Contents lists available at ScienceDirect

MethodsX

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

Method Article

Performance analysis method for model-based irrigation strategies under uncertainty☆

F.D. Mondaca-Duarte[a,*], M. Heinen[b], S. van Mourik[a]

[a] Wageningen University, Farm Technology Group, P.O. Box 16, 6700 AH Wageningen, the Netherlands
[b] Wageningen Environmental Research, Wageningen UR, P.O. Box 47, 6700 AA Wageningen, the Netherlands

A B S T R A C T

There is a necessity to increase the performance of food production in agriculture, this means, that precise management support in farming systems is required to reduce water use and drainage while avoiding crop stress. Management support based on model predictions is used to increase the performance of food production. However, sources of uncertainty affect the model predictions. Uncertainty in soil properties and uncertain evapotranspiration translate into uncertain predictions, and consequently in risk of performance loss. This paper presents the code and method to analyze performance uncertainty (and risk of performance loss) due to uncertain circumstances. The method is based on using the De Graaf evapotranspiration model and the EMMAN3G model, a Richards equation-based soil water model, as modules to conduct a performance uncertainty study.

A R T I C L E I N F O
Method name: Uncertainty framework for model-based irrigation
Keywords: Monte Carlo, Drainage, Richards equation
Article history: Received 3 July 2020; Accepted 21 September 2020; Available online 26 September 2020

Specifications Table

Subject Area	Agricultural and Biological Sciences
More specific subject area	*Model-Based irrigation, Precision Farming, uncertainty analysis*
Method name	*Uncertainty framework for model-based irrigation*
Name and reference of original method	
Resource availability	

☆ **Direct Submission or Co-Submission** Co-submissions are papers that have been submitted alongside an original research paper accepted for publication by another Elsevier journal https://doi.org/10.1016/j.agwat.2019.105990
DOI of original article: 10.1016/j.agwat.2019.105990
* Corresponding author.
E-mail address: Francisco.mondacaduarte@wur.nl (F.D. Mondaca-Duarte).

Method details

In this contribution we present the code used in [3] to study the impact of uncertainty in evapotranspiration and soil parameters on drainage and crop stress predictions. For that, we adopted a model that mimics the real world in terms of the simulation model that describes water movement in soil (the EMMAN3G model). The software package (in Fortran) of the EMMAN3G, as described in [2], is included. Details of the model can be found in [3] and are briefly repeated here.

Based on the uncertainty in model input and model parameters, a Monte Carlo analysis was developed in order to compute the uncertainty in predictions. This framework was developed in a MATLAB 2018b environment.

Main concepts

Richards equation

The Richards equation describes the change of water content in a soil column based on the water pressure head, hydraulic conductivity, water uptake, and irrigation:

$$\frac{\partial\theta}{dt} = \frac{\partial}{\partial z}\left(K(\theta)\frac{\partial h(\theta)}{\partial z}\right) - \frac{\partial K(\theta)}{\partial z} - Sr. \qquad \left[\text{ml cm}^{-3}\ \text{d}^{-1}\right] \tag{1}$$

Here θ is the volumetric water content [ml cm^{-3}], t is the time, z is the vertical coordinate oriented positive downwards [cm], K is the hydraulic conductivity [cm d^{-1}], h is the pressure head [cm], and Sr is a sink term accounting for crop water uptake, uniformly distributed over the crop area and uniformly distributed within the first 30 cm of soil layer [ml cm^{-3} d^{-1}]. Eq. (1) is non-linear due to the non-linear constitutive relationships between h, θ, and K, as presented next, and thus needs to be solved numerically.

The Mualem-Van Genuchten relationship

The Van Genuchten function (Eq. (2); [1]) describes the water retention as a function of the pressure head according to

$$S(h) = \frac{\theta(h) - \theta_r}{\theta_s - \theta_r} = \begin{cases} \frac{1}{(1+|\alpha h|^n)^m} & h \le 0 \\ 1 & h > 0 \end{cases}. \qquad [-] \tag{2}$$

Here S is the effective saturation [dimensionless], θ_r is the residual water content, θ_s is the water content at saturation, and α [cm^{-1}], n [dimensionless], and m [dimensionless] are curve shape parameters. The hydraulic conductivity characteristic is given by the Mualem equation (Eq. (3)) [4] (with $m = 1 - 1/n$)

$$K_r(S) = \frac{K(S)}{K_s} = S^{\lambda}\left[1 - \left(1 - S^{\frac{1}{m}}\right)^m\right]^2. \qquad [-] \tag{3}$$

Here K_r is the relative hydraulic conductivity [dimensionless], K_s is equal to K at saturation [cm d^{-1}], and λ is a curve shape parameter [dimensionless].

Root water uptake

Under well-watered conditions it is known that root water uptake is proportional to the root length density distribution. Since the main aim of crop production in greenhouse conditions is to maintain optimal conditions in the root zone, root water uptake was assumed to be proportional to fractional root distribution in the root zone times the potential transpiration. When this yields a pressure head exceeding a certain threshold value, this is regarded as a signal of water limitation.

The De Graaf model

The De Graaf model (Eq. (4); [5]) is used to calculate the evapotranspiration inside a greenhouse based on the global radiation received, the additional radiation by artificial lighting, the air temperature inside the greenhouse, the additional air temperature supplied by heating pipes, and the ratio between actual crop length and maximum crop length. This is formulated as follows,

$$ET = \left(aR + b\left|T_{tube} - T_{gh}\right|\right)\frac{L}{L_{max}}. \qquad [\mathrm{mm}] \tag{4}$$

Here ET [mm] is the evapotranspiration, R [J cm^{-2}] is the global radiation outside the greenhouse combined with the radiation from supplementary light, T_{tube} [°C] is the temperature from heating pipes, T_{gh} [°C] is the greenhouse indoor air temperature, a [mm cm^2 J^{-1}] is an empirical crop factor for the effect of radiation, b [mm °C^{-1}] is an empirical crop factor of the heating influence, L is the current crop length, and L_{max} is the maximum crop length. In this study we focused on the final stage of the cropping period so that $L = L_{max}$.

The ET is then divided in the potential transpiration (demand for crop water uptake) T_{pot} and potential evaporation at the soil surface E_{pot} according to

$$T_{pot} = ET\left(1 - e^{(-0.6*LAI)}\right). \qquad [\mathrm{mm}] \tag{5}$$

$$E_{pot} = ET - T_{pot}. \qquad [\mathrm{mm}] \tag{6}$$

where LAI is the leaf area index. For a nearly full grown crop, as considered here, ET is dominantly assigned to T_{pot}, and, therefore, we used $T_{pot} = ET$ and $E_{pot} = 0$.

MATLAB framework code

The De Graaf model

First, the De Graaf model [5] was set up as a function

```
function ET = DeGraaf(r, t_gr)
% Compute evapotranspiration[mm] based on De graaf & Spaans, 1989, Voogt
% 2000 Voogt 2002
% ET         = Cumulative evapotranspiration          [mm]
% r          = Global radiation + radiation lightning   [J/cm^-2]
% t_tube     = Tube temperature [centigrades]
% t_gr       = Greenhouse air temperature [centigrades]
% a          = Empirical factor (radiation effect) [mm cm^2 J^-1]
% b          = Empirical factor (heat influence) [mm oC^-1]
% l          = Crop length [meters]
% l_max      = Maximum crop length [meters]


a = 1.8E-3;    % [mm cm^2 J^-1]
b = 1.8E-5;    % [mm oC^-1]
l = 0.5;       % [meters]
l_max = 0.5;   % [meters]
t_tube = t_gr;% [Centigrades]
indoor_r = 1/0.8; %Percentage of radiation from outside to inside green-
house mean (between 80% and 75%)
ET =  ((a*r*indoor_r)+(b*abs(t_tube-t_gr)))*(l/l_max);  % ET0 is the stand-
ard evapotranspiration rate, multiplied with the single crop coefficient
(Kc) forms the crop specific ETc. [cm h-1]
```

Starting the model framework and data input

A script is used to start the framework. First, any stored previous data is cleared and data sets of irrigation, temperature, and radiation are stored as single column double arrays. If multiple irrigation strategies are used then for each irrigation, the values are stored in a separate column, under *irrigation.mat*

```
%Clear previous runs
clear;
clc;
fclose all;

load 'greenhouse temperature.mat' % greenhouse air temperature data [oC]
load 'greenhouse radiation.mat'   % greenhouse radiation data [J/cm^-2]
load 'irrigation.mat'             % irrigation data [mm]
```

Next, the evapotranspiration function is run and stored as the variable *ET*

```
ET = DeGraaf(radiation, temperature); %call to the ET function
```

Simulation settings

An IF statement was used to select if the simulations will include uncertainty or not. The variables *et_unc* and *soil_unc* indicate if uncertainty will be present in the study (>0) or not (= 0). After, the number of days the dataset spans are included. Next, the possible clay and sandy soils names were included in cell values named *clay_soils* and *sand_soils*. If evapotranspiration is uncertain, the evapotranspiration data is divided into the total number of days the dataset represents and stored as the variable *ET_unc*. Depending on the type of soil, an IF statement stores the clay or sand type of soil as the variable t*ype_soil.*

```
%% Evapotranspiration and soil uncertainty input %%
% Soil uncertainty input, Clay = 2, Sand = 1 %
days = 32;    % number of days to study
et_unc = 0;   % is evapotranspiration uncertain? 1= yes, 0 = no
soil_unc = 0; % is soil uncertain, and type of soil? 1= yes, Sand; 2= yes,
Clay; 0 = no

% List of the recorded clay soils in the EMMAN3G input files (soilinfo.dat)
clay_soils =
{'''clay_soil_1''';'''clay_soil_2''';'''clay_soil_3''';'''clay_soil_4''';
'''clay_soil_5'''};

% List of the recorded sandy soils in the EMMAN3G input files
(soilinfo.dat)
sand_soils = {'''sand_soil_1''';'''sand
_soil_2''';'''sand_soil_3''';'''sand_soil_4'''; '''sand_soil_5''';'''sand
_soil_6''';'''sand_soil_7''';'''sand_soil_8'''};

% If statement for uncertain evapotranspiration. When et_unc = 1 a variable
ET_unc is included based on the daily evapotranspiration.
if et_unc == 1
    ET_unc =  reshape (ET, [], days);
end

% If statement for uncertain soils. When soil_unc = 1 the type of soil
selected is a clay soil. When soil_unc = 2 is a sandy soil.
if soil_unc == 2
    type_soil = clay_soils;
elseif soil_unc == 1
    type_soil = sand_soils;
end
```

After selecting whether soil or evapotranspiration uncertainty will be included, the file *PlantToSoil.dat* is read as a ASCII delimited file. Then, the number of different irrigation strategies to run, and the number of sampling repeats using Monte Carlo sampling, are established as variables.

```
%% Open PlantToSoil data and replaces with new values
plant_soil = dlmread(Pathfile\PlantToSoil.dat', ',',8,0); % Read EMMAN3G
PlantToSoil file for plant parameters

irrigation_strategies_size = size(irrigation,2); % Define the number of
strategies

monte_carlo_sampling = 1000; % Number of runs for Monte Carlo sampling
```

Monte Carlo sampling – evapotranspiration uncertainty

A FOR loop of the number of irrigation strategies is included as *irrigations_run*. Within the *irrigations_run* loop, another FOR loop of the number of times the framework will run depending on the number of Monte Carlo samplings as *model_run*. Within the *model_run* FOR loop, an IF statement and a FOR statement are added to indicate the Monte Carlo sampling of ET values. The FOR loop *ET_unc_run* includes a random sampling of *ET_unc* storing them as the variable *ET_sampling* to create a new *ET* variable for the current run simulation.

```
%FOR loop based on the number of irrigations strategies, used to select which
irrigation strategy will be studied
for irrigations_run = 1:irrigation_strategies_size
%FOR loop used in defining the number of Monte Carlo sampling iterations for
the uncertainty study
    for model_run = 1:monte_carlo_sampling

%% include uncertainty in evapotranspiration
    if et_unc == 1 %If statement when evapotranspiration is uncertain
        for ET_unc_run = 1:days %FOR loop defining the number of days that
evapotranspiration will be sampled
        ET_sampling(:,ET_unc_run) = ET_unc(:,randi([1,days],1,1)); %random
sampling of evapotranspiration values
        end
        ET = reshape(ET_sampling, [] , 1); %Reshaping the evapotranspiration
values into one data set for the next Monte Carlo sampling iteration
    end
```

Monte Carlo sampling – soil uncertainty

An IF statement is added to indicate if soil uncertainty is included in the study when the variable *soil_unc* is higher than zero. Within the IF statement, a Monte Carlo random sampling of the soil parameters is included from the soil list *type_soil* and stored as the variable *soil*. The *SoilInfo.dat* is opened using the ‘fopen’ command and giving the ID *fid_soil*. Data from *fid_soil* is read using the command ‘textscan’ and stored as a string variable *soil_data. Fid_soil* is closed using the ‘fclose’ command, and the *soilinfo.dat* file is re-opened using the ‘fopen’ command with the option to rewrite. The new soil values stored in the variable *soil_data* are written into *SoilInfo.dat* using the ‘fprintf’ command. Finally *fid_soil* is closed using the ‘fclose’ command.

```
%% Include uncertainty in Soil
%Inserting the random soil sampling
if soil_unc > 0 %If statement indicating uncertainty in soil
    soil = type_soil(randi([1,(length(type_soil))],1,1),1); %random sampling
from the different types of soils
    fid_soil = fopen(Pathfile\Data\Input\SoilInfo.dat','r'); %open soil-
info.dat file from the EMMAN3G model
    soil_data = textscan(fid_soil, '%s'); %store the data from soilinfo.dat
into the cell variable soil_data
    soil_data = soil_data{1, 1}; %Make the variable soil_data into a single
column cell
    soil_data (76,1) = soil; %Replace the soil layer with the random sampled
type of soil
    fclose(fid_soil); %close the file soilinfo.dat


%Re-writing soilinfo.dat
    fid_soil = fopen(Pathfile\Data\Input\SoilInfo.dat','wt');

    fprintf(fid_soil,' %s', soil_data{1:8});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{9:16});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{17:19});
    fprintf(fid_soil,'\n\n');
    fprintf(fid_soil,' %s', soil_data{20:35});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{36:38});
    fprintf(fid_soil,'\n\n');
    fprintf(fid_soil,' %s', soil_data{39:48});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{49:51});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{52:54});
    fprintf(fid_soil,'\n\n');
    fprintf(fid_soil,' %s', soil_data{55:65});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{66:72});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{73:74});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{75:76});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{77:78});
    fprintf(fid_soil,'\n');
    fprintf(fid_soil,' %s', soil_data{79:80});

    fclose(fid_soil);


end
```

Sending the case studies to the EMMAN3G model

New values for evapotranspiration, temperature and irrigation are written on the previous established variable *plant_soil*. Evaporation and root length are commented out as it was assumed in the study [3] that the crop was fully grown, so evaporation from soil was neglected, and root length was assumed to be a constant of 30 cm. The file *PlantToSoil.dat* is re-opened with writing rights using the 'fopen' command. The headers for the data are included with the 'fprintf' command. The data from *plant_soil* is written to *PlantToSoil.dat* using the command 'dlmwrite' (ASCII delimited file) with a specific precision required by the EMMAN3G model and using '-append' to append the matrix to the file, if not, 'dlmwrite' overwrites the previous included headers.

```
%% Input values into PlantToSoil.dat
 plant_soil (:,2) = ET; %Evapotranspiration values [mm]
%plant_soil (:,3) = evaporation; %Evaporation values [mm]
 plant_soil (:,4) = temperature; %Temperature data [centigrades]
 plant_soil (:,5) = irrigation (:,irrigations_run); %Irrigation data [mm]
%plant_soil (:,6) = root_lenght; %root lenght data [meters]


%% Write planttosoil.dat
fid_plant = fopen(Pathfile\Data\Input\PlantToSoil.dat','wt'); %open plantto-
soil.dat file with writing rights
%Include descriptions of the variables
fprintf(fid_plant,'! Uur     : hour (since start of simulation period)\n');
fprintf(fid_plant,'! Tpot    : potential transpiration during time period
(mm)\n');
fprintf(fid_plant,'! Evap    : soil evaporation during time period (mm)\n');
fprintf(fid_plant,'! Tavg    : average air temperature during time period
(oC)\n');
fprintf(fid_plant,'! Irrig   : amounts of irrigation water during time step
(mm)\n');
fprintf(fid_plant,'! SD      : rooting depth (cm)\n\n');
fprintf(fid_plant,'Uur,Tpot,Evap,Tavg,Irrig,SD\n'); %Overwrite the header
dlmwrite(Pathfile\Data\Input\PlantToSoil.dat' ,plant_soil,'delimit-
er',',','precision','%.5f','-append') %Overwrite the planttosoil.dat file with
the new values from variable plant_soil
fclose(fid_plant); %Close planttosoil.dat file
```

Running the EMMAN3G model

Before running the EMMAN3G model, the previous output file is deleted. EMMAN3G model is run using the 'system' command. A new *EmMan3G_W.csv* file with be created by the EMMAN3G model.

```
%% Delete previous output file
delete ('Pathfile\Data\Output\EmMan3G_W.csv')

%% Run EmMan3G
system('Pathfile\EmMan3G.exe');
```

Gathering results data

The *EMMAN3G_W.csv* file is opened and the number of columns are established as the variable *columns_number*. This is because, depending on the number of Z planes in the *LocalFlux.dat* file, the number of columns of the csv file will change. The data in the csv file is saved in the variable *csv_data* using the 'textscan' command. The fid is closed and the data is reshaped into the number of original rows and columns, and is stored in the variable *data*.

```
%% obtain drainage and pressure head from EmMan3G output csv file
columns_number = 51; %Columns from output csv file
fid_results = fopen('EmMan3G_W.csv', 'r'); %Open the output csv file
csv_data = textscan(fid_results, '%s','Delimiter',',','HeaderLines', 6); %Read
results from the csv file and store them
fclose(fid_results); %close the output csv file
csv_data = csv_data{1, 1}; %Store data as a cell
rows_number = length (csv_data)/columns_number; %define the number or rows of
the data
data = reshape(csv_data ,columns_number,rows_number); %Reshape the data by the
number of rows
data = data'; %Transpose the output data
```

Storing results data as MATLAB variables

The variable *data* is in strings format and it is required to be as double. A FOR loop is used to store the results from the data variable as cells for the results of drainage, water pressure head of the first soil layer (8 Z-planes) and crop transpiration.

```
%FOR loop used to obtain the drainage, pressure head and transpiration from
the raw data of the EMMAN3G model
for results_run = 1:length (data)
     drainage(results_run,1) = str2double (data {results_run,7}); %Store
drainage results

%Store the pressure head of the first eight Z-planes corresponding to the
first soil layer
     pressure_head (results_run,1) = str2double (data {results_run,38});
     pressure_head (results_run,2) = str2double (data {results_run,39});
     pressure_head (results_run,3) = str2double (data {results_run,40});
     pressure_head (results_run,4) = str2double (data {results_run,41});
     pressure_head (results_run,5) = str2double (data {results_run,42});
     pressure_head (results_run,6) = str2double (data {results_run,43});
     pressure_head (results_run,7) = str2double (data {results_run,44});
     pressure_head (results_run,8) = str2double (data {results_run,45});
     transpiration(results_run,1) = str2double (data {results_run,5}); %Store
transpiration results

 end
```

Each Monte Carlo sampling iteration outputs values of drainage, pressure head and transpiration predictions. The values stored in *drainage*, *pressure_head*, and *transpiration* are stored in new variables with the suffix _MC, which changes in which column the data is stored depending on the FOR loop *model_run*. Finally, the FOR loop *model_run* is closed after completing the Monte Carlo samplings.

```
drainage_MC(:,model_run) = drainage; %drainage values from a Monte Carlo
sampling iteration
pressure_head_MC {model_run} = pressure_head; %pressure head values from a
monte Carlo sampling iteration

transpiration_MC(:,model_run) = transpiration; %transpiration values from a
monte Carlo sampling iteration

    end
```

Finally, the Monte Carlo sampling results for drainage, crop transpiration and water pressure head are stored in a cell which changes cell position depending on the irrigation strategy FOR loop *irrigations_run*. The FOR loop *irrigations_run* is closed after completing the different irrigation strategies previously established.

```
    final_drainage{irrigations_run}  = drainage_MC; %Store of the multiple
drainage values from the monte carlo sampling iterations
    final_transpiration{irrigations_run} = transpiration_MC; %Store of the
multiple transpiration values from the monte carlo sampling iterations
    final_pressure_head{irrigations_run} = pressure_head_MC; %Store of the
multiple pressure head values from the monte carlo sampling iterations

end
```

Defining crop stress ratio

The crop stress ratio consists of values of the mean water pressure head that are lower than the selected variable *threshold*. The crop is considered to be under stress when the mean water pressure is below the threshold. Three nested FOR loops were used to run the crop water stress ratio. The first loop runs for the number of irrigation strategies established (*stress_irrigation_run*). The next loop runs

Table 1
Output tabulated data example. Mean and standard deviation values are stored in three variables.

plot_risk		plot_drainage		plot_ET	
mean	std	mean	std	mean	std
0.66	0.14	1.54	1.05	114.6	6.2
0.45	0.18	3.06	2.51	115.5	6.0
0.25	0.15	6.80	3.85	115.1	6.2
0.12	0.10	14.35	5.23	114.0	6.2
0.08	0.07	21.23	5.29	115.8	5.8
0.03	0.04	29.87	5.67	114.7	6.1
0.03	0.04	38.40	5.92	114.9	6.1
0.02	0.03	47.03	6.11	115.6	6.1

for the number of Monte Carlo samples (*stress_MC*). The third FOR loop runs for the number of water pressure head values stored in *final_pressure_head* variable.

The mean pressure head value is obtained from the mean of the bottom 4 Z planes in the first soil layer. The mean value is stored in the variable *mean_p_head*. Also, the variable mean_p_head stores the mean pressure head value from different irrigation strategies studied in separate rows of the variable. If the values of *mean_p_head* are below the *threshold* value then it is stored as a risk event under the variable *risk*. The crop stress ratio represents the ratio of hours the crop was below the threshold value over the total amount of hours. The ratio values are stored in a cell (*final_risk*), this cell changes cell positions to store the ratio values of the different irrigation strategies within the FOR loop *stress_irrigation_run.*

```
%% Crop water stress ratio
threshold = 75; %cm %crop stress pressure head threshold
stress_data_size = size(final_pressure_head{1,1}{1,1}); %define the size of
the stress data
%FOR loop based on the number of different irrigation strategies studied used
to select a specific irrigation strategy to study
for stress_irrigation_run = 1:stress_data_size(2)
%Nested FOR loop used in the Monte Carlo sampling iterations for stress based
on the number of iterations to study
    for stress_MC = 1:monte_carlo_sampling
%Nested FOR loop based on the stress data size, used to obtain the mean values
of the pressure head data
        for stress_data_run = 1:stress_data_size(1)
        mean_p_head(stress_data_run,stress_MC) = mean
      ((final_pressure_head{1,stress_irrigation_run}{1,stress_MC}(stress_data_
      run,5:end))); %mean values for the pressure head data
        end
    risk(:,stress_MC)= ((sum(mean_p_head(:,stress_MC) <= -threshold)));
%Definition of risk as the number of times the mean pressure head was below
the threshold risk value

    end
final_mean_p_head{stress_irrigation_run} = mean_p_head; %Saved mean pressure
head values
final_risk{stress_irrigation_run} = risk; %Saved risk values

end
```

Display of results

The mean and standard deviation values of the model outputs: drainage, risk of crop stress, and evapotranspiration was represented graphically. This was done with the FOR loop *plot_results*, storing the mean and standard deviation results in variables with the first column as the mean and the second column as the standard deviation. See Table 1 for a tabulated data example.

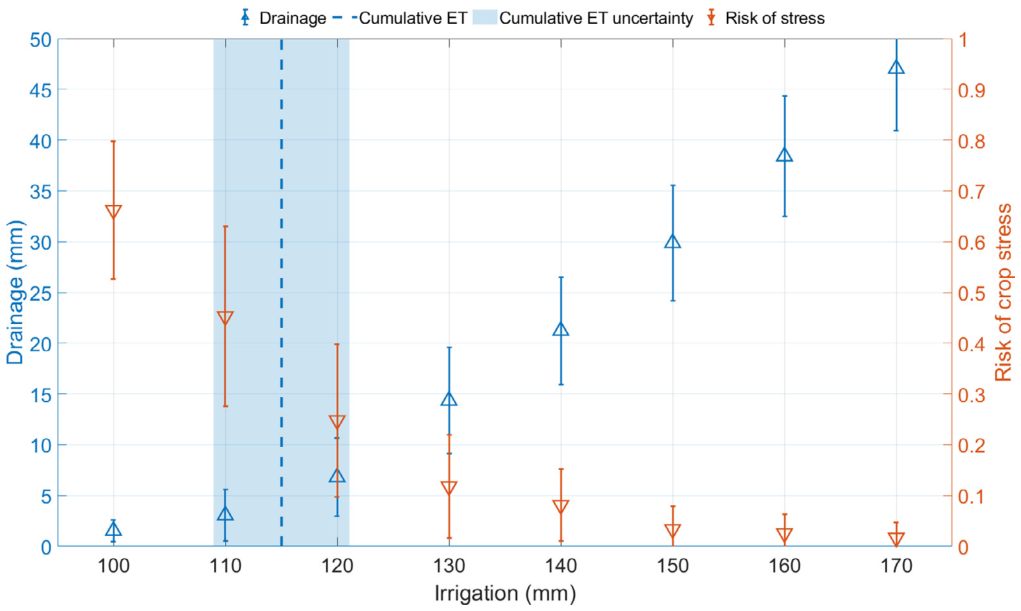

Fig. 1. Example of a case study using the tabulated data. The shadowed area represents the uncertainty in the cumulative evapotranspiration. The bars represent the standard deviation of the prediction due to uncertainty in evapotranspiration.

```
%% Display results %%
%Mean and STD
plot_data = size(final_drainage); %Stablish the size of the array plot_data
based on the size of drainage data

%FOR loop to generate plot data of drainage, risk and ET generating the mean
and standard deviation values
for plot_results = 1:plot_data(2)
    plot_drainage(plot_results,1) = mean (final_drainage{plot_re
    sults}(end,:));
    plot_drainage(plot_results,2) = std (final_drainage{plot_results}(end,:));
   plot_risk(plot_results,1) = (mean (final_risk{plot_re
   sults}(end,:)))/stress_data_size(1);
    plot_risk(plot_results,2) = std ((final_risk{plot_re-
   sults}(end,:)))/stress_data_size(1);
    plot_ET(plot_results,1) = mean(final_transpiration{plot_results}(end,:));
    plot_ET(plot_results,2) = std(final_transpiration{plot_results}(end,:));

end
```

A figure is created, the X-axis is established by including a variable (*total_irrigation*) which includes the different irrigation values used as irrigation strategies. The command ‘Yyaxis right’ is used to set an errorbar figure for the risk, using the *plot_risk* variable first column for the mean and the second column for the error. ‘Yyaxis left’ is used to set another errorbar for the drainage, using *plot_drainage* variable first column for the mean and the second column for the error. The uncertainty in evapotranspiration was represented as a set of limits between the mean and standard deviation of the *plot_ET* variable stored as the variable *rx1* for the lower limit and variable *rx2* for the higher limit. The space between *rx1* and *rx2* was filled an colored using the ‘area’ command. This colored area was made transparent using the ‘FaceAlpha’ and removing the line area with ‘LineStyle’. See Fig. 1 as an example of the figure created using the tabulated data.

```
%%Code used to create the plot figure
figure (1) %Define as figure 1
set(gca,'fontsize', 24); %Define plot
total_irrigation = [  100 110 120 130 140 150 160 170]; %The different values
for irrigation strategies are included
yyaxis right %Select the right Y-axis
%Plot of errorbars of the mean risk with the standard deviation of risk
errorbar (total_irrigation, plot_risk (:,1), plot_risk (:,2),'v',...
          'MarkerSize',16,...
          'LineWidth',2);
ylabel('Risk of crop stress') %Select the label for the Y-axis
axis([95 175 0 1]) %move the Y-axis to the selected points
xlabel('Irrigation (mm)') %Label for the X-axis
hold on

yyaxis left %Select the left Y-axis
%Plot of errorbars of the mean drainage with the standard deviation of drain-
age as its error
errorbar (total_irrigation,plot_drainage (:,1), plot_drainage (:,2),'^',...
          'MarkerSize',16,...
          'LineWidth',2);
rx1 = mean(plot_ET(:,1))-mean(plot_ET(:,2)); %Lower point of evapotranspira-
tion uncertainty
rx2 = mean(plot_ET(:,1))+mean(plot_ET(:,2)); %Higher point of evapotranspira-
tion uncertainty
%Drawn line representing the mean Evapotranspiration
line([mean(plot_ET(:,1)),mean(plot_ET(:,1))],[0,50],'linestyle','--',...
    'LineWidth',3);
r = [50,50]; %Height if the uncertainty area on the figure
xr =  [rx1,rx2]; %Width of the uncertainty area on the figure
a = area (xr,r); %Uncertainty area drawn on the figure
a.FaceAlpha = 0.2; %Transparency of the area
a.LineStyle = 'none'; %Remove line of the area
ylabel('Drainage (mm)') %Label for the right Y-axis
axis([95 175 0 50]) %move the Y-axis to the selected points
grid on %enable grid on the figure
legend({'Drainage','Cumulative ET','Cumulative ET uncertainty','Risk of
stress'},...
   'orientation','horizontal',...
   'FontSize',20);
legend('boxoff');
set(gca,'OuterPosition',[-0.02 0 1 1])
```

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We would like to thank the Consejo Nacional de Ciencias y Tecnologia (CoNaCyT) for financial support.

References

[1] Van Genuchten, M.. 1980. “A closed-form equation for predicting the hydraulic conductivity of unsaturated soils1.” Soil Sci. Soc. Am. J. 44.
[2] Heinen, M., and P. de Willigen. 1998. *FUSSIM2. A two-dimensional simulation model for water flow, solute transport, and root uptake of water and nutrients in partly unsaturated porous media*. Wageningen, The Netherlands. Available at: https://edepot.wur.nl/4408: Quantitative Approaches in Systems Analysis No. 20, DLO Research Institute for Agrobiology and Soil Fertility and the C.T. de Wit Graduate School for Production Ecology. http://edepot.wur.nl/4408.

[3] Mondaca-Duarte, F.D. et al. 2020. “Irrigation, crop stress and drainage reduction under uncertainty: a scenario study.” Agric. Water Manag. 230: 105990. http://www.sciencedirect.com/science/article/pii/S0378377419314076.
[4] Mualem, Y.. 1976. “A new model for predicting the hydraulic conductivity of unsaturated porous media.” Water Resour. Res. 12(3): 513–22. https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/WR012i003p00513.
[5] Voogt, W., J.A. Kipp, R. de Graaf, and L. Spaans. 2000. “A fertigation model for glasshouse crops grown in soil.” In *Acta Horticulturae*, International Society for Horticultural Science (ISHS), Leuven, Belgium, 495–502. doi:10.17660/ActaHortic.2000.537.58.