# Chapter 10: Prediction of runoff hydrographs in ungauged basins - an Austrian example

Juraj Parajka

## 1 Introduction

This Tutorial has been developed by Juraj Parajka to illustrate two regionalisation methods (spatial proximity and similarity) for predicting daily runoff hydrographs (see e.g., Blöschl et al., 2013). A detailed literature review of regionalisation methods is available in Parajka et al. (2013).

First of all load the library:

```
library(PUBexamples)
```

Then the data:

```
help(data4chapter10)
```

```
data(data4chapter10)
head(CatChar, 20)
```

| | idnr | station | river | lat | lon | xcor | ycor | area | elev | slope | forest | xcen | ycen | map | aridity |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 200105 | Garsella | Lutz | 47.2272 | 9.8761 | 138378.8 | 375507.9 | 95.5 | 1598 | 0.454 | 0.274 | 145244.8 | 376011.2 | 1678 | 0.252 |
| 2 | 200147 | Gisingen | Ill | 47.2608 | 9.5789 | 116084.1 | 380282.3 | 1281.0 | 1585 | 0.412 | 0.329 | 138921.5 | 361219.3 | 1539 | 0.332 |
| 3 | 200196 | Lustenau_(Eisenbahnbrue_ke) | Rhein | 47.4483 | 9.6589 | 123114.2 | 400806.2 | 6110.1 | 1687 | 0.388 | 0.329 | 110167.5 | 334731.2 | 1539 | 0.298 |
| 4 | 200204 | Enz | Dornbirnerach | 47.3978 | 9.7613 | 130567.8 | 394836.6 | 51.1 | 1119 | 0.298 | 0.749 | 131490.8 | 390347.4 | 2112 | 0.283 |
| 5 | 200212 | Hoher_Steg | Dornbirnerach | 47.4483 | 9.7000 | 126209.8 | 400660.5 | 112.9 | 885 | 0.220 | 0.585 | 131656.6 | 395014.6 | 1882 | 0.340 |
| 6 | 200220 | Lustenau_(Hofsteig) | Rheintalbinnenkanal | 47.4353 | 9.6969 | 125911.8 | 399221.6 | 77.5 | 558 | 0.148 | 0.326 | 124330.9 | 391553.9 | 1437 | 0.482 |
| 7 | 200246 | Hopfreben | Bregenzerach | 47.2862 | 10.0469 | 151572.3 | 381495.2 | 41.7 | 1685 | 0.431 | 0.234 | 154313.5 | 378393.0 | 1635 | 0.250 |
| 8 | 200253 | Au | Bregenzerach | 47.3300 | 9.9731 | 146204.1 | 386596.7 | 149.3 | 1520 | 0.397 | 0.330 | 148288.5 | 382090.6 | 1815 | 0.248 |
| 9 | 200261 | Mellau | Bregenzerach | 47.3553 | 9.8811 | 139389.0 | 389707.1 | 228.6 | 1431 | 0.389 | 0.377 | 144913.6 | 383793.4 | 1920 | 0.267 |
| 10 | 200287 | Schoenenbach_(Hengstig) | Subersach | 47.3864 | 10.0411 | 151606.5 | 392637.5 | 31.1 | 1460 | 0.328 | 0.383 | 152577.7 | 390048.9 | 1954 | 0.232 |
| 11 | 200311 | Thal_(Martinsbruecke) | Rotach | 47.5289 | 9.8750 | 139786.3 | 409001.9 | 90.1 | 753 | 0.099 | 0.472 | 141023.4 | 413131.9 | 1794 | 0.372 |
| 12 | 200360 | Lech | Zuersbach | 47.2056 | 10.1411 | 158321.8 | 372243.9 | 25.2 | 2070 | 0.388 | 0.203 | 159941.9 | 369090.8 | 1646 | 0.229 |
| 13 | 200378 | Lech_(Tannbergbruecke) | Lech | 47.2083 | 10.1411 | 158334.4 | 372552.2 | 84.3 | 1991 | 0.392 | 0.150 | 154656.5 | 370487.2 | 1548 | 0.225 |
| 14 | 201012 | Steeg | Lech | 47.2425 | 10.2942 | 170063.8 | 375881.6 | 247.9 | 1936 | 0.422 | 0.249 | 161849.5 | 372248.2 | 1520 | 0.244 |
| 15 | 201053 | Vorderhornbach_(Bruecke | Hornbach | 47.3692 | 10.5383 | 189034.8 | 389255.4 | 64.0 | 1689 | 0.542 | 0.330 | 181467.9 | 387812.9 | 1845 | 0.307 |
| 16 | 201087 | Lechaschau | Lech | 47.4975 | 10.7100 | 202468.4 | 403056.9 | 1012.2 | 1734 | 0.452 | 0.383 | 182085.8 | 382033.1 | 1573 | 0.295 |
| 17 | 201111 | Vils_(Laende) | Vils | 47.5506 | 10.6486 | 198051.4 | 409107.2 | 198.1 | 1273 | 0.306 | 0.557 | 189094.4 | 406872.0 | 1770 | 0.327 |
| 18 | 201368 | Vent_(unterh._Niederta_bach) | Venter_Ache | 46.8591 | 10.9131 | 215550.0 | 331649.9 | 164.7 | 2915 | 0.393 | 0.021 | 211462.2 | 327766.3 | 934 | 0.278 |
| 19 | 201533 | Gries_am_Brenner | Obernberger_Seebach | 47.0361 | 11.4792 | 259148.5 | 350129.6 | 58.3 | 1843 | 0.369 | 0.368 | 254794.7 | 347062.5 | 1198 | 0.378 |
| 20 | 201574 | Puig | Sill | 47.1122 | 11.4542 | 257454.2 | 358630.8 | 341.8 | 1927 | 0.425 | 0.343 | 258068.5 | 350884.3 | 1162 | 0.362 |

The dataset consists of 213 Austrian basins analysed in Viglione et al. (2013).

Plot the data on a map:

```
library(rworldmap)
newMap <- getMap(resolution="low")
```

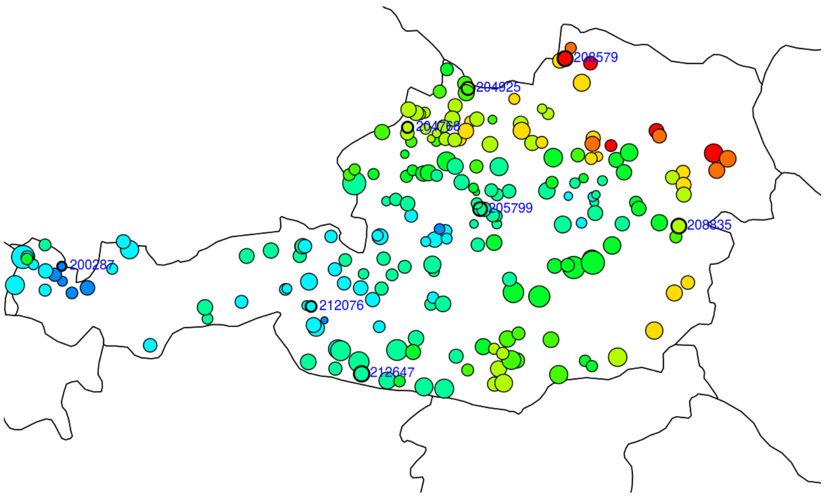
```
colori <- rev(rainbow(10, start=0, end=.65, alpha=1))
plot(newMap, xlim=c(11.5, 15.5), ylim=c(46, 49))
 points(CatChar$lon, CatChar$lat, pch=21,
        bg=colori[round(10*CatChar$aridity)],
        cex=log10(CatChar$area))
# validation catchments
names(ObsDischarges)
```

```
[1] "200287" "204768" "204925" "205799" "208579" "208835" "212076" "212647"
```

```
valCatChar <- CatChar[CatChar$idnr %in% names(ObsDischarges),]
 valCatChar
```

| | idnr | station | river | lat | lon | xcor | ycor | area | elev | slope | forest | xcen | ycen | map | aridity |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 10 | 200287 | Schoenenbach_(Hengstig) | Subersach | 47.3864 | 10.0411 | 151606.5 | 392637.5 | 31.1 | 1460 | 0.328 | 0.383 | 152577.7 | 390048.9 | 1954 | 0.232 |
| 56 | 204768 | Osternach | Osternach | 48.3136 | 13.4539 | 408938.6 | 490432.8 | 68.6 | 431 | 0.096 | 0.173 | 413971.8 | 485148.3 | 964 | 0.689 |
| 66 | 204925 | Hartmannsdorf | Steinerne_Muehl | 48.5731 | 14.0506 | 452911.5 | 519510.8 | 137.8 | 751 | 0.101 | 0.574 | 460207.6 | 518379.4 | 898 | 0.666 |
| 97 | 205799 | Kniewas | Steyr | 47.7678 | 14.1700 | 462686.7 | 430096.1 | 184.9 | 1176 | 0.399 | 0.718 | 460291.4 | 421944.6 | 1513 | 0.383 |
| 147 | 208579 | Hoheneich | Braunaubach | 48.7719 | 15.0089 | 523125.8 | 542706.1 | 291.5 | 574 | 0.044 | 0.430 | 529472.0 | 550207.9 | 628 | 0.959 |
| 153 | 208835 | Warth | Pitten | 47.6547 | 16.1314 | 610053.4 | 420977.1 | 277.0 | 830 | 0.200 | 0.707 | 602316.4 | 412489.4 | 962 | 0.664 |
| 183 | 212076 | Matreier_Tauernhaus | Tauernbach | 47.1186 | 12.5017 | 336915.9 | 357954.7 | 59.9 | 2485 | 0.426 | 0.050 | 331012.2 | 358505.3 | 1143 | 0.255 |
| 195 | 212647 | Mauthen | Gail | 46.6686 | 13.0003 | 374522.2 | 307665.4 | 348.6 | 1628 | 0.478 | 0.548 | 355967.8 | 309782.2 | 1200 | 0.378 |

```
points(valCatChar$lon, valCatChar$lat, pch=1, lwd=2,
       cex=log10(valCatChar$area))
text(valCatChar$lon, valCatChar$lat, valCatChar$idnr,
     pos=4, col=4)
```




## 2 Run TUWmodel for one basin

Load the libraries that will be used:

```
library(zoo)
library(TUWmodel)
```

Read the model inputs for basin Schönenbach (idnr=200287)

```
idnr='200287'
data <- ModelInput[[idnr]]
head(data, 20)
```

```
   day mon   yr     prec     temp     pet     snowd
1    1   1 1999  0.00000  1.01899 0.00303  64.83003
2    2   1 1999  1.61151 -1.07980 0.00000  63.42897
3    3   1 1999  5.08623 -0.76300 0.00000  79.11185
4    4   1 1999  5.10933  1.71687 0.00394  77.52865
5    5   1 1999  0.00000  2.58336 0.00970  72.14805
6    6   1 1999  0.00000  3.22088 0.03970  71.03627
7    7   1 1999  6.50424  2.17731 0.02182  67.17229
8    8   1 1999 18.52378  1.56353 0.00576  66.82926
9    9   1 1999  3.64854 -2.64802 0.00000  86.07787
10  10   1 1999  0.00073 -4.72470 0.00000  92.68480
11  11   1 1999  2.88789 -1.98919 0.00000  91.07223
12  12   1 1999  3.64262 -7.18713 0.00000  87.46131
13  13   1 1999  9.51652 -7.20530 0.00000  89.28735
14  14   1 1999  4.41914 -3.39212 0.00000  97.06496
15  15   1 1999  0.00404 -3.46520 0.00000 111.49722
16  16   1 1999  0.00000 -0.02108 0.00333  95.09336
17  17   1 1999  0.00003 -0.93860 0.00121  98.77010
18  18   1 1999  0.00000 -1.56748 0.00061  94.18652
19  19   1 1999  0.00000 -1.50430 0.00000  93.05502
20  20   1 1999  0.00000 -0.77582 0.00000  92.31326
```

```
days <- as.Date(strptime(paste(data[,1], data[,2], data[,3]), format="%d %m %Y"))
P <- zoo(data[,4], order.by=days)     # daily catchment precipitation (mm/d)
T <- zoo(data[,5], order.by=days)     # mean daily catchment temperature (deg C)
EP <- zoo(data[,6], order.by=days)     # mean daily pot.evaporation (mm/d)
EP[EP < 0] <- 0                       # daily potential evapotranspiration (mm/d)
```

Read the observed discharges:

```
data2 <- ObsDischarges[[idnr]]
head(data2, 20)
```

```
   day mon   yr  disc
1    1   1 1999 0.563
2    2   1 1999 0.540
3    3   1 1999 0.572
4    4   1 1999 0.909
5    5   1 1999 1.210
6    6   1 1999 0.934
7    7   1 1999 0.895
8    8   1 1999 1.190
9    9   1 1999 1.110
10  10   1 1999 0.685
11  11   1 1999 0.669
```

```
12  12   1 1999 0.498
13  13   1 1999 0.433
14  14   1 1999 0.470
15  15   1 1999 0.505
16  16   1 1999 0.473
17  17   1 1999 0.461
18  18   1 1999 0.449
19  19   1 1999 0.437
20  20   1 1999 0.425
```

```
area <- CatChar[CatChar$idnr == idnr, "area"]
days2 <- as.Date(strptime(paste(data2[,1], data2[,2], data2[,3]), format="%d %m %Y"))
Q <- zoo(data2[,4], order.by=days2)    # daily discharge (m3/s)
Qmm <- Q*86.4/area  #conversion of discharge to mm
Qmm[Qmm < 0] <- NA
head(P, 20)
```

```
1999-01-01 1999-01-02 1999-01-03 1999-01-04 1999-01-05 1999-01-06 1999-01-07 1999-01-08 1999-01-09 1999-01-10 1999-01-11 1999-01-12 1999-01-13 1999-01-14 1999-01-15 1999-01-16
   0.00000    1.61151    5.08623    5.10933    0.00000    0.00000    6.50424   18.52378    3.64854    0.00073    2.88789    3.64262    9.51652    4.41914    0.00404    0.00000
1999-01-17 1999-01-18 1999-01-19 1999-01-20
   0.00003    0.00000    0.00000    0.00000
```

```
head(T, 20)
```

```
1999-01-01 1999-01-02 1999-01-03 1999-01-04 1999-01-05 1999-01-06 1999-01-07 1999-01-08 1999-01-09 1999-01-10 1999-01-11 1999-01-12 1999-01-13 1999-01-14 1999-01-15 1999-01-16
   1.01899   -1.07980   -0.76300    1.71687    2.58336    3.22088    2.17731    1.56353   -2.64802   -4.72470   -1.98919   -7.18713   -7.20530   -3.39212   -3.46520   -0.02108
1999-01-17 1999-01-18 1999-01-19 1999-01-20
  -0.93860   -1.56748   -1.50430   -0.77582
```

```
head(EP, 20)
```

```
1999-01-01 1999-01-02 1999-01-03 1999-01-04 1999-01-05 1999-01-06 1999-01-07 1999-01-08 1999-01-09 1999-01-10 1999-01-11 1999-01-12 1999-01-13 1999-01-14 1999-01-15 1999-01-16
   0.00303    0.00000    0.00394    0.00970    0.03970    0.02182    0.00576    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00000    0.00333
1999-01-17 1999-01-18 1999-01-19 1999-01-20
   0.00121    0.00061    0.00000    0.00000
```

```
head(Qmm, 20)
```

```
1999-01-01 1999-01-02 1999-01-03 1999-01-04 1999-01-05 1999-01-06 1999-01-07 1999-01-08 1999-01-09 1999-01-10 1999-01-11 1999-01-12 1999-01-13 1999-01-14 1999-01-15 1999-01-16
  1.564090   1.500193   1.589093   2.525325   3.361543   2.594778   2.486431   3.305981   3.083730   1.903023   1.858572   1.383511   1.202932   1.305723   1.402958   1.314058
1999-01-17 1999-01-18 1999-01-19 1999-01-20
  1.280720   1.247383   1.214045   1.180707
```

Take a subset of data for model running, e.g. 1.Nov 1999 - 31.Dec 2010:

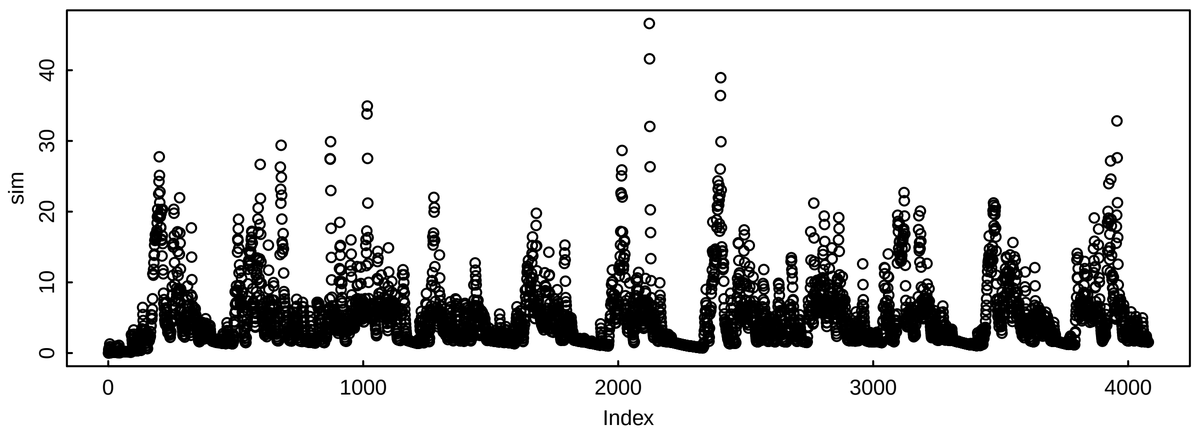
```
P1 <- window(P, start=as.Date("1 11 1999", format="%d %m %Y"), end=as.Date("31 12 2010", format="%d %m %Y"))
T1 <- window(T, start=as.Date("1 11 1999", format="%d %m %Y"), end=as.Date("31 12 2010", format="%d %m %Y"))
EP1 <- window(EP, start=as.Date("1 11 1999", format="%d %m %Y"), end=as.Date("31 12 2010", format="%d %m %Y"))
Q1 <- window(Qmm, start=as.Date("1 11 1999", format="%d %m %Y"), end=as.Date("31 12 2010", format="%d %m %Y"))
```

Run the TUWmodel, model parameters are initial guess (please check help `?TUWmodel` for explanantion about the order and meaning of model parameters):

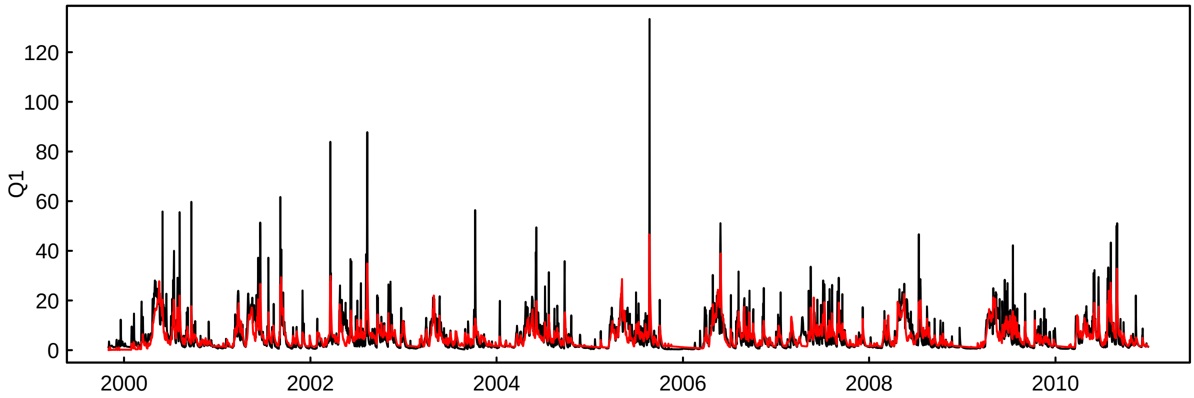
```
simulation1 <- TUWmodel(prec=as.numeric(P1), airt=as.numeric(T1), ep=as.numeric(EP1), area=area,
                        param=c(1.0, 2.0, -1.0, 1.0, 0.0,
                                0.8, 400.0, 0.2,
                                0.3, 7.0, 150.0,
                                50.0, 2.0, 10.0, 25.0))
```

Plot the results:

```
sim <- as.numeric(simulation1$q)
plot(sim)
```

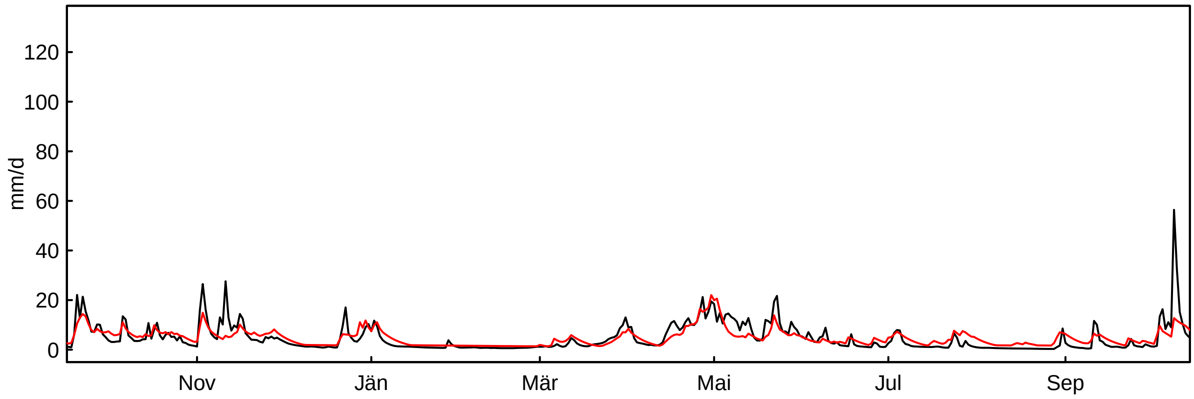


```
Qsim <- zoo(sim, order.by=index(P1))
plot(Q1, xlab="")  # observed discharge
lines(Qsim, col="red") # simulated discharge
```




Zoom-in the plot for selected time period:

```
limit <-  as.Date(strptime(c("1 10 2002", "30 09 2003"), format="%d %m %Y"))
plot(Q1, xlab="", ylab="mm/d", xlim=limit)
lines(Qsim, col="red")
```




Calculate runoff model efficiency (e.g. Nash-Sutcliffe, NSE):

```
NSE <- function(simulations, observations) {
 simu <- simulations[-c(1:304)]  # remove the warming period
 obse <- observations[-c(1:304)]  # remove the warming period
 mobs <- mean(obse, na.rm=TRUE)
 1 - sum((simu - obse)^2, na.rm=TRUE)/sum((obse - mobs)^2, na.rm=TRUE)
}
nse_sim <- NSE(sim, as.numeric(Q1))
 nse_sim
```

```
[1] 0.6006289
```

## 2.1 Automatic calibration of the model

This part needs to run what is in the previous sections first (loading packages, reading data, etc.).

```
library(DEoptim)
# For more details use the R help: help(DEoptim)
```

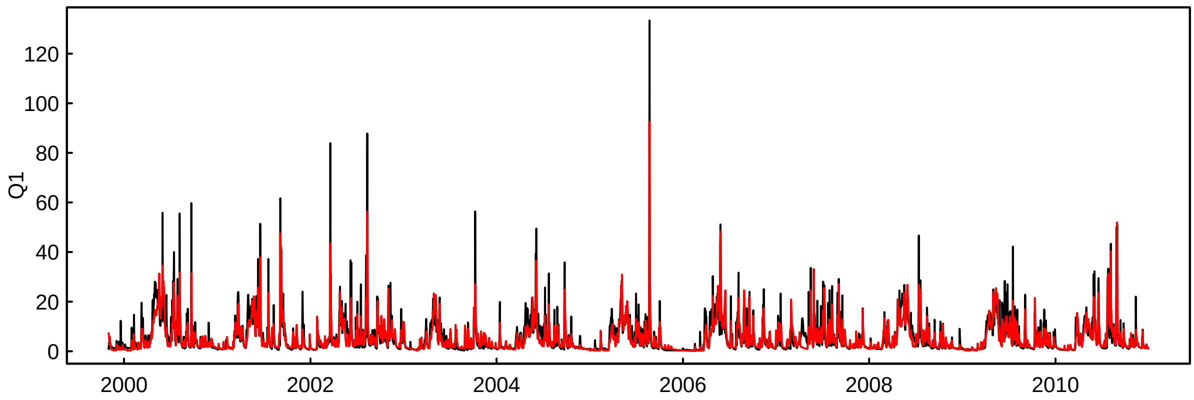
Define objective function for calibration (e.g. mean square error):

```
MSE <- function (param, precip, temp, potevap, runoff, area) {
 simu <- TUWmodel(prec=as.numeric(precip), airt=as.numeric(temp), ep=as.numeric(potevap), area=area, param)$q
 simu <- simu[-c(1:304)]  # remove the warming period (1 year)
 obse <- runoff[-c(1:304)]  # remove the warming period (1 year)
 mobs <- mean(as.numeric(obse), na.rm=TRUE)
 mean((as.numeric(simu) - as.numeric(obse))^2, na.rm=TRUE)  # mean square error
}
```

```
# ATTENTION: THIS TAKES TIME!!
calibrate_period1 <- DEoptim(fn=MSE, lower=c(0.9, 0.0, 1.0, -3.0, -2.0, 0.0, 0.0, 0.0, 0.0, 2.0, 30.0, 1.0, 0.0, 0.0, 0.0),
                                     upper=c(1.5, 5.0, 3.0, 1.0, 2.0, 1.0, 600.0, 20.0, 2.0, 30.0, 250.0, 100.0, 8.0, 30.0, 50.0),
                             control=DEoptim.control(NP=NA, itermax=1000, reltol=1e-4, steptol=50, trace=10, parallelType=0),
                             precip=P1, temp=T1, potevap=EP1, runoff=Q1, area=area)
 bestparameters <- calibrate_period1$optim$bestmem
```

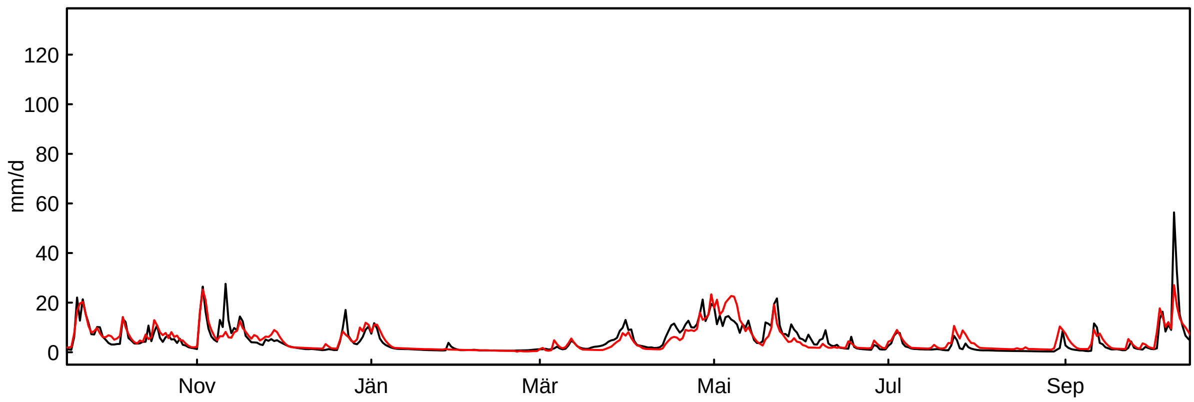
```
simulation1_cal1 <- TUWmodel(prec=as.numeric(P1), airt=as.numeric(T1), ep=as.numeric(EP1), area=area,
                             param=bestparameters)
 simcal <- as.numeric(simulation1_cal1$q)
```

```
Qsimcal <- zoo(simcal, order.by=index(P1))
plot(Q1, xlab="")  # observed discharge
lines(Qsimcal, col="red") # simulated discharge
```




Zoom-in the plot for selected time period:

```
limit <-  as.Date(strptime(c("1 10 2002", "30 09 2003"), format="%d %m %Y"))
plot(Q1, xlab="", ylab="mm/d", xlim=limit)
lines(Qsimcal, col="red")
```



```
nse_simcal <- NSE(simcal, as.numeric(Q1))
 nse_simcal
```

```
[1] 0.7113309
```

Calculation of more runoff efficiency measures:

```
EMs <- function (sim, obs, warmup=304) {
 # obs = observed runoff in mm/d (class numeric)
 # sim = simulated runoff in mm/d (class numeric)
 # warmup = warm-up period in d

 simu <- as.numeric(sim[-c(1:warmup)])
 obse <- as.numeric(obs[-c(1:warmup)])

 # RMSE = root mean square error (mm/d)
 RMSE <- sqrt(mean((simu - obse)^2, na.rm=TRUE))

 # NE = Nash efficiency ()
 mobse <- mean(obse, na.rm=TRUE)
 NE <- 1 - sum((simu - obse)^2, na.rm=TRUE)/sum((obse - mobse)^2, na.rm=TRUE)

 # lNE = log Nash efficiency ()
 mlobse <- mean(log(obse), na.rm=TRUE)
 lNE <- 1 - sum((log(simu) - log(obse))^2, na.rm=TRUE)/sum((log(obse) - mlobse)^2, na.rm=TRUE)

 # B = bias (mm/d)
 B <- mean(simu - obse, na.rm=TRUE)

 # MAE = mean absolute error (mm/d)
 MAE <- mean(abs(simu - obse), na.rm=TRUE)

 # MAlE = mean absolute log error (mm/d)
 MAlE <- exp(mean(abs(log(simu) - log(obse)), na.rm=TRUE))

 # VE = volume error (%/%)
 VE <- (sum(simu[!is.na(obse)]) - sum(obse, na.rm=TRUE))/sum(obse, na.rm=TRUE)

 output <- c(RMSE, NE, lNE, B, MAE, MAlE, VE)
  names(output) <- c("RMSE (mm/d)", "Nash efficiency ()", "log Nash efficiency ()", "bias (mm/d)",
                     "mean absolute error (mm/d)", "mean absolute log error (mm/d)", "volume error (%/%)")
 return(output)
}
# The following code returns a matrix with the efficiencies
efficiencies <- rbind(EMs(as.numeric(simcal), as.numeric(Q1)))
 t(efficiencies)
```

```
                                     [,1]
RMSE (mm/d)                    3.76363326
Nash efficiency ()             0.71133095
log Nash efficiency ()         0.74211299
bias (mm/d)                   -0.14965694
mean absolute error (mm/d)     2.00437426
mean absolute log error (mm/d) 1.52211266
volume error (%/%)            -0.02780626
```

# 3 Regionalisation of TUWmodel parameters - Nearest Neighbor method

The main idea is to estimate/transfer model parameters to selected (validation) basins and run the model in there basins (which are considered as ungauged). The model parameters will be transferred (regionalized) from the nearest basin.

The calibrated model parameters for all 213 basins in Austria are given in `calibPar` variable

```
head(CalibPar, 20)
```

```
     idnr     csf     ddf      tr       ts    meltt   lprat        fc     beta      k0       k1        k2     lsuz   cperc     bmax   croute
1  200105 0.98245 1.00556 2.99685 -2.99515 -1.92923 0.70462 558.66158  0.22698 0.30093  6.90642 165.91207 66.13367 1.29885 29.88823 31.05217
2  200147 1.07682 0.78716 2.99942 -2.97481 -1.88801 0.99629 190.04928  0.05274 0.19176 22.70026 199.16554 99.64206 2.82737 29.76236 17.64499
3  200196 0.97340 1.72153 2.72986 -2.65757  1.20212 0.24799 426.29022  0.27711 0.03425 25.67505 206.76603  4.23960 2.34106  4.83875  5.65189
4  200204 0.90163 2.20764 2.99920 -2.56773  0.28440 0.99838  28.84397  0.90522 0.99953  2.04670  30.19060 28.53517 1.89924 27.95784 36.73145
5  200212 0.90006 1.42752 2.99637 -2.13429  0.05914 0.99105  63.57957  9.93512 0.41693  3.35411  32.22073 33.46342 3.19720 23.36840 25.09204
6  200220 0.90015 0.27228 2.99209 -2.98480 -0.25530 0.99662 154.31558 19.99556 0.37875  5.10231 138.04891 29.13386 3.09982  5.48061 30.85922
7  200246 1.13001 1.64726 1.87107 -1.67406 -0.61862 0.77052 558.79512  0.07119 0.47711  6.39497 245.83287 48.38558 1.00315  7.95822 13.95476
8  200253 0.95933 1.55735 2.90374 -2.13614  0.25497 0.09661 386.45005  0.10731 0.40797  4.35917 222.11416 46.10512 1.16541  8.87345 10.88462
9  200261 0.91929 1.41654 2.97734 -1.79608  0.62356 0.99713  43.08990  0.51049 0.37296  3.80247 248.66566 46.16550 1.11818 14.37807 49.53087
10 200287 1.08417 1.83068 2.85206 -1.76107  0.98971 0.99254  38.92727  0.75957 0.43923  3.17919  31.41253 26.77558 1.85932 24.36621 28.19197
11 200311 0.93786 2.39499 2.96345 -2.95308  1.98197 0.98535  65.48787  2.95705 0.45537  3.36202  30.44862 13.12456 1.48330 25.20450 35.10156
12 200360 1.12801 1.65529 2.77498 -1.37076 -1.54887 0.90475 576.34386  0.00001 0.37786  8.39137 176.21412 65.71037 0.00769  2.29494 47.48226
13 200378 1.04681 1.62624 2.36976 -1.86730 -1.79713 0.74216 198.06720  0.03571 0.37797  5.60576  30.12673 60.59552 3.34603  1.00515 27.14511
14 201012 1.04888 1.28176 2.97841 -2.72854 -1.99377 0.56326 252.06957  0.16055 0.35700  8.62525 204.48408 55.95989 0.61258  1.56071  8.03359
15 201053 0.91309 1.01884 2.84067 -2.99805  0.60401 0.91024  58.71728  0.03095 0.43394  7.22383 190.11488 57.26154 1.12520 26.44141 32.66442
16 201087 0.91325 0.95283 2.27480 -2.99954 -1.14453 0.80596 236.76298  0.21425 0.39143 11.33785 221.51829 75.03757 1.34551 17.82024 41.61053
17 201111 0.90106 1.06989 2.96363 -2.06222 -1.98192 0.99566  83.24343  2.32071 0.35344  5.96568  30.21537 32.23274 4.48914 28.90086 24.71163
18 201368 1.26281 1.85293 2.83585 -2.97933 -0.02011 0.28001 560.94864  0.01635 0.36878  7.64033  77.69032 45.04724 0.05737 22.71848 29.30206
19 201533 1.02229 1.54141 2.42724 -1.60741  0.41882 0.01364 323.66137  0.32719 0.19879 29.08766 101.76126 41.11695 1.82615 16.85901 23.47036
20 201574 0.93111 1.81304 2.51196 -2.45460  0.90613 0.78590 360.79980  0.53560 0.03158 24.25668 134.49856 36.36148 1.42583 15.15349 31.01803
```

The basin characteristics for the same 213 Austria basins are in `CatChar`:

```
head(CatChar, 20)
```

```
     idnr                        station                river     lat     lon     xcor     ycor   area elev slope forest     xcen     ycen  map aridity
1  200105                       Garsella                 Lutz 47.2272  9.8761 138378.8 375507.9   95.5 1598 0.454  0.274 145244.8 376011.2 1678   0.252
2  200147                       Gisingen                  Ill 47.2608  9.5789 116084.1 380282.3 1281.0 1585 0.412  0.329 138921.5 361219.3 1539   0.332
3  200196  Lustenau_(Eisenbahnbrue_ke)                  Rhein 47.4483  9.6589 123114.2 400806.2 6110.1 1687 0.388  0.329 110167.5 334731.2 1539   0.298
4  200204                            Enz        Dornbirnerach 47.3978  9.7613 130567.8 394836.6   51.1 1119 0.298  0.749 131490.8 390347.4 2112   0.283
5  200212                     Hoher_Steg        Dornbirnerach 47.4483  9.7000 126209.8 400660.5  112.9  885 0.220  0.585 131656.6 395014.6 1882   0.340
6  200220            Lustenau_(Hofsteig) Rheintalbinnenkanal 47.4353  9.6969 125911.8 399221.6   77.5  558 0.148  0.326 124330.9 391553.9 1437   0.482
7  200246                      Hopfreben         Bregenzerach 47.2862 10.0469 151572.3 381495.2   41.7 1685 0.431  0.234 154313.5 378393.0 1635   0.250
8  200253                             Au         Bregenzerach 47.3300  9.9731 146204.1 386596.7  149.3 1520 0.397  0.330 148288.5 382090.6 1815   0.248
9  200261                         Mellau         Bregenzerach 47.3553  9.8811 139389.0 389707.1  228.6 1431 0.389  0.377 144913.6 383793.4 1920   0.267
10 200287       Schoenenbach_(Hengstig)            Subersach 47.3864 10.0411 151606.5 392637.5   31.1 1460 0.328  0.383 152577.7 390048.9 1954   0.232
11 200311         Thal_(Martinsbruecke)               Rotach 47.5289  9.8750 139786.3 409001.9   90.1  753 0.099  0.472 141023.4 413131.9 1794   0.372
12 200360                           Lech            Zuersbach 47.2056 10.1411 158321.8 372243.9   25.2 2070 0.388  0.203 159941.9 369090.8 1646   0.229
13 200378        Lech_(Tannbergbruecke)                  Lech 47.2083 10.1411 158334.4 372552.2   84.3 1991 0.392  0.150 154656.5 370487.2 1548   0.225
14 201012                          Steeg                 Lech 47.2425 10.2942 170063.8 375881.6  247.9 1936 0.422  0.249 161849.5 372248.2 1520   0.244
15 201053       Vorderhornbach_(Bruecke              Hornbach 47.3692 10.5383 189034.8 389255.4   64.0 1689 0.542  0.330 181467.9 387812.9 1845   0.307
16 201087                     Lechaschau                 Lech 47.4975 10.7100 202468.4 403056.9 1012.2 1734 0.452  0.383 182085.8 382033.1 1573   0.295
17 201111                 Vils_(Laende)                  Vils 47.5506 10.6486 198051.4 409107.2  198.1 1273 0.306  0.557 189094.4 406872.0 1770   0.327
18 201368 Vent_(unterh._Niederta_bach)            Venter_Ache 46.8591 10.9131 215550.0 331649.9  164.7 2915 0.393  0.021 211462.2 327766.3  934   0.278
19 201533               Gries_am_Brenner Obernberger_Seebach 47.0361 11.4792 259148.5 350129.6   58.3 1843 0.369  0.368 254794.7 347062.5 1198   0.378
20 201574                           Puig                 Sill 47.1122 11.4542 257454.2 358630.8  341.8 1927 0.425  0.343 258068.5 350884.3 1162   0.362
```

Find the nearest neighbour for validation basins and write out the corresponding model parameters:

```
val_catch <- names(ObsDischarges)
# initialise output matrix with idnr, nearest neighbor, distance and 15 model parameters
reg1_Par <- as.data.frame(matrix(NA, nrow=length(val_catch), ncol=3 + 15,
                                 dimnames=list(1:length(val_catch),
                                               c("idnr", "near_neigh", "dist", names(CalibPar)[-1]))))
 reg1_Par$idnr <- val_catch
# loop (notice that the distance refers to station coordinates)
nn <- dim(CatChar)[1]  # total number of basins in dataset
for(j in 1:length(val_catch)) { # loop for all validation basins
 codice0 <- val_catch[j]
 val_catch_Char <- CatChar[CatChar$idnr == codice0,]
 mindist=999999999999
 dist=mindist
 for(jj in 1:nn) {
  codice1 <- CatChar$idnr[jj]
  if (codice1 != codice0) {
   # calculate distance
   dist <- sqrt((val_catch_Char$xcor - CatChar$xcor[jj])^2 +
                (val_catch_Char$ycor - CatChar$ycor[jj])^2)
   if (dist < mindist) {
    finrow <- jj
    mindist <- dist
   }
  }
 }
 codice <- CatChar$idnr[finrow]

 # write on output matrix
 reg1_Par[j, 2] <- codice
 reg1_Par[j, 3] <- mindist
 reg1_Par[j, 4:18] <- CalibPar[CalibPar$idnr == codice, -1]
}
reg1_Par
```

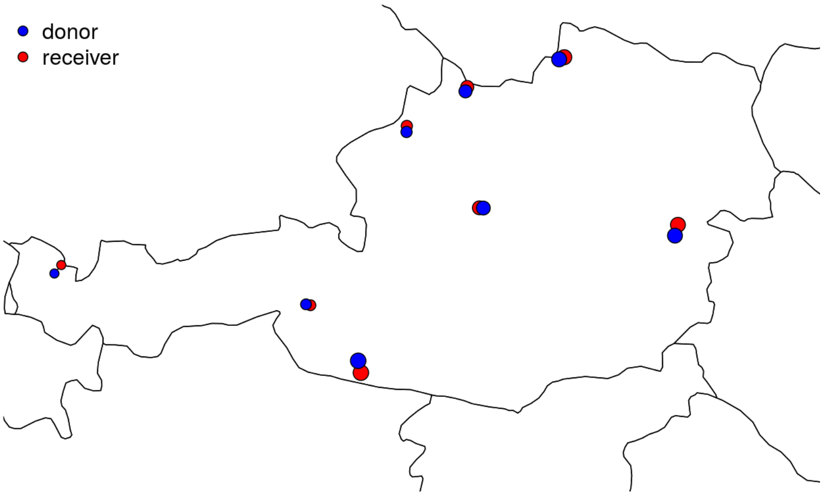
```
    idnr near_neigh     dist     csf     ddf      tr       ts    meltt   lprat        fc    beta      k0       k1        k2     lsuz   cperc     bmax   croute
1 200287     200253 8104.196 0.95933 1.55735 2.90374 -2.13614  0.25497 0.09661 386.45005 0.10731 0.40797  4.35917 222.11416 46.10512 1.16541  8.87345 10.88462
2 204768     204750 4450.742 0.90994 2.93373 1.97840 -0.76355 -0.50107 0.95223 164.26192 4.43013 0.97125  2.81979  57.17539 28.84704 3.05087  8.33758 16.03049
3 204925     204933 3424.860 1.06570 1.15609 1.62558  0.55725 -1.99035 0.98799 226.23669 3.99774 0.22569  8.49055  30.52141 20.03332 3.42047 19.74694 37.32982
4 205799     205831 2980.339 1.34143 0.74481 2.91433 -2.68848 -1.03169 0.36822  13.12582 0.11139 1.78671  4.71831  32.33420 70.14256 3.03799 29.47941 22.00100
5 208579     208462 4035.934 0.94404 2.03598 2.96013 -0.67501 -1.32535 0.93884 327.31442 3.96177 0.31914 12.57328 247.82423 17.24214 0.24981  7.31906 36.87668
6 208835     208819 8320.470 0.90000 1.38809 2.99974 -2.96651 -1.95742 0.13201 267.30728 0.85283 0.53284 12.52209  46.71228 31.51296 2.42351  6.74365 38.74399
7 212076     212068 3350.220 1.49997 1.28944 2.99962  0.99992  1.99941 0.58392 497.08337 0.00000 1.70191  4.89190 213.04929  8.55488 0.00055  9.61146 18.03325
8 212647     212324 9060.693 1.05807 1.37641 2.99655 -2.57137  1.59451 0.91767 287.50792 0.65991 0.04646 23.86294 167.74666  6.00049 1.07357  5.76709 32.81988
```

To store the results in a file do:

```
write.csv(reg1_Par, file="reg1_Par.csv")
```

Plot the nearest neighbor on a map:

```
donorsCatChar <- CatChar[CatChar$idnr %in% reg1_Par$near_neigh,]
plot(newMap, xlim=c(11.5, 15.5), ylim=c(46, 49))
 segments(x0=valCatChar$lon, y0=valCatChar$lat,
         x1=donorsCatChar$lon, y1=donorsCatChar$lat)
 points(valCatChar$lon, valCatChar$lat, pch=21, bg=2,
       cex=log10(valCatChar$area))
 points(donorsCatChar$lon, donorsCatChar$lat, pch=21, bg=4,
       cex=log10(valCatChar$area))
 legend("topleft", legend=c("donor","receiver"), pch=21, pt.bg=c(4,2), bty="n", cex=1.6)
```




Check the runoff model efficiency of the nearest neighbor regionalisation method for the basin analysed in Section 2:

```
idnr
```

```
[1] "200287"
```

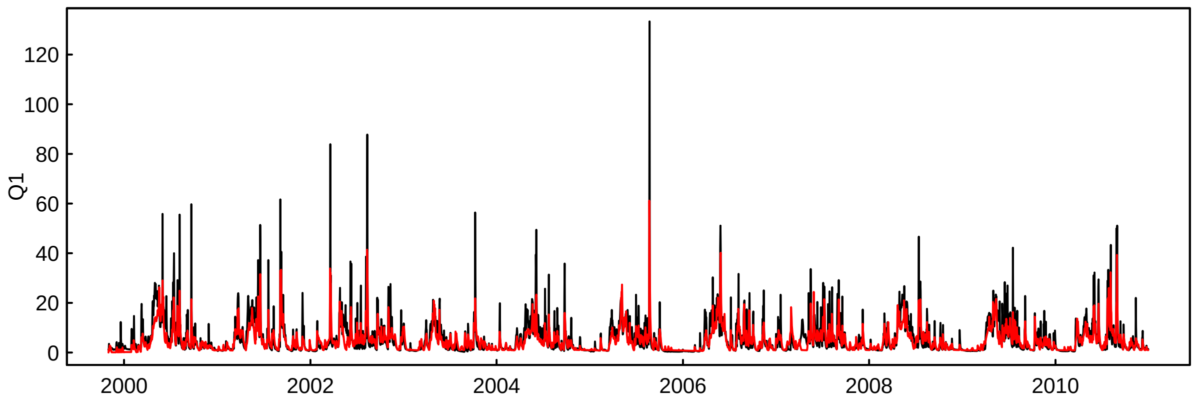
```
reg1_Par[reg1_Par$idnr == idnr,]
```

```
    idnr near_neigh     dist     csf     ddf      tr       ts   meltt   lprat     fc    beta      k0      k1       k2     lsuz   cperc    bmax   croute
1 200287     200253 8104.196 0.95933 1.55735 2.90374 -2.13614 0.25497 0.09661 386.45 0.10731 0.40797 4.35917 222.1142 46.10512 1.16541 8.87345 10.88462
```

```
reg_par <- reg1_Par[reg1_Par$idnr == idnr, 4:18]
simulation1_reg1 <- TUWmodel(prec=as.numeric(P1), airt=as.numeric(T1), ep=as.numeric(EP1), area=area,
                             param=reg_par)
sim_reg1 <- as.numeric(simulation1_reg1$q)
efficiencies_reg1 <- EMs(sim_reg1, as.numeric(Q1))
 efficiencies_reg1
```

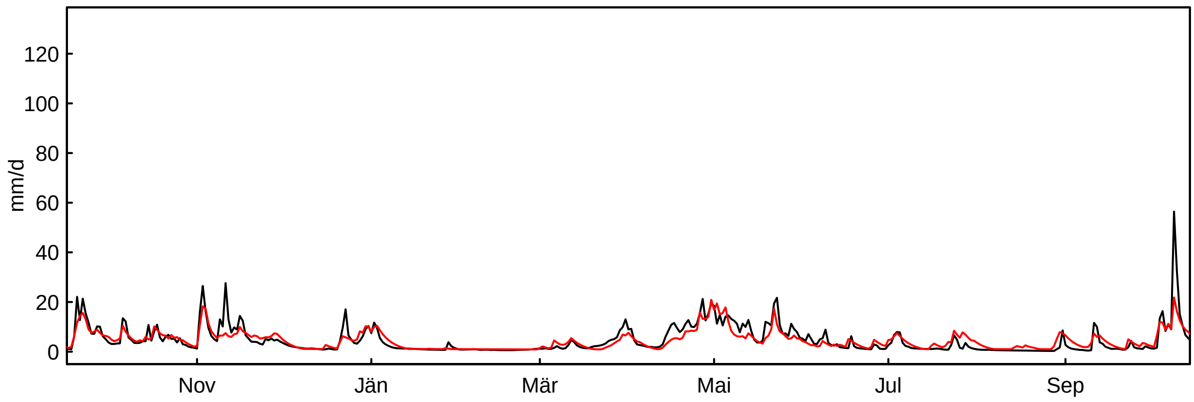
```
                RMSE (mm/d)            Nash efficiency ()          log Nash efficiency ()                  bias (mm/d)     mean absolute error (mm/d)
                  4.0739840                     0.6617605                       0.7454641                   -0.5689575                      2.1088008
mean absolute log error (mm/d)            volume error (%/%)
                  1.5233090                    -0.1057123
```

```
Qsim_reg1 <- zoo(sim_reg1, order.by=index(P1))
plot(Q1, xlab="") # observed discharge
lines(Qsim_reg1, col="red")  # simulated discharge
```




Zoom-in the plot for selected time period:

```
limit <-  as.Date(strptime(c("1 10 2002", "30 09 2003"), format="%d %m %Y"))
plot(Q1, xlab="", ylab="mm/d", xlim=limit)
lines(Qsim_reg1, col="red")
```




Lets run the model for the rest of validation basins and evaluate the efficiency of nearest neighbor regionalisation, e.g.

```
idnr='212076'
```

and repeat the code from:

```
reg1_Par[reg1_Par$idnr == idnr,]
```

onward.

# 4 Regionalisation of TUWmodel parameters - Similarity method

The main idea is to estimate/transfer model parameters to selected (validation) basins and run the model in there basins (which are considered as ungauged). The model parameters will be transferred (regionalized) from the most similar basin in terms of selected basin characteristics.

Normalisation of basin characteristics (find minimum and maximum of catchment attributes and their range):

```
catmin <- apply(CatChar[,-c(1:5)], 2, min)
catmax <- apply(CatChar[,-c(1:5)], 2, max)
catrange <- catmax - catmin
 catrange
```

```
      xcor       ycor       area       elev      slope     forest       xcen       ycen        map   aridity
528086.100 253994.940   6200.300   2620.000      0.516      0.959 504101.240 259077.160   1507.000     0.783
```

```
norm_char <- (CatChar[,-c(1:5)] - matrix(catmin, nrow=dim(CatChar)[1], ncol=10, byrow=TRUE))/
                                    (matrix(catrange, nrow=dim(CatChar)[1], ncol=10, byrow=TRUE))
head(norm_char, 20)
```

```
         xcor      ycor        area      elev     slope    forest       xcen      ycen       map   aridity
1  0.04221802 0.3101297 0.013192910 0.4973282 0.8294574 0.2638165 0.06958390 0.2876615 0.7120106 0.07151980
2  0.00000000 0.3289271 0.204393336 0.4923664 0.7480620 0.3211679 0.05704021 0.2305667 0.6197744 0.17369093
3  0.01331241 0.4097312 0.983242746 0.5312977 0.7015504 0.3211679 0.00000000 0.1283265 0.6197744 0.13026820
4  0.02742670 0.3862284 0.006031966 0.3145038 0.5271318 0.7591241 0.04229958 0.3429970 1.0000000 0.11111111
5  0.01917437 0.4091578 0.015999226 0.2251908 0.3759690 0.5881126 0.04262848 0.3610116 0.8473789 0.18390805
6  0.01861013 0.4034926 0.010289825 0.1003817 0.2364341 0.3180396 0.02809626 0.3476540 0.5520902 0.36526181
7  0.06720156 0.3337025 0.004515911 0.5305344 0.7848837 0.2221064 0.08757356 0.2968546 0.6834771 0.06896552
8  0.05703615 0.3537872 0.021869910 0.4675573 0.7189922 0.3222106 0.07562164 0.3111269 0.8029197 0.06641124
9  0.04413080 0.3660331 0.034659613 0.4335878 0.7034884 0.3712200 0.06892689 0.3176994 0.8725946 0.09067688
10 0.06726636 0.3775705 0.002806316 0.4446565 0.5852713 0.3774765 0.08413022 0.3418450 0.8951559 0.04597701
11 0.04488323 0.4419987 0.012321984 0.1748092 0.1414729 0.4702815 0.06120973 0.4309418 0.7889847 0.22477650
12 0.07998253 0.2972791 0.001854749 0.6774809 0.7015504 0.1897810 0.09873884 0.2609497 0.6907764 0.04214559
13 0.08000652 0.2984928 0.011386546 0.6473282 0.7093023 0.1345151 0.08825404 0.2663395 0.6257465 0.03703704
14 0.10221765 0.3116011 0.037772366 0.6263359 0.7674419 0.2377477 0.10252300 0.2731367 0.6071666 0.06130268
15 0.13814166 0.3642548 0.008112511 0.5320611 1.0000000 0.3222106 0.14144070 0.3332143 0.8228268 0.14176245
16 0.16357995 0.4185926 0.161040595 0.5492366 0.8255814 0.3774765 0.14266626 0.3109049 0.6423358 0.12643678
17 0.15521584 0.4424130 0.029740496 0.3732824 0.5426357 0.5589155 0.15656950 0.4067794 0.7730591 0.16730524
18 0.18835169 0.1374571 0.024353660 1.0000000 0.7112403 0.0000000 0.20094116 0.1014432 0.2183145 0.10472542
19 0.27091109 0.2102134 0.007193200 0.5908397 0.6647287 0.3618352 0.28690112 0.1759235 0.3934970 0.23243934
20 0.26770280 0.2436830 0.052916794 0.6229008 0.7732558 0.3357664 0.29339535 0.1906751 0.3696085 0.21200511
```

Give weights to different characteristics (do it manually), the order is like the names in `catrange`:

```
names(catrange)
```

```
[1] "xcor"    "ycor"    "area"    "elev"    "slope"   "forest"  "xcen"    "ycen"    "map"     "aridity"
```

```
#wcat <- c(0.,0.,1.,0.,0.,0.,0.,0.,0.,0.)   # area only
#wcat <- c(0.,0.,1.,1.,1.,1.,0.,0.,0.,0.)   # area, elevation, slope and forest
wcat <- c(0.,0.,1.,0.,0.,0.,0.,0.,1.,0.)   # area, map
```

Find for each validation basin the most similar one and write out the corresponding model parameters:

```
# initialise output matrix with idnr, similar catchment, normalised distance and 15 model parameters
reg2_Par <- as.data.frame(matrix(NA, nrow=length(val_catch), ncol=3 + 15,
                                 dimnames=list(1:length(val_catch),
                                               c("idnr", "sim_catch", "norm_dist", names(CalibPar)[-1]))))
 reg2_Par$idnr <- val_catch
# take just the weighted characteristics
norm_char0 <- norm_char*matrix(wcat, nrow=dim(norm_char)[1], ncol=10, byrow=TRUE)
# loop (notice that the distance refers to station coordinates)
nn <- dim(CatChar)[1]
for(j in 1:length(val_catch)) {
 codice0 <- val_catch[j]
 val_norm_char0 <- norm_char0[CatChar$idnr == codice0,]  # since norm_char0 and CatChar have corresponding rows
 mindist=999999999999
 dist=mindist
 for(jj in 1:nn) {
  codice1 <- CatChar$idnr[jj]
  if (codice1 != codice0) {
   dist <- sum(abs(val_norm_char0 - norm_char0[jj,]))
   if (dist < mindist) {
    finrow <- jj
    mindist <- dist
   }
  }
 }
 codice <- CatChar$idnr[finrow]

 # write on output matrix
 reg2_Par[j, 2] <- codice
 reg2_Par[j, 3] <- mindist
 reg2_Par[j, 4:18] <- CalibPar[CalibPar$idnr == codice, -1]
}
reg2_Par
```

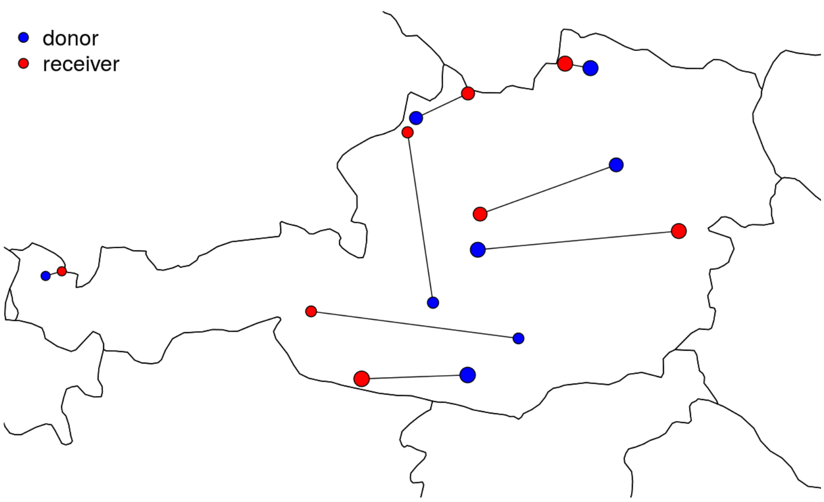
```
    idnr sim_catch   norm_dist     csf     ddf      tr       ts    meltt   lprat       fc     beta      k0       k1        k2     lsuz   cperc     bmax   croute
1 200287    200261 0.054414678 0.91929 1.41654 2.97734 -1.79608  0.62356 0.99713  43.0899  0.51049 0.37296  3.80247 248.66566 46.16550 1.11818 14.37807 49.53087
2 204768    213231 0.004292427 1.19971 0.90758 2.21939 -2.95576 -1.98335 0.98597 137.4294  2.63366 0.13245 16.47993 132.69854 55.90431 4.74312  6.19631 35.05620
3 204925    212928 0.009836029 0.90368 0.00284 1.39527 -2.83617 -0.96942 0.31636 147.0901  1.22114 0.63972 21.53337 148.35860 76.72764 2.40826 18.06992 43.99359
4 205799    210773 0.008499642 1.08722 1.46565 2.98234  0.47448 -0.28262 0.70554 571.0857  1.18230 0.00024 14.63760 177.59210 40.97107 2.21994  1.69635 11.89637
5 208579    208611 0.039943012 1.02303 2.79970 2.82893  0.80273 -0.34782 0.94253 250.6752  3.60770 0.42029  8.80873 188.81216 10.85431 0.40748  2.39693 36.86681
6 208835    204859 0.008223150 0.90093 1.76773 2.49232  0.71489 -1.07812 0.75035  99.9667  2.24901 1.95427  4.41615  69.83186 19.04614 1.34417  4.27061 10.83794
7 212076    203778 0.011757603 1.47305 1.20416 2.97474 -1.38782 -1.73772 0.99873 324.1216  0.63794 0.22258 29.79691 140.82283 43.03662 0.90281  6.02034 45.47951
8 212647    207852 0.011522507 0.90023 2.23578 2.98705 -2.99586 -1.39533 0.02908 112.6537 19.92992 0.54850  5.74910  55.73879 63.56282 6.58128 14.71361 25.07401
```

To store the results in a file do:

```
write.csv(reg2_Par, file="reg2_Par.csv")
```

Plot the nearest neighbor on a map:

```
donorsCatChar <- CatChar[CatChar$idnr %in% reg2_Par$sim_catch,]
plot(newMap, xlim=c(11.5, 15.5), ylim=c(46, 49))
 segments(x0=valCatChar$lon, y0=valCatChar$lat,
          x1=donorsCatChar$lon, y1=donorsCatChar$lat)
 points(valCatChar$lon, valCatChar$lat, pch=21, bg=2,
        cex=log10(valCatChar$area))
 points(donorsCatChar$lon, donorsCatChar$lat, pch=21, bg=4,
        cex=log10(valCatChar$area))
 legend("topleft", legend=c("donor","receiver"), pch=21, pt.bg=c(4,2), bty="n", cex=1.6)
```



Check the efficiency of the similarity regionalisation method for the catchment analised in Section 2:

```
idnr
```

```
[1] "200287"
```

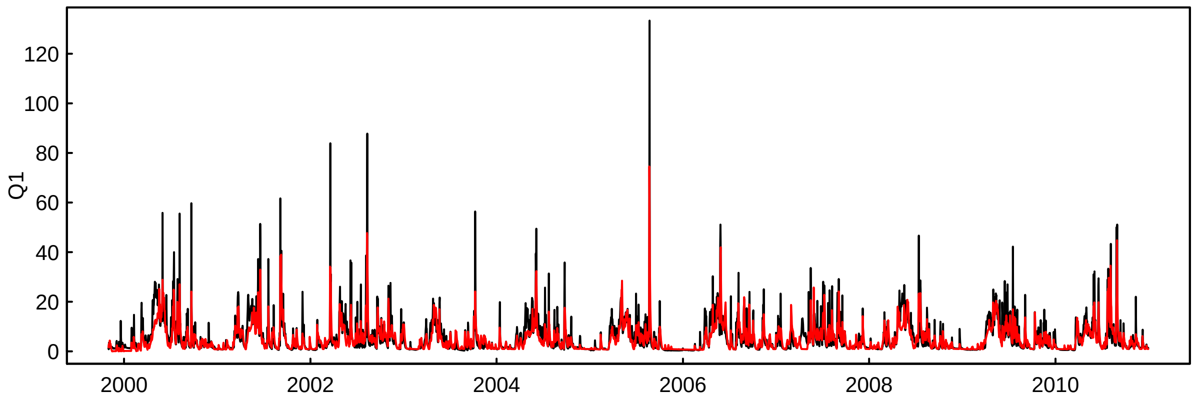
```
reg2_Par[reg2_Par$idnr == idnr,]
```

```
    idnr sim_catch  norm_dist     csf     ddf      tr       ts   meltt   lprat      fc    beta      k0      k1       k2    lsuz   cperc     bmax   croute
1 200287    200261 0.05441468 0.91929 1.41654 2.97734 -1.79608 0.62356 0.99713 43.0899 0.51049 0.37296 3.80247 248.6657 46.1655 1.11818 14.37807 49.53087
```

```
reg_par <- reg2_Par[reg2_Par$idnr == idnr, 4:18]
simulation1_reg2 <- TUWmodel(prec=as.numeric(P1), airt=as.numeric(T1), ep=as.numeric(EP1), area=area,
                             param=reg_par)
sim_reg2 <- as.numeric(simulation1_reg2$q)
efficiencies_reg2 <- EMs(sim_reg2, as.numeric(Q1))
 efficiencies_reg2
```

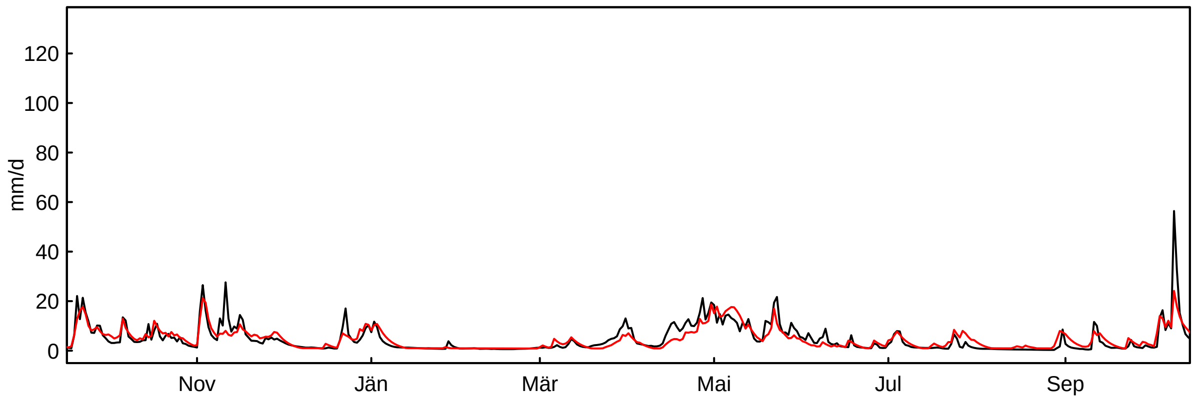
```
                RMSE (mm/d)           Nash efficiency ()        log Nash efficiency ()                 bias (mm/d)     mean absolute error (mm/d)
                 3.93502983                   0.68444019                    0.75378224                 -0.40150421                     2.08958833
mean absolute log error (mm/d)         volume error (%/%)
                 1.51500263                  -0.07459949
```

```
Qsim_reg2 <- zoo(sim_reg2, order.by=index(P1))
plot(Q1, xlab="") # observed discharge
lines(Qsim_reg2, col="red")  # simulated discharge
```




Zoom-in the plot for selected time period:

```
limit <-  as.Date(strptime(c("1 10 2002", "30 09 2003"), format="%d %m %Y"))
plot(Q1, xlab="", ylab="mm/d", xlim=limit)
lines(Qsim_reg2, col="red")
```




Lets repeat it for the other basins and compare it for different similarity definitions (weights) and compare it with the nearest neighbor approach.

# 5 Compare to the PUB book assessment

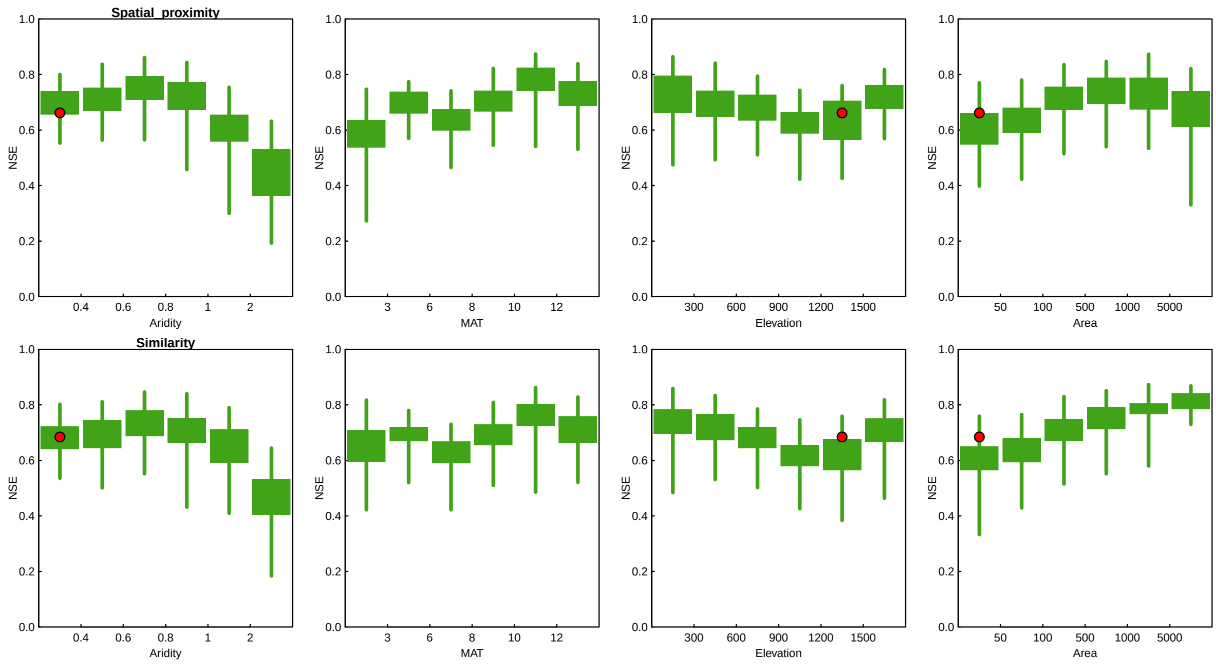
In the Level 2 Assessment of the PUB book (Blöschl et al., 2013) in Chapter 10 the Nasch Sutcliffe efficiency of regional studies is reported.

```
valCatChar[valCatChar$idnr == idnr,]
```

```
     idnr                station     river     lat     lon     xcor     ycor area elev slope forest     xcen     ycen  map aridity
10 200287 Schoenenbach_(Hengstig) Subersach 47.3864 10.0411 151606.5 392637.5 31.1 1460 0.328  0.383 152577.7 390048.9 1954   0.232
```

Fig 10.38 at page 267 of the book:

```
layout(matrix(1:8, nrow=2, byrow=FALSE))
plotPUBfiguresLevel2(chapter=10, method="Spatial_proximity", performance="NSE",
                     characteristic="Aridity", ylim=c(0,1),
                     main="Spatial_proximity")
 points(1, efficiencies_reg1[2], pch=21, bg=2, cex=2)  # because of 1st class of aridity
plotPUBfiguresLevel2(chapter=10, method="Similarity", performance="NSE",
                     characteristic="Aridity", ylim=c(0,1),
                     main="Similarity")
 points(1, efficiencies_reg2[2], pch=21, bg=2, cex=2)  # because of 1st class of aridity
plotPUBfiguresLevel2(chapter=10, method="Spatial_proximity", performance="NSE",
                     characteristic="MAT", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=10, method="Similarity", performance="NSE",
                     characteristic="MAT", ylim=c(0,1))
plotPUBfiguresLevel2(chapter=10, method="Spatial_proximity", performance="NSE",
                     characteristic="Elevation", ylim=c(0,1))
 points(5, efficiencies_reg1[2], pch=21, bg=2, cex=2)  # because of 5th class of elevation
plotPUBfiguresLevel2(chapter=10, method="Similarity", performance="NSE",
                     characteristic="Elevation", ylim=c(0,1))
 points(5, efficiencies_reg2[2], pch=21, bg=2, cex=2)  # because of 5th class of elevation
plotPUBfiguresLevel2(chapter=10, method="Spatial_proximity", performance="NSE",
                     characteristic="Area", ylim=c(0,1))
 points(1, efficiencies_reg1[2], pch=21, bg=2, cex=2)  # because of 1st class of area
plotPUBfiguresLevel2(chapter=10, method="Similarity", performance="NSE",
                     characteristic="Area", ylim=c(0,1))
 points(1, efficiencies_reg2[2], pch=21, bg=2, cex=2)  # because of 1st class of area
```




Lets plot the other basins (points) to the figure using the `points` command, e.g.:

```
points(3, efficiencies_reg1[2], pch=21, bg="blue", cex=2)  # because of 3rd class of aridity
```

# 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.

Parajka, J., V. Andréassian, S. A., Archfield, A. Bárdossy, G. Blöschl, F. Chiew, Q. Duan, A. Gelfan, K. Hlavčová, R. Merz, N. McIntyre, L. Oudin, C. Perrin, M. Rogger, J. L. Salinas, H. G. Savenije, J. O. Skøien, T. Wagener, E. Zehe and Y. Zhang (2013). Prediction of runoff hydrographs in ungauged basins. In *Runoff Prediction in Ungauged Basins: Synthesis Across Processes, Places and Scales*, University Press, Cambridge, 135-162, ISBN:9781107028180.

Viglione, A., J. Parajka, M. Rogger, J.L. Salinas, G. Laaha, M. Sivapalan and G. Blöschl (2013). Comparative assessment of predictions in ungauged basins - Part 3: Runoff signatures in Austria. *Hydrology and Earth System Sciences*, **17**, 2263–2279, doi:10.5194/hess-17-2263-2013.