Course Notes Advanced SWAT: setting up hydroPSO

Willem Vervoort

11-08-2017

Introduction

This document is heavily based on the original hydroPSO vignette from Rodrigo Rojas and Mauricio Zambrano-Bigiarini (2012). I have mainly added specific elements in relation to the calibration using ET. This means this document covers:

- Setting up the basic hydroPSO parameter files
- Defining the specific objective function and functions to run hydroPSO
- Running the model on Windows versus Linux
- Extracting the results and plotting
- Introducing the different ET functions
- rerunning the model and extracting the results

hydroPSO is initially tricky to work with, but once you have it set up, it is quite easy to change the model input and rerun the model, or to run different models. The package can take full advantage of parallel computing and therefore is perfect for use on High Performance Computing systems.

I strongly recommend keeping the Rojas and Zambrano-Bragiarini (2012) text, and particularly chapter 5, as a reference with this set of notes.

In a way, hydroPSO works similar to PEST. The idea is that you have to define a set of parameters and the range of these parameters, within you want to search for a solution. What you would like hydroPSO to do is find the location in the txtinout folder files where these parameters occur and rewrite the parameters for each iteration of the optimisation. So, this means we need to tell hydroPSO where to find the parameters in the files.

More generally, you would like to define the objective function (and you can write your own, or use one of the predefinded objective functions) and you would like a function that extracts the observed and predicted data from the model output and compares the results using the objective function.

See also Figure 16 in Rojas and Zambrano-Bragiarini (2012) on page 46, except that we are now using SWAT 2012 and we will use a different function than `rch2zoo()`.

Setting up the basic parameter files

This is related to page 47 and 48 in Rojas and Zambrano-Bragiarini (2012). In essence we need to develop parameter files for the calibration. The two files that you need are:

- Paramranges.txt: a file indicating the range of values that you want the optimisation to work with for each parameter (page 48 and page 52).
- ParamFiles.txt: a file indicating parameters to calibrate and where these parameters can be found in the different input files (page 48 and 52).

As you can see these files are quite difficult and involved files to type by hand, especially if you have many different subbasins. So I have again developed a script to help. The original script was again developed by Dipangkar Kundu for his PhD, and I have adjusted it to allow more flexibility in defining the different parameters (in particular to separate the files by landuse and soil). This script is called `parval.R` and is in your functions folder. We will discuss the different functions that are in the script while we look at how we develop the different files.

There are still two other files that we might need to develop. The first is required and we do by hand and here is an example of this file:

Table 1: Example input parameterfile to generate hydroPSO parameter files (continued below)

Parametername	Rownumber	colstart	colend	decimalplace	MinValue	MaxValue	Split
ALPHA_BF	5	1	16	7	0.01	1	ALL
CH_K1	28	4	16	3	0.01	50	ALL
CH_K2	7	8	15	3	0.01	100	ALL
CN2_FRST	11	9	17	5	34	95	FRST
CN2_FRPT	11	9	17	5	34	95	FRPT
CN2_AGRL	11	9	17	5	34	95	AGRL
CN2_MAPS	11	9	17	5	34	95	MAPS
EPCO_FRST	11	4	16	7	0.01	1	FRST
EPCO_FRPT	11	4	16	7	0.01	1	FRPT
EPCO_AGRL	11	4	16	7	0.01	1	AGRL

The second file is optional and can be developed using another little script. It is only needed if you have parameters that are split by landuse or soil. For example you might want to calibrate CN2 for AGRL separate from CN2 for GRAS. The code below writes this file and shows the content of the file. It simply extracts from all the management files the management code and the soil code and puts this together in a file.

```r
filelist <- list.files(pattern="mgt")

store <- data.frame(files = as.character(filelist))


for (i in 1:length(filelist)) {
 fline <- read.table(filelist[i], header=F, nrow=1)
 store$LU[i] <-  as.character(substr(fline[,7],6,9))
 store$Soil[i] <- as.character(fline[,9])
}
```

```r
head(store)
```

```
##           files   LU Soil
## 1 000010001.mgt FRST Mw15
## 2 000010002.mgt FRST Mw15
## 3 000010003.mgt FRST  Qb8
## 4 000010004.mgt FRST  Qb8
## 5 000010005.mgt AGRL Mw15
## 6 000010006.mgt AGRL Mw15
```

```r
write.table(store,"PSO.in/LUSoilmgtfiles.txt",row.names=F)
LU_soil <- store
```

Generating the paramranges file

To generate the parameter ranges files (`ParamRanges.txt`), we can use again some code. This requires the `parfile.txt` as an input

```r
# first read in th file with the parameter file extensions in SWAT
 Inparlist <- read.csv("../Inputdata/SWATParamFileExtensions.csv")
# Changed this so it can work from the master parameter file and extension list
 Masterparlist <- paste(Inparlist[,1],Inparlist[,2],sep="")


# Then read in the base parameter file
#
# # writing ParameterRanges file
f <- read.table("../Inputdata/parameterfiles/parfile_goodra.txt",
                 header = TRUE, colClasses=c("character",
                                             rep("numeric",6),
                                             "character"))

pm.file <- f[!duplicated(f$Parametername),]
pm.file <- pm.file[with(pm.file,order(Parametername)),] # ordering the data to work with the function
pm.file <- data.frame(ParameterNmbr = c(1:nrow(pm.file)),Parametername = pm.file$Parametername, MinValu
                     Maxvalue = pm.file$MaxValue)  # Just formating the table
write.table(pm.file, "../PSO_Goodra/PSO.in/ParamRanges.txt", sep = "\t",quote = FALSE, row.names = FALS
pander(pm.file[1:10,], caption="example of ParamRanges.txt file")
```

Table 3: example of ParamRanges.txt file

ParameterNmbr	Parametername	MinValue	Maxvalue
1	ALPHA_BF	0.01	1
2	CH_K1	0.01	50
3	CH_K2	0.01	100
4	CN2_AGRL	34	95
5	CN2_FRPT	34	95
6	CN2_FRST	34	95
7	CN2_MAPS	34	95
8	EPCO_AGRL	0.01	1
9	EPCO_FRPT	0.01	1
10	EPCO_FRST	0.01	1

This files contains the ranges of each of the parameters that we want to calibrate.

Paramfiles.txt

The final file we need to create is the file that describes all the files and the locations of the parameters For this we need to first load a set of functions that allow writing these different files.

```
source("../uruguaycourse/functions/parval.R")
```

We can then use a set of code to extract the right values and rewrite the files

```
 f <- f[with(f,order(Parametername)),]
 # separate out the relevant columns with locations in files
 f1 <- f[,c(2:5)]
# Changed this so it can work from the master parameter file and extension list
parlist <- create_parlist(f)


## Running the function

test <- file.prep(parlist,f1,split=LU_soil,txtinoutdir = getwd())
# reorganise columns
test <- test[,c(2,3,1,4:7)]
colnames(test)<- c("ParameterNmbr", "ParameterName","Filename","Row.Number", "Col.Start", "Col.End", "De



# Now write the file
write.table(test, "../Inputdata/ParamFiles.txt", sep = "\t",quote = FALSE, row.names = FALSE)
```