Environmental Modelling & Software 46 (2013) 292–298

Contents lists available at SciVerse ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

Short communication

Using a parallelized MCMC algorithm in R to identify appropriate likelihood functions for SWAT

J.F. Joseph [a,*], J.H.A. Guillaume [b]

[a] Department of Civil and Environmental Engineering, The University of Texas at San Antonio, One UTSA Circle, San Antonio, TX 78249, USA
[b] Integrated Catchment Assessment and Management (iCAM) Centre and National Centre for Groundwater Research and Training, Fenner School of Environment and Society, The Australian National University, Canberra ACT 2600, Australia

ARTICLE INFO

Article history:
Received 17 April 2012
Received in revised form
12 March 2013
Accepted 23 March 2013
Available online 20 April 2013

Keywords:
Likelihood function
Bayesian MCMC analysis
Parallel processing
SWAT
R
Uncertainty analysis
Parameter

ABSTRACT

Markov Chain Monte Carlo (MCMC) algorithms allow the analysis of parameter uncertainty. This analysis can inform the choice of appropriate likelihood functions, thereby advancing hydrologic modeling with improved parameter and quantity estimates and more reliable assessment of uncertainty. For long-running models, the Differential Evolution Adaptive Metropolis (DREAM) algorithm offers spectacular reductions in time required for MCMC analysis. This is partly due to multiple parameter sets being evaluated simultaneously. The ability to use this feature is hindered in models that have a large number of input files, such as SWAT. A conceptually simple, robust method for applying DREAM to SWAT in R is provided. The general approach is transferrable to any executable that reads input files. We provide this approach to reduce barriers to the use of MCMC algorithms and to promote the development of appropriate likelihood functions.

1. Introduction and motivation

The Soil and Water Assessment Tool (SWAT) is a basin-scale, semi-distributed, precipitation-runoff hydrologic model with open source code (Arnold et al., 1998). Over 1000 peer-reviewed publications utilize or review SWAT applications since its development in the early 1990s (http://swatmodel.tamu.edu/publications/peer-reviewed-publications/). Applications of SWAT include but are not limited to modeling water availability and water quality, stream channel erosion, plant growth, climate change impact, and comparing watershed management options.

SWAT contains non-linearities in its equations approximating the complexities of nature, and relies on input that inevitably contains errors. Since the work of Sorooshian and Dracup (1980) and Troutman (1983), the hydrologic community has been growing in awareness that, due to input and model inaccuracies, calibration and uncertainty analysis may lead to substantial bias in parameter and quantity estimates, and unreliable confidence intervals. As a theoretically sound remedy, Kavetski et al. (2006) propose a Markov chain Monte Carlo (MCMC) analysis within a Bayesian framework. Central to this analysis is the development of a likelihood function which realistically models residuals.

The MCMC approach originates in the seminal work by Metropolis et al. (1953) in the field of Physics and was generalized for other fields by Hastings (1970) as the Metropolis–Hastings (M–H) MCMC algorithm. MCMC has been growing in popularity (Diaconis, 2009). Yang et al. applied the traditional M–H MCMC algorithm to SWAT models of the Thur Basin in Switzerland (2007) and the Chaohe Basin in China (2008) with likelihood functions based on rigorous analysis of residuals.

Yet the application of MCMC approaches to SWAT has remained absent from the peer-reviewed literature since the work of Yang et al. (2007, 2008). As SWAT typically requires a fraction of a minute to several minutes per simulation, and traditional MCMC algorithms require a very high number of simulations for convergence, the application of MCMC to SWAT has required prohibitively long computational time. Before the algorithm converges, estimates of the probability distribution of the model parameters are likely to be unreliable. Yang et al. (2007, 2008) needed to use the Shuffled Complex Evolution-University of Arizona (SCE-UA) algorithm (Duan et al., 1992) to find an initial estimate of the global maximum as their starting point before applying M–H MCMC for

* Corresponding author.
E-mail address: john.joseph@utsa.edu (J.F. Joseph).

1364-8152/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.envsoft.2013.03.012

their SWAT models. With such difficulties in applying the Bayesian MCMC framework, progress has been lacking in the development of appropriate likelihood functions. Smith et al. (2010) have noted particular need for improvement in developing a likelihood function for even a single ephemeral catchment. Joseph (2011) identifies several challenges remaining to develop an appropriate likelihood function for two intermittent catchments.

The problem of the high number of simulations required in the M–H MCMC algorithm is certainly not limited to the SWAT model, and has precipitated various other MCMC algorithms that have appreciably accelerated movement of the MCMC chain towards equilibrium. Nonetheless, the problem of prohibitively high numbers of simulations has yet to be surmounted within the SWAT community. The MCMC algorithm that is most widely available to SWAT practitioners is the M–H MCMC algorithm itself, coded by Yang et al. (2007) for incorporation along with several other algorithms into the SWAT Calibration and Uncertainty Analysis Programs (SWAT-CUP) public domain software available through the Swiss Institute of Aquatic Science and Technology (http://www.eawag.ch/forschung/siam/software/swat/index_EN). SWAT user support group postings relating to the MCMC option in SWAT-CUP are few in number compared to those regarding the other (non-MCMC) algorithms, and these often refer merely to failed attempts at its usage.

The Delayed Rejection and Adaptive Metropolis (DRAM) MCMC algorithm is also now available to SWAT users through the R Flexible Modeling Environment (FME) package (Soetaert and Petzoldt, 2010). Wu and Liu (2012) allow SWAT to be called from R as a linked library and wrap it with FME, and this allows for the application of DRAM to SWAT. DRAM has been shown to have superior performance to that of M–H MCMC (Haario et al., 2006) for an algae growth model. However, a comparison (Vrugt et al., 2009) for the 13-parameter Sacramento-Soil Moisture Accounting (SAC-SMA) hydrologic model (Burnash, 1995) suggests that the improvement of DRAM over the more traditional MCMC is not sufficiently dramatic to attract SWAT users to DRAM, though they may be attracted to FME for its other algorithms and tools.

An MCMC algorithm that does offer a dramatic improvement over M–H MCMC is the Differential Evolution Adaptive Metropolis (DREAM) algorithm (Vrugt et al., 2009), When run in its parallelized version such as that of the MatLab code written by its developer, DREAM generates multiple Markov chains in parallel, which increases the parameter space explored and can speed up convergence. The cross-referencing reduces the number of simulations compared to that of DRAM and, as each chain may be assigned to a node, the total time for the MCMC analysis is spectacularly reduced compared to that of DRAM. DREAM has been run in parallel for a variety of models in the literature (e.g., He et al., 2011; Huisman et al., 2010). Vrugt et al. (2009) illustrate that DREAM identifies a global optimum RMSE more reliably than Random Walk Metropolis MCMC, DRAM, and SCE-UA, and requires fewer simulations. In addition, a near linear speed-up can often be achieved in methods that run the model simultaneously in parallel for parameter values in multiple chains (Vrugt et al., 2006), and MCMC methods may therefore be designed with this capability in mind (Laloy and Vrugt, 2012). Based on our own experience, with an 8-parameter calibration of a SWAT model of subbasin I of the Little River Experimental Watershed in Georgia, USA for an 8-thread desktop computer requiring 15 s per simulation, DREAM is able to reach completion in 12 h. For comparison, SCE-UA requires 18 h, does not provide estimates of posterior densities and therefore does not lend itself to the development of appropriate likelihood functions. Running DREAM without parallelization would have required 4 days, based on linear extrapolation, and was therefore considered impractical.

There are few examples of SWAT being run in parallel. Whittaker (2004) and Whittaker et al. (2010) have applied an unpublished approach to modify SWAT parameters in R by editing and recompiling the SWAT source code with a Linux operating system. Rouholahnejad et al. (2012) developed proprietary software in which a non-MCMC algorithm is parallelized for SWAT for Windows. Confesor and Whittaker (2007) and Zhang et al. (2012) have also applied parallel processing to SWAT. In any case, no application of a parallelized MCMC algorithm for SWAT has been found in the literature, let alone one that is open source and requires no proprietary software. As will be discussed in the following section, the general lack of parallel algorithm application to SWAT may be due to the high number of diverse input files that must be edited with each simulation.

The motivation of this paper is to present a reliable method for applying DREAM to SWAT in R, using tools to which practitioners have immediate access, while leaving them full freedom in specifying the likelihood function. We hope that this free access will aid in the development of appropriate likelihood functions, and thus more reliable confidence intervals and a correspondingly deeper scientific understanding of hydrology-related dynamics. The commonly used Windows 7 operating system was used, but a similar procedure should be possible in other operating systems.

Recently, a modified DREAM algorithm, MT-DREAM$_{(ZS)}$, has been developed and shown to be more efficient than DREAM in at least some contexts (Laloy and Vrugt, 2012), though not compared for SWAT. Additionally, unlike running DREAM in parallel, MT-DREAM$_{(ZS)}$ maintains detailed balance and reversibility throughout the execution process and is thus more correct theoretically (Laloy and Vrugt, 2012). The translation of MT-DREAM$_{(ZS)}$ from MATLAB to R is not yet achieved. However, the only obstacle to applying SWAT to DREAM in R has been the parallelizing aspect of DREAM, and, as this is the only obstacle anticipated for MT-DREAM$_{(ZS)}$ as well, we expect the method presented in this paper to be useful in applying MT-DREAM$_{(ZS)}$ to SWAT in R.

2. A conceptual description of the parallelization approach

DREAM runs a user-defined number of parallel chains and shares information among chains so that the proposal densities of randomized jumps are adjusted, resulting in accelerated movement of the MCMC chains towards equilibrium (Vrugt et al., 2009). Within each iteration, the model must be run for a set of parameters in each chain. These model runs are however entirely independent, which allows them to be run simultaneously, making use of the multiple processors, cores, or threads on a modern computer. To apply DREAM to SWAT in R, we propose establishing a separate folder for each chain, so that each chain has its own execution environment.

Our proposal springs from the fact that a SWAT project typically consists of hundreds or thousands of input parameter and data files to represent the multitude of hydrologic response units (HRUs) in the subbasins spatially distributed throughout the basin, and that these files must be edited and read before each simulation in an autocalibration process. Parameter values may vary in each HRU or subbasin, and, for model calibration and uncertainty analysis, a multiplier or addend or replacement will be associated with each default or best-guess value of the parameter. For example, a multiplier of 1.05 may be applied to the initially estimated permeabilities of all soil layers in each HRU, or to only the top three soil layers in the eastern half of the basin. These multipliers and addends and replacements themselves can thus be thought of as parameters, and throughout the remainder of this paper will be referred to as “adjustment parameters”. These adjustment parameters for the SWAT model include a channel roughness adjustment parameter, a soil hydraulic conductivity adjustment parameter, an adjustment parameter for the maximum volume of rainfall to be intercepted by the canopy, and several dozen others. A more

detailed description of these adjustment parameters may be found on pp. 40–43 of Abbaspour (2011), and the origin of their development may be found in Yang et al. (2005). Each adjustment parameter value must be used to re-write each of its many corresponding input files. This cumbersome process is handled by calling SWAT_Edit.exe, a component of the public domain SWAT-CUP software. To ensure input file edits and model execution occur as a single atomic action, commands are placed in a batch file. The R script calls this batch file, which in turn executes SWAT_Edit.exe and the SWAT model. Using this editing process within a parallel algorithm such as DREAM in R for Windows is facilitated by having a separate execution environment for each chain. A folder containing SWAT_Edit.exe, the SWAT model executable, and all input files provides the separate environment for each chain, and one batch file for each folder links DREAM to each of these environments.

Our strategy requires an R implementation of DREAM (Guillaume and Andrews, 2011) that includes a setting named "snow.chains" (http://CRAN.R-project.org/package=dream) which allows us to keep track of chains separately while running the models in parallel using the Simple Network of Workstations (snow) package (Tierney et al., 2011). The general approach is transferrable to any executable that reads input files. This approach is intended for use on a single machine with multiple threads or processors, not for application of DREAM to SWAT on a network of machines, or on a supercomputing cluster (e.g., Vrugt et al., 2006). The availability of 8 or more threads on relatively inexpensive desktops or workstations today ensures immediate opportunities for usage of the software, without the additional barriers of accessing and learning how to use specialized computing facilities.

3. Steps to applying DREAM to SWAT in R

Before applying DREAM to SWAT with our methodology, R is to be downloaded and installed (R Development Core Team, 2011), and then the "snow" and "dream" packages are to be installed through the "Install package(s)…" option under the "Packages" tab on the R Graphical User Interface (RGui). Also, good modeling practice dictates that parameters and their ranges should be first established by literature review, experienced judgement, and reasonable manual calibration before submission to autocalibration. The steps for applying DREAM to SWAT in R are then as follows:

1. After creating a SWAT project through any interface available at the SWAT home page (http://swatmodel.tamu.edu/), open the *TxtInOut* project folder, create a new folder named "Backup" within it, and copy all files in the *TxtInOut* folder into this Backup folder. This step is required for use of the editing software SWAT_Edit.exe that is to be borrowed from SWAT-CUP. Place the *TxtInOut* folder (with the Backup folder within it) into an appropriately named SWAT DREAM project folder. In the example presented in the next section, DREAM is applied to a SWAT model of subbasin *I* of the Little River Experimental Watershed (LREW) in Georgia, USA, so we name our folder *SWAT_DREAM_LREW*.
2. Place the file of the observed daily discharge values (or of other observed time series) into the first level of this SWAT DREAM project folder, i.e. inside of the SWAT DREAM project folder but outside of the *TxtInOut* folder. In our example, the file is of observed daily discharge and is named "Qo.txt".
3. Copy the SWAT executable model "swat2009.exe" from the ArcSWAT or MWSWAT directory, or from the SWAT website (http://swatmodel.tamu.edu/software/swat-model/), and paste it into the first level of the *TxtInOut* folder.
4. Copy "SWAT_Edit.exe" and "Absolute_SWAT_Values.txt" from the "SourceData/Shared" folder of public domain software SWAT-CUP4 (http://www.neprashtechnology.ca/Downloads.aspx) and paste them into the first level of the *TxtInOut* folder.
5. Launch R, change the directory to that of the SWAT DREAM project folder by navigating to it through the "Change dir…" option under the "File" tab of the RGui menu bar. Begin a "New Script" (under the "File" tab) and copy and paste the script of Appendix A into the R-Editor screen that appears. Modify the highlighted portions as needed to accommodate folder names and locations. To execute this script, highlight it, and then click on the "Run line or selection" icon on the menu bar. This will generate all the folders and batch files needed to apply DREAM to SWAT in R. Save the file using the familiar "File" and "Save as…" Close the script.
6. As in step 5, begin a "New Script". Copy and paste the R script of Appendix B. Modify the highlighted portions for the particular project as guided by the comment lines in italics. Executing the script will run the DREAM algorithm.

Noteworthy in the script is that the user may directly modify the likelihood function.

Descriptions of various plots and statistical output options for DREAM is also provided in the R DREAM package documentation, which may be accessed with the command *"help(dream)"*. Additional R code for plotting and comparing simulated and observed streamflow and for examining residuals may also be found in the Appendices of Joseph (2011).

4. Example of flexibility in likelihood function development

The literature has already provided comparisons of the performance of the DREAM algorithm to other MCMC algorithms and SCE-UA (Vrugt et al., 2009), comparisons of the adequacy of various likelihood functions (Yang et al., 2007; Schoups and Vrugt, 2010; Smith et al., 2010), and quantitative evidence of the advantages of parallelization of MCMC algorithms in general (Laloy and Vrugt, 2012; Vrugt et al., 2006). We instead present the following example to demonstrate the importance of developing an appropriate likelihood function, and how one might be sought while applying DREAM to SWAT in R.

Two scripts like that in Appendix B were executed. The first used a conventional Gaussian-based likelihood function. The second incorporates the much less frequently utilized Laplace distribution (Schoups and Vrugt, 2010), as shown in the last three highlighted lines of Appendix B. For both the Gaussian and the Laplace distributions, the true mean of the residuals was set equal to zero, and the scale parameters were estimated from the full data set of residuals for each simulation. A synthetic version of subbasin I of the Little River Experimental Watershed was used as the basin to be modeled, with realistic precipitation measurement errors based on a comparison between an isolated rain gauge and a dense rain gauge network (Joseph, 2011). The use of synthetic data for testing likelihood functions is common in the literature (e.g., Smith et al., 2010).

For both likelihood functions, the histogram of the SWAT model residuals formed a sharp peak near the center, and then exponentially decayed on either side, as shown in Fig. 1 (Note: Ten outliers were removed when graphing the histogram in order to provide a visually manageable scale.). The form is more similar to that of the Laplace distribution (with its theoretical density curve shown in solid black) than it is to that of the normal distribution (dotted black). Fig. 1 suggests that the Laplace distribution provides a better representation of residuals than does the Gaussian distribution, though a more rigorous comparison by utilizing quantile–quantile plots and other tools would be warranted for a complete study. Encouragingly, the use of the Laplace distribution

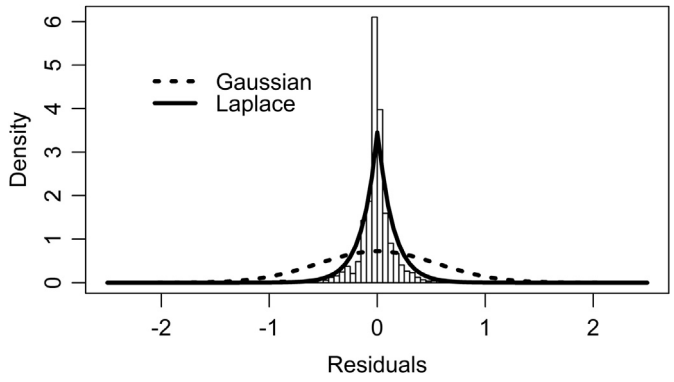

Fig. 1. Histogram of residuals of SWAT model compared with theoretical densities of Gaussian and Laplace distributions.

reduced the error in the hydrologic quantity estimates by an average of 42% when the various hydrologic quantity estimates are compared with the observed hydrologic quantities of the synthetic watershed (including surface runoff, lateral flow, groundwater recharge, shallow groundwater flow, percolation, evapotranspiration, and streamflow). There is likely still room for further improvement. The residuals are substantially more concentrated around zero than even the double exponential distribution would suggest, and the autocorrelation of the residuals (identified as significant for this site by Joseph [2011]) would need to be addressed. A rigorous analysis and modeling of the residuals is beyond the scope of this paper, but may be one of many future studies facilitated by the robust and simple application of DREAM (or such algorithms as MT-DREAM$_{(ZS)}$) to SWAT presented in this paper.

Example scripts similar to that of Appendix B, as well as corresponding SWAT files and all necessary software to run examples are at http://dream.r-forge.r-project.org. Examples include the use of a Box–Cox transformation parameter in a Gaussian likelihood function, with the script modified to account for the parameter being external to the SWAT model. The script is also shown to accommodate parameters in excess of the number of machine threads and MCMC chains (in our experience, DREAM is very effective with SWAT even if the number of MCMC chains is somewhat fewer than the number of parameters.). Examples of applying DREAM to SWAT models of the actual subbasin I of LREW may be obtained from the corresponding author.

5. Summary

SWAT and the MCMC approach to calibration and uncertainty analysis enjoy increasingly widespread usage, but the application of MCMC to SWAT has been hindered by the prohibitively high number of simulations and computational time required. The DREAM algorithm is able to reduce the number of simulations required compared to other MCMC algorithms, and it utilizes parallel Markov chains, each of which may be run concurrently, to reduce the time requirement by an order of magnitude or more, depending on the number of parallel chains. Applying a Bayesian MCMC approach to SWAT models, which may require a fraction of a minute to several minutes for a single simulation, and tens of thousands of simulations for convergence, then becomes feasible. Other models with high computational time will see similar benefits.

Application of DREAM to SWAT in R requires input data stored in hundreds or thousands of files be edited for parameter adjustments before each simulation in parallel. This paper provides a conceptually simple, robust means for achieving this parallelization. The general approach is applicable to any executable that reads input files, though the details of application to executables other than SWAT and SWAT_Edit are beyond the scope of this paper.

We hope that this paper, by making the DREAM algorithm more accessible, helps modelers, and in particular, the SWAT community, apply and develop likelihood functions that lead to less biased parameter estimates and more reliable confidence intervals, and to thereby advance hydrologic and other environmental modeling.

Appendix A. R script for generating folders and batch files to be used by DREAM

```
# Create vectors of names to be assigned to each folder and to each batch file. A batch file and
# corresponding folder is needed for adjustment parameters and running simulations within each
# chain.
folder_name<-c("LREW1","LREW2","LREW3","LREW4","LREW5","LREW6","LREW7","LREW8")
bfname<-c("runLREW1.bat","runLREW2.bat", "runLREW3.bat","runLREW4.bat",
          "runLREW5.bat","runLREW6.bat","runLREW7.bat","runLREW8.bat")

# Define a template for the text of each batch file, and write one batch file for each folder to be
# created.
batch.text.template<-'@echo off
C:
cd "C:\\Users\\John\\Documents\\SWAT_DREAM_LREW\\!FOLDER_NAME!"
start /w SWAT_Edit.exe
swat2009.exe
if %errorlevel% == 0 exit 0
echo.'
for (i.batch in 1:length(folder_name)){
this.folder.name<- folder_name[i.batch]
this.bfname<- bfname[i.batch]
```

```
# Generate a uniquely named folder for each chain.
dir.create(this.folder.name)
file.copy(dir("TxtInOut",full.names=T),this.folder.name,recursive=TRUE)

# Replace this.folder.name into batch.text.template
batch.text<-gsub("!FOLDER_NAME!",this.folder.name, batch.text.template)
write(batch.text,this.bfname)
}
#
################################## E N D ##############################
```

Appendix B. R script for execution of DREAM algorithm to SWAT

```
#
# R script for application of DREAM to SWAT                    Joseph Guillaume
#                                                              John Joseph
#                                                              March 2012
#
# Load dream. This line need not be executed if dream has already been loaded in the R session.
#
library(dream)
#
# Load and start the SNOW parallel processing library. This line need not be executed if snow
# has already been loaded in the R session. Also, set the number of nodes equal to the number
# of MCMC chains to be used.
#
library(snow)
cl<-makeCluster(8,type="SOCK")
#
# The number of nodes and MCMC chains (nseq) is set at 8.
# The maximum number of simulations (ndraw) is limited to 24,000.
# The "snow.chains" method of parallelization should be used (parallel).
# The convergence criteria is left at its default value of control$Rthres=1.01.
# The modeler may consider limiting the number of simulations to a relatively low number until
# discovering how well the chains converge for the particular model and likelihood function.
# We found that when the Gaussian likelihood function was used for our SWAT model the
# convergence criteria of control$R.thres=1.2 recommended by Vrugt et al (2009) was usually
# satisfied within 24,000 simulations, whereas with the Laplace likelihood function it was not.
# However, best-estimates of parameters did not change significantly after 24,000
# simulations for either likelihood function.
#
control<-list(nseq=8,ndraw=24000,parallel="snow.chains")
#
# The SWAT adjustment parameters and their ranges are shown below for our example. See pp. 40-43
# of the SWAT-CUP4 user manual (Abbaspour, 2011) for additional guidance. These parameters must
# appear in the same order as they do in the "model.in.rows"statement.
#
pars<-list("r__CN2.mgt"=c(-0.25,0.25),"v__ESCO.hru"=c(0.9,1.0),"v__GW_REVAP.gw"=c(0.02,0.2),
   "v__CANMX.hru"=c(0.1,10),"v__GW_DELAY.gw"=c(1,50),"r__SOL_Z().sol"=c(-0.25,0.25),
   "v__GWQMN.gw"=c(0,2000),"r__SOL_AWC().sol"=c(-0.25,0.25))
#
# Define the function that will be called by DREAM for every parameter set in every chain.
#
callSWAT<-function(id,pars){
model.in.rows<- c("r__CN2.mgt","v__ESCO.hru","v__GW_REVAP.gw","v__CANMX.hru",
   "v__GW_DELAY.gw","r__SOL_Z().sol","v__GWQMN.gw","r__SOL_AWC().sol")
model.in.file <-as.character(c("LREW1/model.in","LREW2/model.in",
                  "LREW3/model.in","LREW4/model.in","LREW5/model.in",
                  "LREW6/model.in","LREW7/model.in","LREW8/model.in"))
```

```
output.rch.file<-as.character(c("LREW1/output.rch","LREW2/output.rch",
                  "LREW3/output.rch","LREW4/output.rch","LREW5/output.rch",
                  "LREW6/output.rch","LREW7/output.rch","LREW8/output.rch"))
#   Extract values from the observed data file to create the vector of observed values,
#   and create vector of names of the batch files that will execute SWAT and SWAT_Edit.
Qo<-read.table("Qo.txt")[,1]
bfname<-as.character(c("runLREW1.bat","runLREW2.bat",
                  "runLREW3.bat","runLREW4.bat","runLREW5.bat",
                  "runLREW6.bat","runLREW7.bat","runLREW8.bat"))
#   From the parameters given by DREAM, select out the parameter set for this chain.
this.par.set=pars[id,]
#   Set the parameters in SWAT for this instance.
this.file<-model.in.file[id]
output.file<-output.rch.file[id]
write.table(this.par.set,this.file,
        quote=FALSE,row.names=model.in.rows,col.names=FALSE)
# Run the batch script that modifies the parameters and runs the model.
system(bfname[id])
# Read the model output.
Qsim.data<-read.table(output.file,skip=9)
Qs<-Qsim.data[Qsim.data$V2==7,6] #The "7" here is the basin number of observed discharge data.  The
#                 "6" is the column number in which the corresponding simulated discharge appears in
#                 SWAT's "output.rch", and may vary depending on options specified in  SWAT's "file.cio".
# Calculate the log likelihood.  Here we are basing the likelihood function on the
# Laplace distribution, with 0 as the location parameter and the maximum likelihood estimator for the
scale parameter
res<-Qo-Qs
mean.res<-mean(res)
mean.abs.res<-mean(abs(res))
logp<-sum(sapply(res,function(res_k)
        log((1/(2*mean.abs.res))*exp(-abs(res_k)/(mean.abs.res)))))
return(logp)
}
#
dream.object<-dream(
        FUN=callSWAT,
        pars=pars,
        func.type="logposterior.density",
        control=control)
summary(dream.object)
## Stop SNOW parallelisation
stopCluster(cl)
#
################################## E N D ##############################
```

References

Abbaspour, K., 2011. SWAT-CUP4: SWAT Calibration and Uncertainty Programs – A User Manual. Swiss Federal Institute of Aquatic Science and Technology.

Arnold, J.G., Srinivasan, R., Muttiah, R.S., Williams, J.R., 1998. Large-area hydrologic modeling and assessment: part I. Model development. Journal of the American Water Resources Association 34 (1), 73–89.

Burnash, R.J.C., 1995. The NWS river forecast system-catchment modeling. In: Singh, V.P. (Ed.), Computer Models of Watershed Hydrology. Water Resour. Publ., Littleton, Colo, pp. 311–366.

Confesor, R.B., Whittaker, G.W., 2007. Automatic calibration of hydrologic models with multi-objective evolutionary algorithm and Pareto optimization. Journal of the American Water Resources Association 43 (4), 981–989.

Diaconis, P., 2009. The Markov chain Monte Carlo revolution. Bulletin of the American Mathematical Society 46 (2), 179–205.

Duan, Q.Y., Sorooshian, S., Gupta, V., 1992. Effective and efficient global optimization for conceptual rainfall–runoff models. Water Resources Research 28 (4), 1015–1031.

Guillaume, J.H.A., Andrews, F.T., 2011. Dream: DiffeRential Evolution Adaptive Metropolis. R Package Version 0.4. https://r-forge.r-project.org/projects/dream/.

Haario, H., Laine, M., Mira, A., Saksman, E., 2006. DRAM: efficient adaptive MCMC. Statistical Computing 16, 339–354. http://dx.doi.org/10.1007/s11222-006-9438-0.

Hastings, W.K., 1970. Monte Carlo sampling methods using Markov chains and their applications. Biometrika 57 (1), 97–109.

He, M., Hogue, T.S., Franz, K.J., Margulis, S.A., Vrugt, J.A., 2011. Corruption of parameter behavior and regionalization by model and forcing data errors: a Bayesian example using the SNOW17 model. Water Resources Research 47, W07546.

Huisman, J.A., Rings, J., Vrugt, J.A., Sorg, J., Vereecken, H., 2010. Hydraulic properties of a model dike from coupled Bayesian and multi-criteria hydrogeophysical inversion. Journal of Hydrology 380 (2010), 62–73.

Joseph, J.F., 2011. Preliminaries to Watershed Instrumentation System Design. PhD Dissertation. University of Texas at San Antonio, Department of Civil and Environmental Engineering, San Antonio, Antonio, Texas.

Kavetski, D., Kuczera, G., Franks, S.W., 2006. Bayesian analysis of input uncertainty in hydrological modeling: 1. Theory. Water Resources Research 42, W03407.

Laloy, E., Vrugt, J.A., 2012. High-dimensional posterior exploration of hydrologic models using multiple-try DREAM$_{(ZS)}$ and high-performance computing. Water Resources Research 48, W01526. http://dx.doi.org/10.1029/2011WR010608.

Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E., 1953. Equation of state calculations by fast computing machines. The Journal of Chemical Physics 21, 1087–1092.

R Development Core Team, 2011. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0. URL: http://www.R-project.org/.

Rouholahnejad, E., Abbaspour, K.C., Vejdani, M., Srinivasan, R., Schulin, R., Lehman, A., 2012. A parallelization framework for calibration of hydrological models. Environmental Modelling & Software 31, 28–36.

Schoups, G., Vrugt, J.A., 2010. A formal likelihood function for parameter and predictive inference of hydrologic models with correlated, heteroscedastic, and non-Gaussian errors. Water Resources Research 46, W10531.

Smith, T., Ashish, S., Marshall, L., Mehrotra, R., Sisson, S., 2010. Development of formal likelihood function for improved Bayesian inference of ephemeral catchments, catchments. Water Resources Research 46, W12551.

Soetaert, K., Petzoldt, T., 2010. Inverse modelling, sensitivity and Monte Carlo analysis in R using package FME. Journal of Statistical Software 33 (3), 1–28.

Sorooshian, S., Dracup, J.A., 1980. Stochastic parameter estimation procedures for hydrologic rainfall–runoff models: correlated and heteroscedastic error cases. Water Resources Research 16 (2), 430–442. http://dx.doi.org/10.1029/WR016i002p00430.

Tierney, L., Rossini, A.J., Li, N., Sevcikova, H., 2011. Snow: Simple Network of Workstations. R Package Version 0.3-8. http://CRAN.R-project.org/package=snow.

Troutman, B.M., 1983. Runoff prediction errors and bias in parameter estimation induced by spatial variability of precipitation. Water Resources Research 19 (6), 791–810.

Vrugt, J.A., Nullain, B.O., Robinson, B.A., Bouten, W., Dekker, S.C., Sloot, P.M.A., 2006. Application of parallel computing to stochastic parameter estimation in environmental models. Computers & Geosciences 32, 1139–1155. http://dx.doi.org/10.1016/j.cageo.2005.10.015.

Vrugt, J.A., ter Braak, C.J.F., Diks, C.G.H., Robinson, B.A., Hyman, J.M., Higdon, D., 2009. Accelerating Markov Chain Monte Carlo simulation by differential evolution with self-adaptive randomized subspace sampling. International Journal of Nonlinear Sciences & Numerical Simulation 10, 273–290. http://dx.doi.org/10.1515/IJNSNS.2009.10.3.273.

Whittaker, G., Confesor, R., Di Luzio, M., Arnold, J.G., 2010. Detection of overparameterization and overfitting in an automatic calibration of SWAT. Transactions of the ASABE 43 (5), 1487–1499.

Whittaker, G., 2004. Use of a Beowulf cluster for estimation of risk using SWAT. Agronomy Journal 96 (5), 1495–1497.

Wu, Y., Liu, S., 2012. Automating calibration, sensitivity and uncertainty analysis of complex models using the R package Flexible Modeling Environment (FME): SWAT as an example. Environmental Modelling & Software 31, 99–109.

Yang, J., Abbaspour, K.C., Reichert, P., 2005. Interfacing SWAT with systems analysis tools: a generic platform. In: Srinivasan, R., et al. (Eds.), 2005 3rd International SWAT Conference Proceedings. Swiss Fed. Inst. For Aquat. Sci.. and Technol., Duebendorf, Switzerland, pp. 169–178.

Yang, J., Reichert, P., Abbaspour, K.C., 2007. Bayesian uncertainty analysis in distributed hydrologic modeling: a case study in the Thur River Basin (Switzerland). Water Resources Research 43, W10401.

Yang, J., Reichert, P., Abbabspour, K.C., Xia, J., Yang, H., 2008. Comparing uncertainty analysis techniques for a SWAT application to the Chaohe Basin in China. Journal of Hydrology 358, 1–23.

Zhang, X., Izaurralde, R.C., Zong, Z., Zhao, K., Thomson, A.M., 2012. Evaluating the efficiency of a multi-core aware multi-objective optimization tool for calibrating the SWAT model. Transactions of the ASABE 55 (5), 1723–1731.