   : 4.47615   Mean   : 21.0335   Mean   :  8.7993   Mean   : 3.0998   Mean   : 2.37645   Mean   : 6.0180
 3rd Qu.: 3.5985   3rd Qu.: 14.835   3rd Qu.: 5.16890   3rd Qu.: 25.1309   3rd Qu.:  9.5596   3rd Qu.: 3.7272   3rd Qu.: 2.68531   3rd Qu.: 6.7097
 Max.   :42.1872   Max.   :153.129   Max.   :58.33493   Max.   :266.2760   Max.   :115.0007   Max.   :33.5461   Max.   :29.52861   Max.   :80.8596
```

```
predMAFcv
```

```
      801       901       902      1002      1004      1701      2101      2201
 3.153424 12.691078  4.423709 20.790911  8.642742  3.074781  2.345096  5.944231
```

```
prednormAFDC50cv[,1:10]
```

```
        [,1]      [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]     [,9]    [,10]
801  13.37822 11.719909 9.207614 8.222317 7.130478 6.525333 5.956568 5.339054 5.106793 4.953723
901  12.06588 10.189363 8.696050 7.391886 6.943439 6.475037 5.912400 5.553627 5.278000 5.068006
902  13.18688 11.421505 9.486505 8.092889 7.703404 7.217434 6.491289 6.148511 5.732638 5.584385
1002 12.80733 10.196120 8.986894 7.874694 7.223952 6.713962 6.292799 6.037961 5.547709 5.369149
1004 13.30604 11.181333 9.721711 7.989412 7.734490 7.516205 6.822404 6.560635 6.212644 5.884214
1701 10.91007  7.995849 7.538374 5.993505 5.425817 5.290635 4.971117 4.912642 4.731756 4.501294
2101 12.59164 10.908727 9.581797 7.693632 7.451953 7.217755 6.495031 6.226691 5.777922 5.470790
2201 13.60303 11.407323 9.492281 8.234084 7.849736 7.393587 6.727735 6.389723 6.088091 5.891229
```

```
summary(t(prednormAFDC50cv))
```

```
      801                901               902                1002               1004               1701               2101               2201
 Min.   : 0.07581   Min.   : 0.0453   Min.   : 0.02255   Min.   : 0.04655   Min.   : 0.02151   Min.   : 0.09021   Min.   : 0.01956   Min.   : 0.02666
 1st Qu.: 0.18982   1st Qu.: 0.1371   1st Qu.: 0.09809   1st Qu.: 0.14098   1st Qu.: 0.10385   1st Qu.: 0.17118   1st Qu.: 0.11197   1st Qu.: 0.09782
 Median : 0.50484   Median : 0.4846   Median : 0.42792   Median : 0.47044   Median : 0.43742   Median : 0.58058   Median : 0.43882   Median : 0.43541
 Mean   : 1.00004   Mean   : 0.9940   Mean   : 1.01185   Mean   : 1.01167   Mean   : 1.01811   Mean   : 1.00815   Mean   : 1.01337   Mean   : 1.01242
 3rd Qu.: 1.14115   3rd Qu.: 1.1690   3rd Qu.: 1.16846   3rd Qu.: 1.20874   3rd Qu.: 1.10609   3rd Qu.: 1.21220   3rd Qu.: 1.14508   3rd Qu.: 1.12878
 Max.   :13.37822   Max.   :12.0659   Max.   :13.18688   Max.   :12.80733   Max.   :13.30604   Max.   :10.91007   Max.   :12.59164   Max.   :13.60303
```

Figure.1 - Comparison of Median AFDC's:

```
layout(matrix(1:8, nrow=2, byrow=TRUE))
for (i in 1:length(Descriptors$Code)) {
 # Plot empirical AFDC
 plot(D_AFDC, AFDC50[i,], type="l", lty=1, col="black", lwd=2.75,
      log="y", yaxt="n",
      main=paste(Descriptors$Code[i], ": ", Descriptors$Stream[i], " at ", Descriptors$Gauge[i], sep=""),
      sub="Median AFDC",
      xlab="Duration", ylab=expression(paste("Discharge (",m^3,"/s)")))

 axis(2, at=c(0.01,0.1,1,10,100,1000))
 # Plot regional prediction
 lines(D_AFDC, predAFDC50cv[i,], lty=1, col="red", lwd=2.75)
  grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
       lwd=par("lwd"), equilogs=TRUE)
}
```

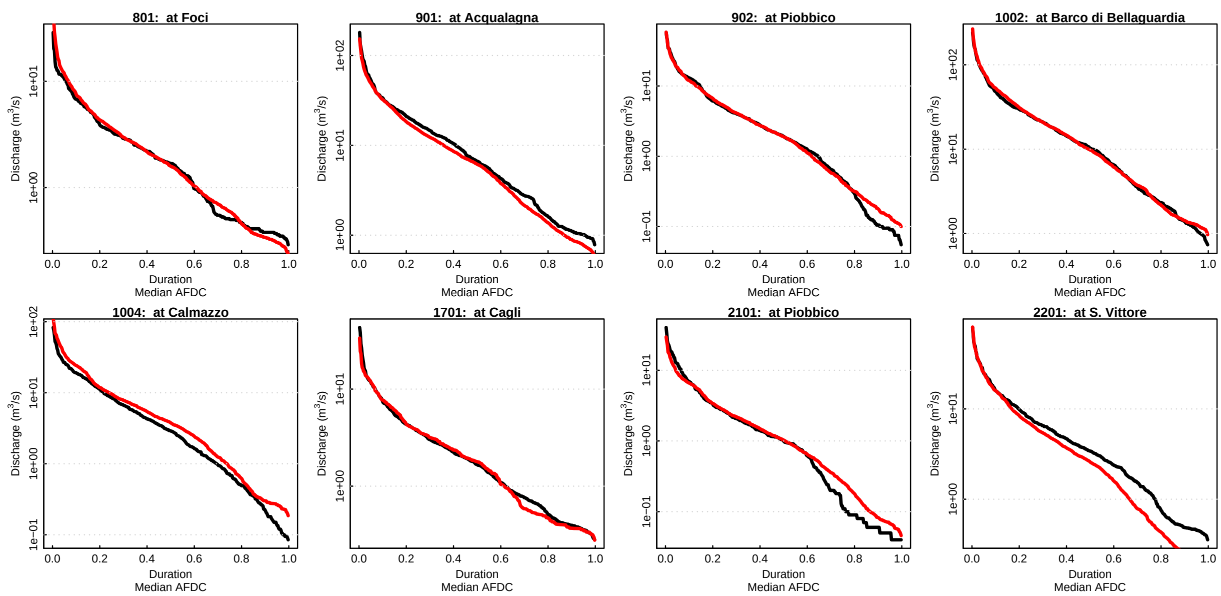
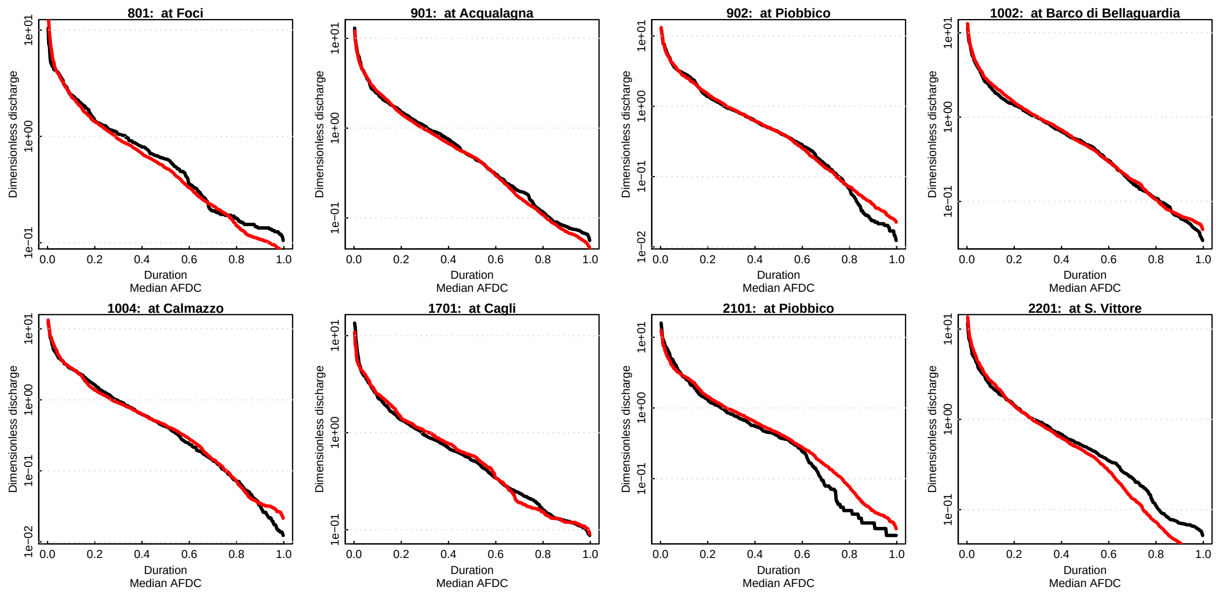


Figure.2 - Comparison of Dimensionless Median AFDC's:

```
layout(matrix(1:8, nrow=2, byrow=TRUE))
for (i in 1:length(Descriptors$Code)) {
 # Plot empirical AFDC
 plot(D_AFDC, normAFDC50[i,], type="l", lty=1, col="black", lwd=2.75,
      log="y", yaxt="n",
      main=paste(Descriptors$Code[i], ": ", Descriptors$Stream[i], " at ", Descriptors$Gauge[i], sep=""),
      sub="Median AFDC",
      xlab="Duration", ylab=paste("Dimensionless discharge"))

 axis(2, at=c(0.01,0.1,1,10,100,1000))
 # Plot regional prediction
 lines(D_AFDC, prednormAFDC50cv[i,], lty=1, col="red", lwd=2.75)
  grid(nx=NA, ny=NULL, col="lightgray", lty="dotted",
       lwd=par("lwd"), equilogs=TRUE)
}
```



6 Compare to the PUB book assessment

In the Level 2 Assessment of the PUB book (Blöschl et al., 2013) the performance assessment in Chapter 7 was based on the slope of the middle part of the FDC defined as the difference between the 30% and 70% normalised runoff quantiles divided by 40. This slope quantifies the relative change of runoff for 1% difference in exceedance probability.

Let's calculate the normalised error and the absolute normalised error in the estimation of the slope of the middle part of the FDC for the catchments considered in this exercise and compare it with Figures 7.22 and 7.23 in the PUB book.

```
obsq30q70 <- apply(normAFDC50, 1, quantile, prob=c(.7, .3))
 obsq30q70
```

```
          801      901       902      1002      1004      1701       2101      2201
70% 1.0505944 1.048520 0.8987145 0.9799412 0.9638072 0.9371686 0.81975047 0.9501708
30% 0.1993882 0.199087 0.1464538 0.1682248 0.1410379 0.2381944 0.07874644 0.2234753
```

```
predq30q70cv <- apply(prednormAFDC50cv, 1, quantile, prob=c(.7, .3))
 predq30q70cv
```

```
          801       901       902      1002      1004      1701      2101      2201
70% 0.9403149 0.9643612 0.9262579 0.9975213 0.9033317 1.0300856 0.9275115 0.9103100
30% 0.2223682 0.1693609 0.1323272 0.1800182 0.1445651 0.1904682 0.1530413 0.1327664
```

```
obsslFDC <- (obsq30q70[1,] - obsq30q70[2,])/40
predslFDCcv <- (predq30q70cv[1,] - predq30q70cv[2,])/40
```

```
NE <- (predslFDCcv - obsslFDC)/obsslFDC
ANE <- abs(NE)
tabella <- data.frame(Descriptors[,c("Code", "Area", "Elev", "MAT")], Aridity=PETovP, NE=round(NE, 3), ANE=round(ANE, 3))
 tabella
```

```
     Code   Area Elev  MAT   Aridity     NE   ANE
801   801  124.1 1702 12.0 0.5808359 -0.157 0.157
901   901  613.8 1702 12.1 0.6097750 -0.064 0.064
902   902  186.7 1526 12.7 0.6322758  0.055 0.055
1002 1002 1043.6 1702 12.4 0.6438393  0.007 0.007
1004 1004  375.9 1384 12.7 0.6474229 -0.078 0.078
1701 1701  126.1 1526 12.2 0.5596857  0.201 0.201
2101 2101   95.2 1526 13.0 0.6674429  0.045 0.045
2201 2201  263.6 1702 13.4 0.8280144  0.070 0.070
```

```
Aridity_class <- cut(tabella$Aridity, breaks=c(-Inf,0.4,0.6,0.8,1,2,Inf))
MAT_class <- cut(tabella$MAT,  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))
```

Notice that the method used here to calculate the normalised FDC is an index method.

```
add_points <- function(performance="ANE", variable="Area", classes, table) {
 # to add points in a nice way
 for (j in 1:length(levels(classes))) {
  dummy <- table[as.numeric(classes) == j,]
  perf <- dummy[, performance]
  stratif <- dummy[, variable]
  if (length(stratif) > 0) {
   if (length(stratif) == 1) {
    points(j, perf, pch=21,
           bg=colori[round(10*dummy$Aridity)],
           cex=log10(dummy$Area))
   } else {
      points(j + 0.1*(stratif - mean(stratif))/sd(stratif),
             perf, pch=21,
             bg=colori[round(10*dummy$Aridity)],
             cex=log10(dummy$Area))
   }
  }
 }
}
```

Fig 7.22 at page 159 of the book:

```
layout(matrix(1:4, nrow=1, byrow=TRUE))
plotPUBfiguresLevel2(chapter=7, method="Index", performance="ANE",
                     characteristic="Aridity", ylim=c(0.5,0),
                     main="Index")
 add_points(performance="ANE", variable="Aridity", classes=Aridity_class, table=tabella)
plotPUBfiguresLevel2(chapter=7, method="Index", performance="ANE",
                     characteristic="MAT", ylim=c(0.5,0))
 add_points(performance="ANE", variable="MAT", classes=MAT_class, table=tabella)
plotPUBfiguresLevel2(chapter=7, method="Index", performance="ANE",
                     characteristic="Elevation", ylim=c(0.5,0))
 add_points(performance="ANE", variable="Elev", classes=Elev_class, table=tabella)
plotPUBfiguresLevel2(chapter=7, method="Index", performance="ANE",
                     characteristic="Area", ylim=c(0.5,0))
 add_points(performance="ANE", variable="Area", classes=Area_class, table=tabella)
```

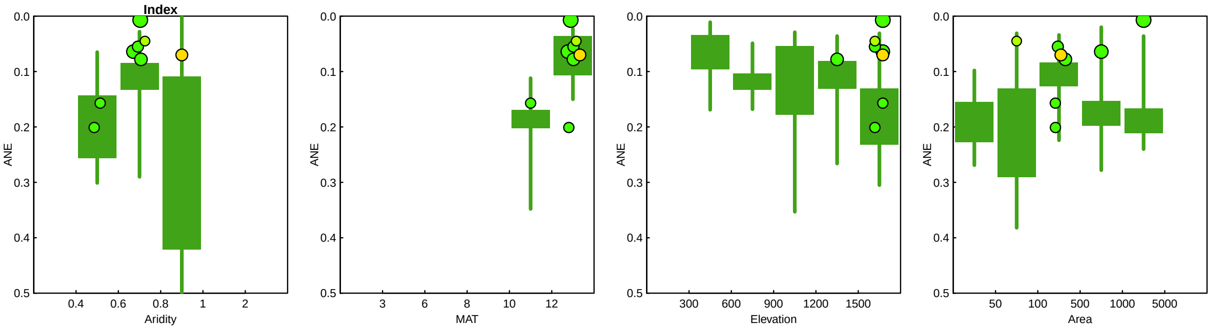
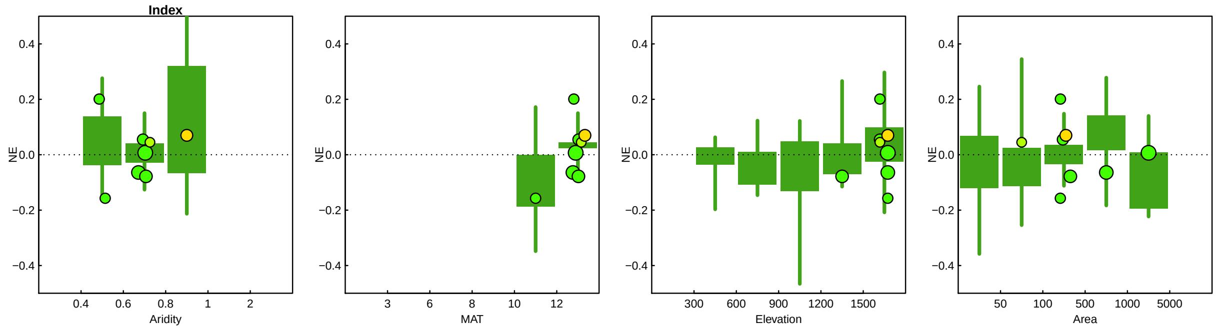


Fig 7.23 at page 160 of the book:

```
layout(matrix(1:4, nrow=1, byrow=TRUE))
plotPUBfiguresLevel2(chapter=7, method="Index", performance="NE",
                     characteristic="Aridity", ylim=c(-0.5,0.5),
                     main="Index"); abline(h=0, lty=3)
 add_points(performance="NE", variable="Aridity", classes=Aridity_class, table=tabella)
plotPUBfiguresLevel2(chapter=7, method="Index", performance="NE",
                     characteristic="MAT", ylim=c(-0.5,0.5)); abline(h=0, lty=3)
 add_points(performance="NE", variable="MAT", classes=MAT_class, table=tabella)
```

```
plotPUBfiguresLevel2(chapter=7, method="Index", performance="NE",
                     characteristic="Elevation", ylim=c(-0.5,0.5)); abline(h=0, lty=3)
 add_points(performance="NE", variable="Elev", classes=Elev_class, table=tabella)
plotPUBfiguresLevel2(chapter=7, method="Index", performance="NE",
                     characteristic="Area", ylim=c(-0.5,0.5)); abline(h=0, lty=3)
 add_points(performance="NE", variable="Area", classes=Area_class, table=tabella)
```



In this exercise we also have estimated the mean annual discharge through regression. How does it compare with the PUB book?

```
NE <- (predMAFcv - MAF)/MAF
ANE <- abs(NE)
names(tabella)[6:7] <- c("NEslope","ANEslope")
tabella <- data.frame(tabella, NE=round(NE, 3), ANE=round(ANE, 3))
 tabella
```

```
     Code   Area Elev  MAT   Aridity NEslope ANEslope     NE   ANE
801   801  124.1 1702 12.0 0.5808359  -0.157    0.157  0.143 0.143
901   901  613.8 1702 12.1 0.6097750  -0.064    0.064 -0.086 0.086
902   902  186.7 1526 12.7 0.6322758   0.055    0.055 -0.006 0.006
1002 1002 1043.6 1702 12.4 0.6438393   0.007    0.007 -0.023 0.023
1004 1004  375.9 1384 12.7 0.6474229  -0.078    0.078  0.251 0.251
1701 1701  126.1 1526 12.2 0.5596857   0.201    0.201 -0.040 0.040
2101 2101   95.2 1526 13.0 0.6674429   0.045    0.045 -0.077 0.077
2201 2201  263.6 1702 13.4 0.8280144   0.070    0.070 -0.133 0.133
```

Fig 5.27 at page 98 of the book:

```
layout(matrix(1:4, nrow=1, byrow=TRUE))
plotPUBfiguresLevel2(chapter=5, method="Regional_regr", performance="ANE",
                     characteristic="Aridity", ylim=c(3,0),
                     main="Regional_regr")
 add_points(performance="ANE", variable="Aridity", classes=Aridity_class, table=tabella)
plotPUBfiguresLevel2(chapter=5, method="Regional_regr", performance="ANE",
                     characteristic="MAT", ylim=c(3,0))
 add_points(performance="ANE", variable="MAT", classes=MAT_class, table=tabella)
plotPUBfiguresLevel2(chapter=5, method="Regional_regr", performance="ANE",
                     characteristic="Area", ylim=c(3,0))
 add_points(performance="ANE", variable="Area", classes=Area_class, table=tabella)
```

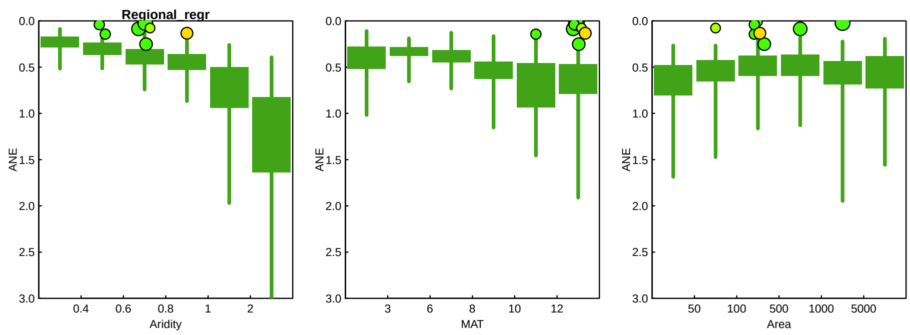
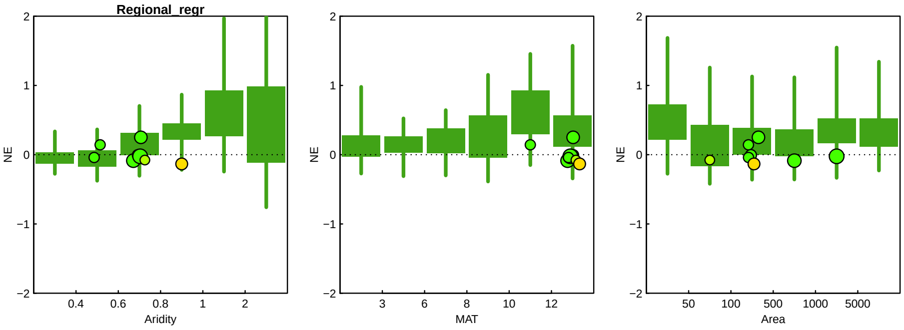


Fig 5.28 at page 99 of the book:

```
layout(matrix(1:4, nrow=1, byrow=TRUE))
plotPUBfiguresLevel2(chapter=5, method="Regional_regr", performance="NE",
                     characteristic="Aridity", ylim=c(-2,2),
                     main="Regional_regr"); abline(h=0, lty=3)
```

```
 add_points(performance="NE", variable="Aridity", classes=Aridity_class, table=tabella)
plotPUBfiguresLevel2(chapter=5, method="Regional_regr", performance="NE",
                     characteristic="MAT", ylim=c(-2,2)); abline(h=0, lty=3)
 add_points(performance="NE", variable="MAT", classes=MAT_class, table=tabella)
plotPUBfiguresLevel2(chapter=5, method="Regional_regr", performance="NE",
                     characteristic="Area", ylim=c(-2,2)); abline(h=0, lty=3)
 add_points(performance="NE", variable="Area", classes=Area_class, table=tabella)
```



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.

Castellarin, A., Botter, G., Hughes, D.A., Liu, S., Ouarda, T.B.M.J., Parajka, J., Post, D.A., Sivapalan, M., Spence, C., Viglione, A. and Vogel, R.M. (2013). Prediction of flow duration curves in ungauged basins. In *Runoff Prediction in Ungauged Basins: Synthesis Across Processes, Places and Scales*, University Press, Cambridge, 135-162, ISBN:9781107028180.

Vogel, R.M. and Fennessey, N.M. (1994). Flow-duration Curves. I: New Interpretation and Confidence Intervals. *Journal of Water Resources Planning and Management-ASCE*, **120**(4):485–504, doi:10.1061/(ASCE)0733-9496(1994)120:4(485).

Vogel, R.M. and Fennessey, N.M. (1995). Flow Duration Curves II: A Review of Applications in Water Resources Planning. *Journal of the American Water Resources Association*, **31**(6):1029–1039, doi:10.1111/j.1752-1688.1995.tb03419.x.