MethodsX 5 (2018) 184–203

Contents lists available at ScienceDirect

MethodsX

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

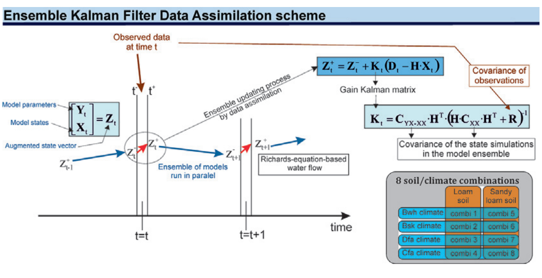
Method Article

MATLAB algorithm to implement soil water data assimilation with the Ensemble Kalman Filter using HYDRUS

Javier Valdes-Abellan[a,*], Yakov Pachepsky[b], Gonzalo Martinez[c]

[a] Department of Civil Engineering, University of Alicante, Alicante, Spain
[b] USDA-ARS, Environmental Microbial and Food Safety Lab., Beltsville, MD, USA
[c] Department of Applied Physics, University of Córdoba, Córdoba, Spain

G R A P H I C A L   A B S T R A C T



A B S T R A C T

Data assimilation is becoming a promising technique in hydrologic modelling to update not only model states but also to infer model parameters, specifically to infer soil hydraulic properties in Richard-equation-based soil water models. The Ensemble Kalman Filter method is one of the most widely employed method among the different data assimilation alternatives. In this study the complete Matlab© code used to study soil data assimilation efficiency under different soil and climatic conditions is shown. The code shows the method how data assimilation through EnKF was implemented. Richards equation was solved by the used of Hydrus-1D software which was run from Matlab.

- MATLAB routines are released to be used/modified without restrictions for other researchers

* Corresponding author.
E-mail address: javier.valdes@ua.es (J. Valdes-Abellan).

https://doi.org/10.1016/j.mex.2018.02.008

• Data assimilation Ensemble Kalman Filter method code.
• Soil water Richard equation flow solved by Hydrus-1D.

ARTICLE INFO
Method name: Climate/soil EnKF efficiency
Keywords: Hydrus, EnKF, Soil water flux modelling
Article history: Received 22 December 2017; Accepted 26 February 2018; Available online xxx

Specifications Table

Subject area	*Select one of the following subject areas:* • *Agricultural and Biological Sciences* • *Computer Science* • *Engineering* • *Environmental Science*
More specific subject area	*Data assimilation by Ensemble Kalman Filter applied to soil water flux modelling to infer soil hydraulic properties.*
Method name	*Climate/soil EnKF efficiency.*
Name and reference of original method	
Resource availability	

Method details

Data assimilation, DA, methods improve the model performance by integrating observed data (i.e., system states) into the modelling process in order to correct the model predictions and or model parameters [1,2]. Among the different DA alternatives, the Ensemble Kalman Filter (EnKF) is one of the most widely used DA methods [6,8]. Shortly, an ensemble of models is randomly generated, then propagated in time to the to the next update event. For each update event, a state error covariance matrix is calculated from the state values simulated by the different ensemble members before the update (a priori). A covariance matrix of observations is also obtained at the same time. Both covariance matrices are used to obtain a new set of model states and model parameters.

In the present contribution, we share the Matlab code used in Valdes-Abellan et al. [3] to apply the EnKF data assimilation method to soil water flow modelling. The code was employed to infer soil model parameters by updating both states and parameter according to the approach showed in Chen and Zhang [7].

The code was created considering a 1-layer soil profile; nevertheless, is straightforward modifying and adapting the code to more complex profiles. Additionally, it is prepared to consider different climates and soil types. This feature can also easily modify to be adapted to the new user aims.

All required subroutines and other files required to run the program are included in the present study as Supplementary materials.

Procedure

This lines are not required to run the model but to clearly exposed the variables used in the model

```
%=====================================================================
====
%% DUAL ENSEMBLE KF CALIBRATION
%=====================================================================
====
% It calibrates all five parameters of the layer
% Made for 4 climates and 2 soil types.

%====Variable
definition===================================================
%    ntobs: number of observation times
%    ndepth: number of reference depths
%    nsensor: number of sensors at a given depth
%    nmat:  number of materials
%    nDAtimes: number of data assimilation times
%    nnode: number of nodes in  soil water simulations with HYDRUS
%    xref: reference depths, cm; they include measurement depths and
%          additional interpolation depths
%    ipick: shows if the reference depth has measurements: 1 - yes, 0
no
%    nref: computational node for reference depths, 1:ndepth
%    wobs: array to store observed water contents, cm^3 cm^-3
%    bias: bias corrected deviation of the observed soil water content
%       for a specific sensor from the average at a given depth, cm^3
cm^-3
%       at a given time
%   ProfileFile: name of the profile file template
%   AtmosphFile: name of the atmospher file template
%   Selector: name of the selector file template

%   tinitial: initial time after the warm-up period
%   inc: period of time between two updates
%   ult: last updating time:
%   prim: first updating time after tinitial
%   NDATIMES: number of DA times
%   DATIMES: times when DA takes place

%   NPTF: number of pdtf

%   winit1: water initial content and all reference depths
%   winit= replication of winit1 for each of PDTF ,
winit(NPTF,ndepths)

%   soil hydralic properties for each material and pdtf.
%   thr(iptf,nmat); ths(iptf,nmat); alpha(iptf,nmat);
%   vgn(iptf,nmat); aks(iptf,nmat); al(iptf,nmat);

%   imodtype= 0 if van Genuchten, 2 if Brooks and Corey
%   NNODES: number of computational nodes in the profile
%   dist=constant distance between two computational nodes

%   x: vector that collects depths in cm for each node
%   h: vector that collects soil pressure head for each node
%   mat: vector that collects soil material for each node

%   aver: average observed water content at a given depth, cm^3 cm^-3
%   cm: covariance matrix of water content observations, cm^6 cm^-6
%   cm: correlation matrix of water content observations, []
%   nob: number of available observations at a given depth (<= number
of sensors)

%   R= covariance matrix covariance matrix of experimental data
%   C= covariance matrix of simulations
%   D= random data matrix, for each ensemble unit
%   H= observation matrix, one/zero according to observed data
%   FI= augmented param-state vector, before updating
%   Cyx= covariance parameters-states
%   K= Kalman update matrix
%   FIP= augmented param-state vector, after updating
%   XP= state vector (in present study, states are only vwc), after
updating


%=====FOLDER STRUCTURE
INSTRUCTIONS===================================================
% The current folder has to contain a folder called "Test_Hydrus_GUI".
This
% folder will include all required HYDRUS templates and all in/out
HYDRUS files.
```

First lines clean all previous results and identify the folder of the input information.

```
%======================================================================
====
clear; clc; fclose all;
conterror=0;
%======================================================================
===
% Initial statements
cDataPath=[pwd '\Test_Hydrus_GUI'];
cFileName=[cDataPath '\Input_dual.dat'];
```

Following lines read the 'InputSHP.dat' file. This file contains the soil hydraulic information for the correct values, the initial values for the searching process, boundaries of existence domain for soil properties, etc.

```
%======================================================================
====
% Reading the inputSHP file.
cFileName=[cDataPath '\InputSHP.dat'];
f20=fopen(cFileName,'r');
nsoiltype=3;
for i=1:5; fgetl(f20);end
shpcorrect=zeros(nsoiltype,6); %pre
for insoil=1:nsoiltype
    shpcorrect(insoil,:)=fscanf(f20,'%f',6); fgetl(f20);
end
fgetl(f20);fgetl(f20);
shpini=zeros(nsoiltype,6); %pre
for inmat=1:nsoiltype; stextall{inmat}=fgetl(f20); end %capture soil
texture
for inmat=1:nsoiltype;
    shpini(inmat,:)=fscanf(f20,'%f',6); fgetl(f20); %capture shp
end
fgetl(f20);
shpborder(1,1:6)=fscanf(f20,'%f%f%f%f%f%f\n',6);
shpborder(2,1:6)=fscanf(f20,'%f%f%f%f%f%f\n',6);
f10=fopen('LEVEL_01.DIR','w');
fprintf(f10,'%s',cDataPath);
fclose(f10); %closing level_01.dir
fclose(f20); %closing inputSHP
clear f10 f20;
```

Next, time information is introduced. *tinitial* informed when the warm-up period finished and when the updating process begins. *inc* collects the time interval in days between different updatings. Time between *tinitial* and the first updating is collected in *prim* variable. Finally, *ult* collects the last day.

```
%===================================================================
% Data assimilation time setup
tinitial=365; %-->-->-->-->--> Initial time after the warm-up period
inc=7;        %-->-->-->-->-->--> MODIFY IF REQUIRED
ult=1460;     %-->-->-->-->-->--> MODIFY IF REQUIRED
prim=7;       %-->-->-->-->-->--> MODIFY IF REQUIRED
DATIMES=[tinitial tinitial+prim:inc:ult];
NDATIMES=length(DATIMES)-1;
```

In the next step, geometrical information: Number of nodes, location of observation points, and others can be modified here.

```
%====================================================================
% x. Geometrical information
NNODES=111;
dist=1; %distance between nodes in cms
xref=[-15;-55;-75]; m=length(xref);
x(1:NNODES,1)=0:-dist:-110;
nmat=1;
mat(1:NNODES,1)=1; %only 1 material.
nref(m)=0;         %preallocation
for j=1:m; nref(j)=find(x==xref(j)); end
ipick(NNODES,1)=0; ipick(nref,1)=1;
```

As abovementioned, the code is prepared to run different climates and soil types. Here the climate-soil loop starts

There are four different climate alternatives. Users may create new climate files following the same structure.

```
%% Beginning of loop clim-soil
for iclima=4       %========================== LOOP OF CLIMATES

%======================================================================
====


    % 2. Selecting the correct Atmospher file
    switch iclima
        case 1; cFileNameAtm=[cDataPath '\ATMOSPH4YR_1_AZ.IN'];
        case 2; cFileNameAtm=[cDataPath '\ATMOSPH4YR_2_ID.IN'];
        case 3; cFileNameAtm=[cDataPath '\ATMOSPH4YR_3_IN.IN'];
        case 4; cFileNameAtm=[cDataPath '\ATMOSPH4YR_4_TX.IN'];
    end
    for isoil=1:2    %========================= LOOP OF SOIL TYPES


%======================================================================
```

In the following lines, the code reads the observation data according to the selected climate and soil type. In Valdes-Abellan et al. [3] is exposed a method to create synthetic data.

```
        % 1. Read observations
        file=['wsynthetic_C' num2str(iclima) '_S' num2str(isoil) ];
        load(file);
        wobs=wsin; clear wsin
        [ntobs,nsensor,~]=size(wobs);       %m is number of
obsrevation depths
        tobs=1:ntobs;

        %========================================
        % Selection of pdtf (ALWAYS ROSSETA IN THIS CODE)
        shpaver=shpini(isoil,:);
```

The ensemble of models is created in the following lines. Different alternatives can be chosen by changing the *generation* variable value. The first option uses the covariance matrix show in Faulkhner et al. [4]; the second uses a diagonal covariance matrix (i.e., only standard deviation are considered but not covariance between different parameters).

Additionally, users can choose the number of units in *nunit* variable. The code let the user to decide what soil parameters are going to be upated. *elecshp* is a logical variable: 1 means that the parameter will be considered during the updating process, and 0 means the opposite. *elecshp* length is five, according to θr, θs, n, α and Ks.

```
======================================================================
==
        % Generation of the ensemble.
        generation=1;   %-->-->-->-->-->-->--> MODIFY IF REQUIRED
        switch generation
            case 1 % Randomly with complete covariance matrix from
Faulkhner
                nunit=10;  %number of units in the ensemble
                % -->-->-->IMP: election of the shp considered to make
the ensemble
                % order--> Qr, Qs, alpha, n, Ks
                randshp=   [1 1 1 1 1];
                stext{nmat}=stextall{isoil};
                shp=gen_ensemble(nunit,nmat,shpaver,stext, randshp);
            case 2  % Randomly from a normal distribution with a user-
defined dispersion. Covariance=0.
                nunit=10;  %number of units in the ensemble
                % -->-->-->IMP: election of the shp considered to make
the ensemble
                % order--> Qr, Qs, alpha, n, Ks
                randshp=   [1 1 1 1 1];
                stan=[50 50 50 50 50];  %-->-->-->Standard deviation,
values in percentage respect to the average
                shp=gen_ensembleV2(nunit,nmat,shpaver,stan, randshp);
        end
        % Election of the shp to be updated
        elecshp=   [1 1 1 1 1];    % order--> Qr, Qs, alpha, n, Ks

        %
======================================================================
==
        % Variables collecting results
        rsquare=zeros(1,NDATIMES);      %preallocation
        rsquarefut=zeros(1,NDATIMES);   %preallocation
        rmse=zeros(1,NDATIMES);         %preallocation
        rmsefut=zeros(1,NDATIMES);      %preallocation
        nse=zeros(1,NDATIMES);          %preallocation
        nsefut=zeros(1,NDATIMES);       %preallocation

        %
==============================================================
```

The previous lines create the variables where results will be saved. At this time they are equal to zero.

In the following the updating process for a specific soil-climate case begins from the initial time, after the warm-up period, to the end of the updating process

```
% Starting the loop of DA
        %
====================================================================
        wsimu(DATIMES(NDATIMES+1)-DATIMES(1),1+nunit,m)=0;
%preallocation of average wc simulated
        Thetas=zeros(NNODES,nunit);
%preallocation, water content in the simulation time
        shpevol=zeros(3+NDATIMES,1+sum(elecshp)); cont=1;
        shpstd=zeros(3+NDATIMES,1+sum(elecshp));
        for inmat=1:nmat
            for j=1:5  %number of soil hydraulic properties
                if elecshp(inmat,j)==1
                    shpevol(1,1+cont)=inmat; shpstd(1,1+cont)=inmat;
%logging material
                    shpevol(2,1+cont)=j;
shpstd(2,1+cont)=j;%logging propertie
                    if j<3;
                        shpevol(3,1+cont)=mean(shp(inmat,j,:));
                        shpstd(3,1+cont)=std(shp(inmat,j,:));
                    else
                       temp1=mean(log10(shp(inmat,j,:)));
                       temp2=std(log10(shp(inmat,j,:)));
                       shpevol(3,1+cont)=10^temp1;
                       shpstd(3,1+cont)=10^(temp1+temp2)-10^temp1;
                   end
                   cont=cont+1;
               end
           end
       end
       shpevol(3,1)=DATIMES(1);
       shpstd(3,1)=DATIMES(1);
```

The Hydrus software is used to solve the Richards-equation-based soil water flux. It requires three input files: ATMOSPH.IN, PROFILE.DAT and SELECTOR.IN. The following code lines are devoted to create those files for each unit of the ensemble for the period ranging from the previous to the next updating time.

The PROFILE.DAT file requires the definition of the initial condition for the complete profile. Criterion to get this is to translate volumetric water content into soil pressure head according to the specific soil hydraulic properties in the observation depths (i.e., those depths were there were data) and to interpolate linearly soil pressure head between observation depths. This is made in the function called *h = W_TO_H_Bv3*. Other options were considered but finally were discarded.

```
        % ==============================================================
         %% Loop DA times
         for idatimes=1:NDATIMES
             delete([cDataPath '\*.out'],[cDataPath
'\ATMOSPH.IN'],[cDataPath '\PROFILE.DAT'],[cDataPath '\SELECTOR.IN']);
%  [cDataPath '\FIT.IN'],[cDataPath '\HYDRUS1D.DAT'])
             nerrors=zeros(NDATIMES,nunit);
             t1=DATIMES(idatimes);
             t2=DATIMES(idatimes+1);
             % ====================================================
             % Generation bias data
             bias(m,nsensor)=0;
             for mm=1:m
                 l1=find(tobs==DATIMES(1)); %find the ini line, the
first time for all loops
                 l2=find(tobs==t2); %find the end line
                 wobused=wobs(l1:l2,:,mm);
                 bias(mm,:)=mean(wobused)-mean(mean(wobused));
             end

             %
==========================================================
             % Generation ATMOSPHER file, from t1 to t2.
             % Same file for all units of the ensemble
             %==============================
             period=t2-t1;
             temp=importdata(cFileNameAtm, '\t', 9); temp=temp.data;
             f25= fopen(cFileNameAtm, 'r');
             cFileName=[cDataPath '\ATMOSPH.IN'];
             f26= fopen(cFileName, 'w');
             while f26==-1
                 pause(1);
                 conterror=conterror+1;
                 f26= fopen(cFileName, 'w');
             end
             for k=1:3; line=fgetl(f25); fprintf(f26,line);
fprintf(f26, '\n'); end %copy-paste 3
             fgetl(f25); fprintf(f26,'%7.0f',period);fprintf(f26,
'\n');
             for k=1:5; line=fgetl(f25); fprintf(f26,line);
fprintf(f26, '\n'); end %copy-paste 3
             %
             l1=find(temp(:,1)==t1+1); %find the ini line,
             l2=find(temp(:,1)==t2); %find the end line
             temp=temp(l1:l2,:);  %election of the right lines
according to t1,t2
             temp(:,9:end)=[];  %to remove extra columns from Gonzalo

fprintf(f26,'%11.0f%12.3f%12.3f%12.0f%12.0f%12.0f%12.0f%12.0f\n',temp'
);
             fprintf(f26,'end');
             fclose(f25);
             fclose(f26);
             clear f25 f26;
             for inunit=1:nunit
                 %==============================
                 % Build the 'SELECTOR.IN' file
                 %==============================
                 cFileName=[cDataPath '\SELECTOR_Template_direct.IN'];
                 f21= fopen(cFileName, 'r');
                 cFileName=[cDataPath '\Selector.in'];
                 f22= fopen(cFileName, 'w');
                 for k =1:26; line=fgetl(f21);fprintf(f22,line);
fprintf(f22, '\n'); end %copy-paste

fprintf(f22,'%7.3f%8.4f%8.4f%8.4f%11.2f%8.1f\n',shp(:,:,inunit)');
%paste shp
                 for inmat=1:nmat; fgetl(f21);end
                 for k=1:4; line=fgetl(f21);fprintf(f22,line);
fprintf(f22, '\n'); end  %copy-paste
                 fgetl(f21);
                 fprintf(f22, '%11.4f%15.4f', [t1 t2]);fprintf(f22,
'\n');
                 for k=1:3; line=fgetl(f21);fprintf(f22,line);
fprintf(f22, '\n');end;%copy-paste
                 fgetl(f21);
                 fprintf(f22, '%11.4f', t2);fprintf(f22, '\n');
                 line=fgetl(f21);
                 fprintf(f22,line);
                 fprintf(f22, '\n');
                 fclose(f21);
                 fclose(f22);
                 clear f21 f22;
                 % ====================================================
                 % Generation PROFILE.DAT file
                 % A particular subroutine for idatimes=1 and other for
the rest
                 %
```

```
=====================================================
                thr1=shp(:,1,inunit);         ths1=shp(:,2,inunit);
                alpha1=shp(:,3,inunit);       vgn1=shp(:,4,inunit);
                imodel=0;
               if idatimes==1
                   %obtains the h for all nodes from swc at observed
depths
                   temp=wobs(find(tobs==t1),:,:);
                   winit1(1:m,1)=mean(temp);

h=W_TO_H_Av2(NNODES,x,mat,xref,winit1,thr1,ths1,alpha1,vgn1,imodel);
                   hini=h;
               else
                   flag=2;
                   if flag==0         %version that translate wtoh in
all profile.

h=W_TO_H_Bv2(XP,inunit,mat,thr1,ths1,alpha1,vgn1,imodel);
                   else                %version that translates wtoh
just in nobs depths.
                       Thetasav=Thetas(:,inunit);
                       winit1=XP(nref,inunit);  %vwc at observation
depths

h=W_TO_H_Bv3(NNODES,x,mat,xref,winit1,thr1,ths1,alpha1,vgn1,Thetasav,i
model);
                   end
               end
               ja=find(h<-1e6);
               if ~isempty (ja);
                   h(ja)=-1e6; %filtering sph under -1e6 cm
               end
               clear ja
               %
               cFileName=[cDataPath '\PROFILE_Template.DAT'];
               f24= fopen(cFileName, 'r');
               cFileName=[cDataPath '\Profile.dat'];
               f32= fopen(cFileName, 'w');
               for k=1:4; line=fgetl(f24); fprintf(f32,line);
fprintf(f32, '\n'); end %copy the first 4 lines
               line=fgetl(f24);        %5th line
               line(1:4)=[' ' num2str(NNODES)];            %tendré
problemas si NNODES no tiene tres digitos
               fprintf(f32,line); fprintf(f32, '\n');
               profileend=zeros(NNODES, 9); %preallocation
               profileend(:,1)=1:NNODES;   %nodes
               profileend(:,2)=x;          %depths
               profileend(:,3)=h;          %initial sph
               profileend(:,4)=mat;        %material
               profileend(:,5)=mat;        %layer
               profileend(:,7:9)=1;        %scaling factor

fprintf(f32,'%5.0d%15.2f%15.3f%5.0d%5.0d%15.3f%15.3f%15.3f%15.3f\n',pr
ofileend');
               fprintf(f32, '%d',m); fprintf(f32,'\n');
               fprintf(f32, '%5.0f%5.0f%5.0f\n', nref);
               fclose(f32); fclose(f24);
               clear f32 f24;
```

With all required files, HYDRUS is finalled called. In the present code, simulations requiring more than 6 s to finished computations were considered uncorrect and discarded. To interrupt a HYDRUS running, a system function called taskkill was used.

```
                %
================================================================
                %% Running Hydrus and reading results
                %
================================================================
                % Running HYDRUS on each assimilation time
                tic;
                % The following path has to be modified according to
each PC
                !G:\JVA\05 INVESTIGACION\10.14 Pap. Data
assimilation\SSDA_MAT\H1D_CALC.EXE &

%===============================================================
                % identification of non-convergence runs

%===============================================================
                je=toc;
                [~,temp]=system('tasklist /FI "imagename eq
H1D_CALC*"');
                while je<6 && strcmp(temp(1:4),'INFO')==0
                    pause (1)
                    je=toc;
                    [~,temp]=system('tasklist /FI "imagename eq
H1D_CALC*"');
                end
                if je>6
                    system('taskkill /f /im H1D_CALC.EXE')
                end
                keyInject('Administrador:
C:\Windows\SYSTEM32\cmd.exe','exit\r','MATLAB R2015a')
```

After computations finish, results from this run are read. This run implies a specific unit of the ensemble, a specific climate, a specific soil and a specific time.

```
                %
=========================================================
                % Reading results
                %
======================================================
                cFileName=[cDataPath '\Error.msg'];
                f23= fopen(cFileName, 'r');
                if f23~=-1;
                    fclose(f23); nerrors(idatimes,inunit)=1; clear
f23;
                    delete ([cDataPath '\Error.msg']);
                else
                    cFileName=[cDataPath '\OBS_NODE.OUT'];
                    a=importdata(cFileName, ' ',11); a=a.data;
                    if max(a(:,1))<t2
                        nerrors(idatimes,inunit)=1;
                    else
                        %==========================================
                        % Reading simulated h and w from HYDRUS output
files
                        cFileName=[cDataPath '\Nod_Inf.out'];
                        f23= fopen(cFileName, 'r');
                        if      length(num2str(t2))==4;  comp=[' Time:
' num2str(t2) '.0000'];
                        elseif  length(num2str(t2))==3;  comp=[' Time:
' num2str(t2) '.0000'];
                        else                             comp=[' Time:
' num2str(t2) '.0000'];
                        end
                        line=fgetl(f23);          cont=1;
                        while strcmp(comp,line)==0; line=fgetl(f23);
cont=cont+1; end
                        a=importdata(cFileName, ' ',cont+5); a=a.data;
                        xhydrus=a(:,2); % nodal coordinates used in Hy
4.14.
                        Thetas(:,inunit)=a(:,4);% ThNew: water
simulated with HYDRUS on the assimilation time
                        fclose(f23); clear f23;

                        % ======================================
                        % save simulated data
                        cFileName=[cDataPath '\OBS_NODE.OUT'];
                        a=importdata(cFileName, ' ',11); a=a.data;
                        clear temp; temp(1,2*m)=0;
                        for mm=1:m
                            temp(1,mm*2-1)=3*mm-1;
temp(1,mm*2)=3*mm+1;      %columns to be deleted
                        end
                        a(:,temp)=[];%election of just wvc

                        %logging simulated values except t2
                        for t=1:t2-t1-1
                            l1=find(a(:,1)>t1-1+t,1,'first');
                            l2=find(a(:,1)==(t1+t));
                            wsimu(t1-
DATIMES(1)+t,1+inunit,1:m)=mean(a(l1:l2,2:m+1));
                            wsimu(t1-DATIMES(1)+t,1,1:m)=t1+t;
                        end
                    end
                end
                disp (['End of simulation ' num2str(inunit)])
            end% inunit, end of units of the ensemble; before DA
update

            if sum(nerrors(idatimes,:))>0
                clear temp;
                temp=Thetas;
                temp(:,find(nerrors(idatimes,:)))=[]; %elimino errores
                for i=1:nunit
                    if nerrors(idatimes,i)==1;
Thetas(:,i)=mean(temp,2); end
                end
            end
```

Once all units of the ensemble have been run and their results collected, the updating process can be undertaken. First, all required matrices are obtained (*R*, *X*, *C*, *D*, *H*, *Cyx*) and second an updated augmented vector of states and parameters (*FIP*) is obtained

```
%=====================================================================
==
            %% DA core starts here
            %============================================
            % Observation data on the assimilation time
            temp=wobs(find(tobs==t2),:,:); %find the line
            for inmd=1:m; v(inmd,:)=temp(1,:,inmd); end
            aver=mean(v,2);
            % Correction for bias, JVA: removed the component that
substract the
            % average, not required because v is just used to computed
the cov
            for mm=1:m; v(mm,:)=v(mm,:)- bias(mm,:); end
            %=========== R(m,m), covariance matrix of experimental
data.
            R=cov(v');

            %=========== C(n,n), covariance matrix of simulations
            nunitOK=nunit-sum(nerrors(idatimes,:));
            X=Thetas; %XP(n,N),  predictions prior updating
            X(:,find(nerrors(idatimes,:)))=[];   %remove simulations
results with errors
            C=cov(X');

            %========== D(m,n), random data matrix, for each ensemble
unit
            temp=chol(R);
            D=repmat(aver',nunitOK,1)+randn(nunitOK,m)*temp;
            D=D'; % now D(m,n)

            %========== H(m,n), observation matrix, one/zero according
to observed data
            H(m,NNODES)=0; cont=1;
            for i=1:NNODES
                if ipick(i)>0; H(cont,i)=1; cont=cont+1; end
            end

            %========== Cyx(t+n,n) covariance parameters-states
            clear temp
            t=sum(sum(elecshp));
            temp=zeros(t,nunit); %preallocationg
            cont=1;
            for inmat=1:nmat
                for j=1:5 %number of soil hydraulic properties, l not
considered
                    if elecshp(inmat,j)==1
                        temp(cont,:)=shp(inmat,j,:);
                        cont=cont+1;
                    end
                end
            end
            temp(:,find(nerrors(idatimes,:)==1))=[]; %elimino comb que
dieron error
            FI=[temp;X];  %augmented param-state vector, before
updating
            Cyx=cov(FI');
            Cyx(:,1:t)=[];

            %=========== K(n,m), Kalman update matrix
            K=Cyx*H'/(H*C*H'+R);

            %=========== FIP(n,N),  corrected predictions after
updating
            FIP=FI+K*(D-H*X);
```

When the updated vector is obtained, averages values from the successful runs are assigned to the runs which reported an error and therefore they had no results.

```
%=========== filling the error runnings
FIP2=zeros(NNODES+t,nunit); cont=1;
for inunit=1:nunit
    if nerrors(idatimes,inunit)==1
        FIP2(:,inunit)=mean(FIP,2);
    else
        FIP2(:,inunit)=FIP(:,cont);
        cont=cont+1;
    end

end
FIP=FIP2; clear FIP2

XP=FIP;   XP(1:t,:)=[]; %updated vector of just states
%% DA finishs here
%
============================================================
```

If updated soil parameters falls out of logical boundaries during the updating process (e.g., residual water content below zero), then they are moved to the closest border of a logical domain. Border values are included in the 'InputSHP.dat' file and have been read in the first stages.

```
%  ========================================
%  Filtering shp out of borders
for i=1:5
    temp=find(FIP(i,:)<shpborder(1,i));  %minimum border
    if ~isempty(temp);
        FIP(i,temp)=shpborder(1,i);
    end
    temp=find(FIP(i,:)>shpborder(2,i));    %maximum border
    if ~isempty(temp);
        FIP(i,temp)=shpborder(2,i);
    end
end
```

After the previous filter, results from soil hydraulic parameters updating are saved

```
            % ==========================================
            %  Saving upated shp
            cont=1;
            for inmat=1:nmat
                for j=1:5 %number of soil hydraulic properties, 1 not
considered
                    if elecshp(inmat,j)==1
                        shp(inmat,j,:)=FIP(cont,:);  %actualizo las
shp
                        shpevol(3+idatimes,1)=t2;
                        shpstd(3+idatimes,1)=t2;
                        if j<3;

shpevol(3+idatimes,1+cont)=mean(shp(inmat,j,:));

shpstd(3+idatimes,1+cont)=std(shp(inmat,j,:));
                        else
                            temp1=mean(log10(shp(inmat,j,:)));
                            temp2=std(log10(shp(inmat,j,:)));
                            shpevol(3+idatimes,1+cont)=10^(temp1);

shpstd(3+idatimes,1+cont)=10^(temp1+temp2)-10^temp1;
                        end
                        cont=cont+1;
                    end
                end
            end
```

Soil states (volumetric water content and soil pressure head) are also corrected to avoid illogical values, similarly to the process undertaken with soil hydraulic properties.

```
            % ==========================================
            % Filtering updated states
            clear temp
            for inunit=1:nunit
                if ~isempty(find(XP(:,inunit)<0,1))
                    XP(find(XP(:,inunit)<0),inunit)=0;  %Not acepteced
below 0
                end
                for i=1:NNODES
                    if XP(i,inunit)>shp(mat(i),2,inunit) %Compare with
Qs
                        XP(i,inunit)=0.999*shp(mat(i),2,inunit);
                    end
                end
            end


%=====================================================================
=
            % Log of simulated wvc in t2 after updating
            for mm=1:m
                wsimu(t2-DATIMES(1),1,mm)=t2; %log wvc in DA time
                wsimu(t2-DATIMES(1),2:nunit+1,mm)=XP(nref(mm),:); %log
wvc in DA time
            end
```

After the updating process has finished completely, and results have been saved, a direct run is developed with the last updated set of soil properties to obtain the model performance. The direct model is run from the beginning to the time of the last updating.

As all Hydrus runs, before to run the model, all required files have to be built. After the model has run

```
            %

%=============================================================
            %% Rerunning-PAST PERIOD FOR STATISTICS
            %
=========================================================

            delete([cDataPath '\*.out'],[cDataPath
'\ATMOSPH.IN'],[cDataPath '\PROFILE.DAT'],[cDataPath '\SELECTOR.IN']);
%  [cDataPath '\FIT.IN'],[cDataPath '\HYDRUS1D.DAT'])
            % Generation atmospher file, from tinitial to t2.
            period=t2-tinitial;
            temp=importdata(cFileNameAtm, '\t', 9); temp=temp.data;
            f25= fopen(cFileNameAtm, 'r');
            cFileName=[cDataPath '\ATMOSPH.IN'];
            f26= fopen(cFileName, 'w');
            %
            for k=1:3; line=fgetl(f25); fprintf(f26,line);
fprintf(f26, '\n'); end %copy-paste 3
            fgetl(f25); fprintf(f26,'%7.0f',period);fprintf(f26,
'\n');
            for k=1:5; line=fgetl(f25); fprintf(f26,line);
fprintf(f26, '\n'); end %copy-paste 3
            %
            l1=find(temp(:,1)==tinitial+1); %find the ini line,
            l2=find(temp(:,1)==t2); %find the end line
            temp=temp(l1:l2,:);  %election of the right lines
according to tinitial,t2
            temp(:,9:end)=[];  %to remove extra columns from Gonzalo

fprintf(f26,'%11.0f%12.3f%12.3f%12.0f%12.0f%12.0f%12.0f%12.0f\n',temp'
);
            fprintf(f26,'end');
            fclose(f25);
            fclose(f26);
            clear f25 f26;
            %==============================
            % Build the 'SELECTOR.IN' file
            %==============================
            cFileName=[cDataPath '\SELECTOR_Template_direct.IN'];
            f21= fopen(cFileName, 'r');
            cFileName=[cDataPath '\Selector.in'];
            f22= fopen(cFileName, 'w');
            for k =1:26; line=fgetl(f21);fprintf(f22,line);
fprintf(f22, '\n'); end %copy-paste
            clear temp; temp=[shpevol(3+idatimes,2:6) 0.5];
            fprintf(f22,'%7.3f%8.4f%8.4f%8.4f%11.2f%8.1f\n',temp');
%paste shp after updating
            for inmat=1:nmat; fgetl(f21);end
            for k=1:4; line=fgetl(f21);fprintf(f22,line); fprintf(f22,
'\n'); end  %copy-paste
            fgetl(f21);
            fprintf(f22, '%11.4f%15.4f', [tinitial t2]);fprintf(f22,
'\n');
            for k=1:3; line=fgetl(f21);fprintf(f22,line); fprintf(f22,
'\n');end;%copy-paste
            fgetl(f21);
            fprintf(f22, '%11.4f', t2);fprintf(f22, '\n');
            line=fgetl(f21);
            fprintf(f22,line);
            fprintf(f22, '\n');
            fclose(f21);
            fclose(f22);
            clear f21 f22;
            % ===================================================
            % Generation PROFILE.DAT file
            % ===================================================
              thr1=shpevol(3+idatimes,2);
  ths1=shpevol(3+idatimes,3);
              alpha1=shpevol(3+idatimes,4);
  vgn1=shpevol(3+idatimes,5);
              imodel=0;
              temp=wobs(find(tobs==tinitial),:,:);
              winit1(1:m,1)=mean(temp);
              %active the version of just 3 observations depths

  h=W_TO_H_Av2(NNODES,x,mat,xref,winit1,thr1,ths1,alpha1,vgn1,imodel);
              cFileName=[cDataPath '\PROFILE_Template.DAT'];
              f24= fopen(cFileName, 'r');
              cFileName=[cDataPath '\Profile.dat'];
              f32= fopen(cFileName, 'w');
              for k=1:4; line=fgetl(f24); fprintf(f32,line);
  fprintf(f32, '\n');end %copy the first 5 lines
              line=fgetl(f24);                %5th line
              line(1:4)=[' ' num2str(NNODES)];            %tendré
  problemas si NNODES no tiene tres digitos
              fprintf(f32,line); fprintf(f32, '\n');
              profileend=zeros(NNODES, 9); %preallocation
              profileend(:,1)=1:NNODES;   %nodes
              profileend(:,2)=x;          %depths
              profileend(:,3)=h;          %initial sph
              profileend(:,4)=mat;        %material
              profileend(:,5)=mat;        %layer
              profileend(:,7:9)=1;        %scaling factor

  fprintf(f32,'%5.0d%15.2f%15.3f%5.0d%5.0d%15.3f%15.3f%15.3f%15.3f\n',pr
  ofileend');
              fprintf(f32, '%d',m); fprintf(f32,'\n');
              fprintf(f32, '%5.0f%5.0f%5.0f\n', nref);
              fclose(f32); fclose(f24);
              clear f32 f24;
              %
  =========================================================
              %% Re-running with final shp-PAST PERIOD
              %
```

```
=======================================================
            % Running HYDRUS on each assimilation time
            tic;
            % The following path has to be modified according to each
PC
            !G:\JVA\05 INVESTIGACION\10.14 Pap. Data
assimilation\SSDA_MAT\H1D_CALC.EXE &
            %=======================================================
            % identification of non-convergence runs
            %=======================================================
            je=toc;
            [~,temp]=system('tasklist /FI "imagename eq H1D_CALC*"');
            while je<6 && strcmp(temp(1:4),'INFO')==0
                pause (1)
                je=toc;
                [~,temp]=system('tasklist /FI "imagename eq
H1D_CALC*"');
            end
            if je>6
                system('taskkill /f /im H1D_CALC.EXE')
            end
            keyInject('Administrador:
C:\Windows\SYSTEM32\cmd.exe','exit\r','MATLAB R2015a')
            disp(['-->-->-->--> Rerunning with updated shp, PAST
period statistics DA time = ' num2str(t2)])
            %
=========================================================
            % Reading results
            %
=========================================================
            cFileName=[cDataPath '\Error.msg'];
            f23= fopen(cFileName, 'r');
            if f23~=-1;
                fclose(f23); nerrors(idatimes,inunit)=1; clear f23;
                delete ([cDataPath '\Error.msg']);
                rmse(idatimes)=nan;
                rsquare(idatimes)=nan;
                nse(idatimes)=nan;
            else
                cFileName=[cDataPath '\OBS_NODE.OUT'];
                a=importdata(cFileName, ' ',11); a=a.data;
                if max(a(:,1))<t2
                    rmse(idatimes)=nan;
                    rsquare(idatimes)=nan;
                    nse(idatimes)=nan;
                else
                    %============================================
                    %  Reading simulated data in the re-running turn
                    cFileName=[cDataPath '\OBS_NODE.OUT'];
                    a=importdata(cFileName, ' ',11); a=a.data;
                    clear temp; temp(1,2*m)=0;
                    for mm=1:m
                        temp(1,mm*2-1)=3*mm-1; temp(1,mm*2)=3*mm+1;
%columns to be deleted
                    end
                    a(:,temp)=[];%election of just wvc
                    % ========
                    % Selection observed data in the re-running
                    clear wobused esta cova
                    temp=mean(wobs(tinitial+1:t2,:,:),2);
                    for im=1:m;wobused(:,im)=temp(:,1,im);end
%number of observations
                    esta(:,1)=wobused(:,1);
%col 1: observed
                    esta(:,2)=a(1:t2-tinitial,2);
%col 2: simulated
                    for im=2:m
                        temp=[wobused(:,im) a(1:t2-tinitial,im+1)];
%observed and simulated
                        esta=[esta;temp];
%col1=obs, col2=sim
                    end
                    %============================================
                    %  Obtaining the statistics.
                    rmse(idatimes)=sqrt(sum(sum((a(:,2:m+1)-
wobused).^2))/length(esta));
                    cova=cov(esta);

rsquare(idatimes)=((cova(1,2))^2)/(cova(1,1)*cova(2,2));
                    nse(idatimes)=1-sum((esta(:,1)-
esta(:,2)).^2)/sum((esta(:,1)-mean(esta(:,1))).^2);
                end
            end
```

Similarly to the direct run for the past period, another direct run is accomplished with the last updated value of soil properties to obtain the model performance in case of future predictions. As abovementioned, first all required files are made, then Hydrus software is call. Finally the root mean square error, RMSE, the coefficient of determination, R^2, and the Nash-Sutcliffe efficiency index, NSE [5], statistics are computed.

```
%==============================================================
            %% Rerunning-FUTURE PERIOD FOR STATISTICS
            %
==========================================================
            fclose all;
            delete([cDataPath '\*.out'],[cDataPath
'\ATMOSPH.IN'],[cDataPath '\PROFILE.DAT'],[cDataPath '\SELECTOR.IN']);
%  [cDataPath '\FIT.IN'],[cDataPath '\HYDRUS1D.DAT'])
            % Generation atmospher file, from t2 to ult.
            period=ult-t2;
            if period==0
                rmsefut(idatimes)=nan;
                rsquarefut(idatimes)=nan;
                nsefut(idatimes)=nan;
            else
                temp=importdata(cFileNameAtm, '\t', 9);
temp=temp.data;
                f25= fopen(cFileNameAtm, 'r');
                cFileName=[cDataPath '\ATMOSPH.IN'];
                f26= fopen(cFileName, 'w');
                %
                for k=1:3; line=fgetl(f25); fprintf(f26,line);
fprintf(f26, '\n'); end %copy-paste 3
                fgetl(f25); fprintf(f26,'%7.0f',period);fprintf(f26,
'\n');
                for k=1:5; line=fgetl(f25); fprintf(f26,line);
fprintf(f26, '\n'); end %copy-paste 3
                %
                l1=find(temp(:,1)==t2+1); %find the ini line,
                l2=find(temp(:,1)==ult); %find the end line
                temp=temp(l1:l2,:);  %election of the right lines
according to tinitial,t2
                temp(:,9:end)=[];  %to remove extra columns

fprintf(f26,'%11.0f%12.3f%12.3f%12.0f%12.0f%12.0f%12.0f%12.0f\n',temp'
);
                fprintf(f26,'end');
                fclose(f25);
                fclose(f26);
                clear f25 f26;
                %==============================
                % Build the 'selector.in' file
                cFileName=[cDataPath '\SELECTOR_Template_direct.IN'];
                f21= fopen(cFileName, 'r');
                cFileName=[cDataPath '\Selector.in'];
                f22= fopen(cFileName, 'w');
                for k =1:26; line=fgetl(f21);fprintf(f22,line);
fprintf(f22, '\n'); end %copy-paste
                clear temp; temp=[shpevol(3+idatimes,2:6) 0.5];
fprintf(f22,'%7.3f%8.4f%8.4f%8.4f%11.2f%8.1f\n',temp'); %paste shp
after updating
                for inmat=1:nmat; fgetl(f21);end
                for k=1:4; line=fgetl(f21);fprintf(f22,line);
fprintf(f22, '\n'); end  %copy-paste
                fgetl(f21);
                fprintf(f22, '%11.4f%15.4f', [t2 ult]);fprintf(f22,
'\n');
                for k=1:3; line=fgetl(f21);fprintf(f22,line);
fprintf(f22, '\n');end;%copy-paste
                fgetl(f21);
                fprintf(f22, '%11.4f', ult);fprintf(f22, '\n');
                line=fgetl(f21);
                fprintf(f22,line);
                fprintf(f22, '\n');
                fclose(f21);
                fclose(f22);
                clear f21 f22;
                % ====================================================
                % Generation profile.dat file. En el re-running
                thr1=shpevol(3+idatimes,2);
ths1=shpevol(3+idatimes,3);
                alpha1=shpevol(3+idatimes,4);
vgn1=shpevol(3+idatimes,5);
                imodel=0;
                temp=wobs(find(tobs==t2),:,:);
                winit1(1:m,1)=mean(temp);
                %active the version of just 3 observations depths

h=W_TO_H_Av2(NNODES,x,mat,xref,winit1,thr1,ths1,alpha1,vgn1,imodel);
                cFileName=[cDataPath '\PROFILE_Template.DAT'];
                f24= fopen(cFileName, 'r');
                cFileName=[cDataPath '\Profile.dat'];
                f32= fopen(cFileName, 'w');
                for k=1:4; line=fgetl(f24); fprintf(f32,line);
fprintf(f32, '\n');end %copy the first 5 lines
                line=fgetl(f24);                %5th line
                line(1:4)=[' ' num2str(NNODES)];
                fprintf(f32,line); fprintf(f32, '\n');
                profileend=zeros(NNODES, 9); %preallocation
                profileend(:,1)=1:NNODES;   %nodes
                profileend(:,2)=x;          %depths
                profileend(:,3)=h;          %initial sph
                profileend(:,4)=mat;        %material
                profileend(:,5)=mat;        %layer
                profileend(:,7:9)=1;        %scaling factor

fprintf(f32,'%5.0d%15.2f%15.3f%5.0d%5.0d%15.3f%15.3f%15.3f%15.3f\n',pr
ofileend');
                fprintf(f32, '%d',m); fprintf(f32,'\n');
                fprintf(f32, '%5.0f%5.0f\n', nref);
                fclose(f32); fclose(f24); clear f23 f24;
                %
```

```
=========================================================
                %% Re-running with final shp-FUTURE PERIOD
                %
=========================================================
                % Running HYDRUS on each assimilation time
                tic;

                 % The following path has to be modified according to
 each PC
                 !G:\JVA\05 INVESTIGACION\10.14 Pap. Data
 assimilation\SSDA_MAT\H1D_CALC.EXE &

 %========================================================
                 % identification of non-convergence runs

 %========================================================
                 je=toc;
                 [~,temp]=system('tasklist /FI "imagename eq
 H1D_CALC*"');
                 while je<6 && strcmp(temp(1:4),'INFO')==0
                     pause (1)
                     je=toc;
                     [~,temp]=system('tasklist /FI "imagename eq
 H1D_CALC*"');
                 end
                 if je>6
                     system('taskkill /f /im H1D_CALC.EXE')
                 end
                 keyInject('Administrador:
 C:\Windows\SYSTEM32\cmd.exe','exit\r','MATLAB R2015a')
                 disp(['-->-->-->--> Rerunning with updated shp, FUTURE
 period statistics DA time = ' num2str(t2)])
                 %
 ==================================================================
                 %Reading results
                 cFileName=[cDataPath '\Error.msg'];
                 f23= fopen(cFileName, 'r');
                 if f23~=-1;
                     fclose(f23); nerrors(idatimes,inunit)=1; clear
 f23;
                     delete ([cDataPath '\Error.msg']);
                     rmsefut(idatimes)=nan;
                     rsquarefut(idatimes)=nan;
                     nsefut(idatimes)=nan;
                 else
                     cFileName=[cDataPath '\OBS_NODE.OUT'];
                     a=importdata(cFileName, ' ',11); a=a.data;
                     if max(a(:,1))<ult
                         rmsefut(idatimes)=nan;
                         rsquarefut(idatimes)=nan;
                         nsefut(idatimes)=nan;
                     else
                         %============================================
                         %  Reading simulated data in the re-running
 turn
                         cFileName=[cDataPath '\OBS_NODE.OUT'];
                         a=importdata(cFileName, ' ',11); a=a.data;
                         clear temp; temp(1,2*m)=0;
                         for mm=1:m
                             temp(1,mm*2-1)=3*mm-1;
 temp(1,mm*2)=3*mm+1; %columns to be deleted
                         end
                         a(:,temp)=[];%election of just wvc
                         % ========
                         % Selection observed data in the re-running
                         clear wobused esta cova
                         temp=mean(wobs(t2+1:ult,:,:),2);
                         for im=1:m;wobused(:,im)=temp(:,1,im);end
 %humber of observations
                         esta(:,1)=wobused(:,1);
 %col 1: observed
                         esta(:,2)=a(1:ult-t2,2);
 %col 2: simulated
                         for im=2:m
                             temp=[wobused(:,im) a(1:ult-t2,im+1)];
 %observed adn simulated
                             esta=[esta;temp];
 %col1=obs, col2=sim
                         end
                         %============================================
                         %  Obtaining the statistics.
                         rmsefut(idatimes)=sqrt(sum(sum((a(:,2:m+1)-
 wobused).^2))/length(esta));
                         cova=cov(esta);

 rsquarefut(idatimes)=((cova(1,2))^2)/(cova(1,1)*cova(2,2));
                         nsefut(idatimes)=1-sum((esta(:,1)-
 esta(:,2)).^2)/sum((esta(:,1)-mean(esta(:,1)))).^2);
                     end
                 end
             end
             fclose all;
         end
```

Last stages of the code are devote to save all results.

```
        %
        %==========================================================
        %% SAVING RESULTS
        % =========================================
        %  Saving observation in file
        filename = ['wout_obs.txt'];
        cFileName=[cDataPath '\' filename];
        f23= fopen(cFileName, 'w');
        line='    Day';
        for mm=1:m;  line=[line '        ' num2str(abs(xref(mm)))
'cm']; end
        l1=find(tobs==DATIMES(1));
        l2=find(tobs==DATIMES(NDATIMES+1));
        wobsaver(l2-l1+1,m+1)=0;
        wobsaver(1:l2-l1+1,1)=tobs(l1:l2);
        for mm=1:m
            wobsaver(1:l2-l1+1,mm+1)=mean(wobs(l1:l2,:,mm),2);
        end
        fprintf(f23,line);  fprintf(f23, '\n');
        fprintf(f23, '%8.2f%12.4f%12.4f%12.4f\n', wobsaver');
        fclose (f23); clear f23;
        % =========================================
        %  Saving simulation in file
        filename = ['wout_sim.txt'];
        cFileName=[cDataPath '\' filename];
        f23= fopen(cFileName, 'w');
        line='    Day';
        for mm=1:m;  line=[line '        ' num2str(abs(xref(mm)))
'cm'];  end
        wsimuaver=wobsaver(1,:);  %copy the first line from obs to
simu. It's the initial time
        wsimuaver(DATIMES(NDATIMES+1)-DATIMES(1)+1,m+1)=0;
%preallocation
        wsimuaver(2:DATIMES(NDATIMES+1)-DATIMES(1)+1,1)=wsimu(:,1,1);
%copy times;
        for mm=1:m
            logi=wsimu(:,2:end,mm)>0; % to not consider error
simulations
            wsimuaver(2:DATIMES(NDATIMES+1)-
DATIMES(1)+1,mm+1)=sum(wsimu(:,2:end,mm),2)./sum(logi,2);
        end
        fprintf(f23,line);  fprintf(f23, '\n');
        fprintf(f23, '%8.2f%12.4f%12.4f%12.4f\n', wsimuaver');
        fclose (f23); clear f23;


%======================================================================
====
        % 8. Saving results
        if generation==1
            line=['results\I05_dualv2_C' num2str(iclima) '-S'
num2str(isoil) '-Inc_' num2str(inc) '-Discre_' num2str(dist)];
        else
            line=['results\I05_dualv2_gen2_C' num2str(iclima) '-S'
num2str(isoil)];
        end
        save(line,'shpevol2','shpstd','rsquare','rmse',
'nse','rsquarefut','rmsefut','nsefut')
        fclose all;
    end     % end of soil type
end         % end of climate type
disp('The end!!!!!')
```

Acknowledgements

This study forms part of the CGL2013-48802-C3-3-R project financed by the Spanish Ministry of Science and Innovation, the FPDI-2013-16742 from the Spanish Ministry of Economics, and GRE15-19 financed by the University of Alicante. A post-doctoral research fellowship (CAS 15/00244) funded by the Spanish Ministry of Science and Innovation was awarded to J. Valdes-Abellan for this project.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.mex.2018.02.008.

List of submitted files and routines:

- *gen_ensemble.mandgen_ensemblev2.m: functions used to generate the ensemble of models.*
- *HID_CALC.EXE: executable to run HYDRUS-1D from MATLAB©.*
- *keyinject.m: function to abort HYDRUS-1D when it has convergence problems.*
- *LEVEL_01.DIR: it collects folder location of HYDRUS-1D.*
- *shp_cov_faulkner2003.txt: it collects covariance matrix according to Faulkner* et al. [4]
- *SSDA_DUAL_v2_le_201711_MX.m: main script*
- *W_TO_H_A.m,W_TO_H_Av2.m, W_TO_H_Av2.m and W_TO_H_Av2.m: subroutines to get the initial soil pressure head profile*
- *wsynthetic_Ci_Sj.mat: Matlab file collecting observed soil water content for climate i K and soil j.*

List of folders:

- *results: it collects all results*
- *Test_Hydrus_GUI: it collects all required HYDRUS templates and all in/out HYDRUS files.*

References

[1] G. Evensen, Data Assimilation: The Ensemble Kalman Filter, Springer Berlin Heidelberg, 2009, pp. 307.
[2] D.A. Plaza, R. De Keyser, G.J.M. De Lannoy, L. Giustarini, P. Matgen, V.R.N. Pauwels, The importance of parameter resampling for soil moisture data assimilation into hydrologic models using the particle filter, Hydrol. Earth Syst. Sci. 16 (2012) 375–390.
[3] J. Valdes-Abellan, Y. Pachepsky, G. Martinez, Obtaining soil hydraulic parameters from soil water content data assimilation under different climatic/soil conditions, Catena 163 (2018) 311–320.
[4] B.R. Faulkner, W.G. Lyon, F.A. Khan, S. Chattopadhyay, Modeling leaching of viruses by the monte carlo method, Water Res. 37 (2003) 4719–4729.
[5] J.E. Nash, J.V. Sutcliffe, River flow forecasting through conceptual models part i — a discussion of principles, J. Hydrol. 10 (1970) 282–290.
[6] H. Moradkhani, S. Sorooshian, H.V. Gupta, P.R. Houser, Dual state-parameter estimation of hydrological models using ensemble Kalman filter, Adv. Water Resour. 28 (2005) 135–147.
[7] Y. Chen, D. Zhang, Data assimilation for transient flow in geologic formations via ensemble Kalman filter, Adv. Water Resour. (2006) 1107–1122.
[8] H. Zhang, H.J. Hendricks Franssen, X. Han, J. Vrugt, H. Vereecken, Joint State and Parameter Estimation of Two Land Surface Models Using the Ensemble Kalman Filter and Particle, Filter. Hydrol. Earth Syst. Sci. Discuss. (2016) 1–39.