Documentation of Edcrop - version 1

A python package to simulate field-scale evapotranspiration and drainage from crop, wetland, or forest

Steen Christensen, assoc. prof. emeritus
Department of Geoscience
Faculty of Natural Sciences

AARHUS UNIVERSITY

ISBN: 978-87-7507-566-9
DOI: 10.7146/aul.539

Preface

This report documents Edcrop, a Python package designed to simulate local evapotranspiration. The author has been coding since the late seventies, first in Algol W and Fortran, later also in C. A few years ago, many students and younger colleagues told me, that python was the programming language of the future. To begin to learn python, I decided to code a modified version of the Evacrop program (Olesen and Heidmann, 2002), which through a couple of decades has been useful for many of our students to simulate daily evapotranspiration and drainage from a field with a crop on basis of climatic data. However, today Evacrop is difficult or impossible to use due to its coding and compilation in Turbo Pascal. This made me choose to code it in python and distribute it for everyone to use. During the development, I made several new developments and additions to Evacrop, for which reason the name of the developed code (python package) is Edcrop – short for "Evapotranspiration and Drainage from CROP, wetland, or forest".

September 27th, 2024.

Steen Christensen
Geoscience, Aarhus University

Table of Contents

1 Introduction

Evapotranspiration, the sum of evaporation and plant transpiration from land and ocean surfaces, is a major component of Earth's water balance. Factors that affect evapotranspiration includes solar radiation, wind, humidity, temperature, growth stage of vegetation, and water availability. Water availability depends on factors such as precipitation, irrigation, and soil characteristics.

Local- or field-scale evapotranspiration can be measured using a weighing lysimeter, Bowen-ratio energy balance, or eddy covariance, but these methods are expensive, error-prone, or require correction (Healy, 2010). Alternatively, field-scale evapotranspiration can be estimated from climatic data by simulating the water balance of an area with a specific vegetation growing on a specific soil. This report documents a code (a package), programmed in Python and named Edcrop, which can do such local simulations for various types of soil and vegetation. It does not simulate surface flow, lateral flow, or processes occurring in the saturated zone, such as water loss to drains. The water balance equation of Edcrop is therefore

$$P = E_a + D + \Delta V \tag{1}$$

where P is precipitation (possibly including irrigation), E_a is actual evapotranspiration, D is downward drainage to the unsaturated zone beyond the root zone, and ΔV is change in water storage. Evapotranspiration includes the evaporation of snow, intercepted water, and soil water, as well as the transpiration of plants. ΔV includes changes in snow pack, intercepted water, and water content in the root zone and subzone. The subzone is the zone between the bottom of the root zone and the bottom of the model's soil profile.

The conceptual model implemented in Edcrop is a modification of the Evacrop model by Olesen and Heidmann (2002), which was based on the Watcros model (Aslyng and Hansen, 1982). The Edcrop conceptualization is based on considerations regarding the physical processes that are important for turning precipitation and irrigation into either evaporation, transpiration, or drainage from the root zone: Temperature determines whether precipitation falls as rain or snow, and when snow thaws and infiltrates. The vegetation intercepts a part of precipitation, while the rest infiltrates into the ground. The infiltrated water will either evaporate, be absorbed by plant roots, be stored in the soil, or drain from the root zone. Potential evaporation is divided between vegetation and soil: the former drives evaporation of intercepted water and transpiration from the green leaf area, while the latter drives soil evaporation. The soil's ability to store water is determined by its field capacity. When soil moisture exceeds this capacity, water gradually drains downwards. Furthermore, it is assumed that the annual life cycle of crops and wetland vegetation is driven by growing degree-days only (and not by any other climatic variables), while for forests the life cycle is determined by a calendar. For irrigation, either (i) date and amount are input, or (ii) they are determined automatically by Edcrop using certain criteria.

There are two alternative soil water balance functions to choose between in Edcrop. The first alternative is an almost straight copy of the function used in the original Evacrop code by Olesen and Heidmann (2002), simulating flow through the soil profile as flow through two linear reservoirs using daily time steps. However, it can simulate macro-pore drainage, which the original Evacrop cannot. The second alternative simulates flow through the soil profile as flow through four linear or nonlinear reservoirs using daily or sub-daily time steps. For nonlinear reservoirs, Edcrop uses Mualem – van Genuchten like functions. It also simulates gravity driven macro-pore flow as well as loss of infiltration due to surface runoff.

As input, given in text files, Edcrop requires daily temperature, precipitation, and reference evapotranspiration. It also requires information about combination(s) of soil type and vegetation type to simulate. One can choose between seven default soil types and fifteen default vegetation types, or one can manually input information for other types of soil or vegetation. In a single model run, Edcrop can loop through lists of climate files, soils, and vegetation.

The seven default soil types vary from coarse sandy soil to clayey soil. The fifteen default vegetation types include bare soil, ten types of crop, two types of forest, and two types of wetland.

As said, the water balance simulation of Edcrop is similar to that of Evacrop (Olesen and Heidmann, 2002), but in other ways Edcrop is different from Evacrop. Edcrop allows more flexible and easier specification of input and output; it can loop through lists of climate, soil, and vegetation combinations in a single model run; it simulates macro-pore drainage; it contains more crops than Evacrop; it contains forest and wetland types, which are new compared to Evacrop; it has a more advanced irrigation module; and data and results can be plotted.

Edcrop cannot simulate capillary rise of shallow groundwater to the root zone or surface. If downward drainage (just called drainage in the following) simulated by Edcrop is to be used as recharge input for a groundwater model, there can be ways to partly correct for this lack of Edcrop ability. For example, for simulation of groundwater flow using Modflow (McDonald and Harbaugh, 1988), drainage from Edcrop can be used as recharge input for the Modflow RCH package, while the difference between Edcrop's potential evapotranspiration and actual evapotranspiration can be used as maximum ET input for the Modflow EVT package. Similar can be done using newer versions of Modflow.

In the following, Chapter 2 instructs how to install and run Edcrop. Chapter 3 gives details about the Evacrop conceptual model and the equations used for this by Edcrop. Chapter 4 gives details about the alternative soil water balance model of Edcrop. Chapter 5 supplemented by the Tables of Chapter 7 give the input instructions. Chapter 6 explains the output files. Chapter 8 gives the list of references. Appendix A compares results obtained by using the alternative water balance functions of Edcrop with alternative settings. Appendix B illustrate similarities and differences in simulation results obtained by the simpler Edcrop code and the more advanced code named Daisy (Hansen et al., 1990), respectively.

2 Installing and running Edcrop

2.1 Installation

Edcrop (v. 1.0.0) requires Python 3.8.8 or higher. Furthermore it requires the numpy (v. 1.20.1), pandas (v. 1.2.4), matplotlib (v. 3.3.4), and yaml (pyyaml v. 5.4.1) packages.

Edcrop is available from the Python Package Index (PyPI.org) repository. It is installed by executing from the command line

pip install edcrop

This installation uses the following dependencies:

```
dependencies = [
   "numpy >=1.20.1",
   "matplotlib >= 3.3.4",
   "pandas >=1.2.4",
   "pyyaml >=5.4.1",
]
```

To install Edcrop without caring for dependencies, use instead

pip install --no-deps edcrop

The above commands install Edcrop into the Python site-packages directory. The edcrop directory contains a sub-directory named Examples, containing example of data files required to make Edcrop run.

"For more information on pip installation, consult the pip documentation at https://pip.pypa.io/en/stable/cli/pip_install/.

Edcrop and example releases can also be found on https://github.com/SteenChr/edcrop.

A website with a brief documentation of Edcrop, mainly developed by Paul J.M. McLachlan, can be found here: https://steenchr.github.io/edcrop.

2.2 Running Edcrop

There are two ways to run Edcrop.

The first way is to run Edcrop from own script by including in the script both the statement

from edcrop import edcrop

and the function call

edcrop.run_model().

The function call may include two optional arguments

yaml=<name of yaml input file>

log=<name of log output file>

with the defaults

yaml='edcrop.yaml'

log='edcrop.log'

The second way is to run Edcrop as a script by executing from the command line

python –m edcrop

with the optional arguments

--yaml <name of yaml input file>

--log <name of log output file>

3 Description of Evacrop – a conceptual model for evapotranspiration and drainage

This chapter describes the conceptual model of Edcrop for simulation of evapotranspiration from areas with crop, wetland, or forest. To run simulation, Edcrop requires input of climatic time series and of soil and vegetation parameters. Figure 1 visualizes the conceptualization, which is similar to that of the Evacrop model by Olesen and Heidmann (2002), which built on the Watcros model by Aslyng and Hansen (1982). The conceptualization is based on considerations of the physical processes that convert precipitation into either evapotranspiration or drainage. The conceptualization of processes is formulated as equations that are often semi-empirical (Olesen and Heidmann, 2002). The considerations are the following.

1. Precipitation falls as either rain or snow. When it falls as snow, it accumulates on the vegetation or on the ground. Snow evaporates, or it melts, infiltrates, and becomes available for plant roots or for drainage.
2. Part of the precipitation is intercepted by the leaves (and branches) of the vegetation, while the rest falls on the ground. There is a limit to how much water can be intercepted by the vegetation. Intercepted water will be available to evaporation.
3. Rain falling or snow thawing on the ground infiltrates the surface. The infiltrated water can evaporate, be captured by plant roots and transpired, be stored in the soil, or drainage if the water content exceeds field capacity of the soil.
4. During the life cycle of vegetation, leaf area and root depth may develop and disappear. This determines interception of precipitation, absorption of infiltrated water by plant roots, and distribution of potential evapotranspiration between vegetation and soil.
5. Intercepted water and soil water evaporate when potential evaporation is available for either or both.
6. Water will transpire from the stomata of the green leaf area when intercepted water has evaporated and potential transpiration is available.
7. The life cycle of a crop depends on climatic and other parameters. However, growing degree-days can often be used as a fair predictor for crop growth.
8. Growing of a tree is also growing degree-day dependent, but the dependency is very different for different tree species (Murray et al., 1989). Since a forest often contains a mix of tree species, as an approximation, growth of a mixed forest may be described by a calendar.

The following expands these considerations and turns them into equations that can be solved in sequence to calculate how a time series of precipitation turns into time series of actual evapotranspiration and drainage. For simplification, we assume that the following are available: time series of daily precipitation, mean temperature, and reference evaporation; soil data; and vegetation data. The required soil and vegetation data will be explained below.

The water balance calculation is carried out in daily time steps. The daily water balance calculations involve six model elements for which a sequence of equations needs to be solved. When solved for a particular day, the calculations proceed to the following day.

There is also a seventh model element, modeling of vegetation growth. Edcrop models growth independently from the water balance by predicting growth from alone either growing degree-days or a calendar.

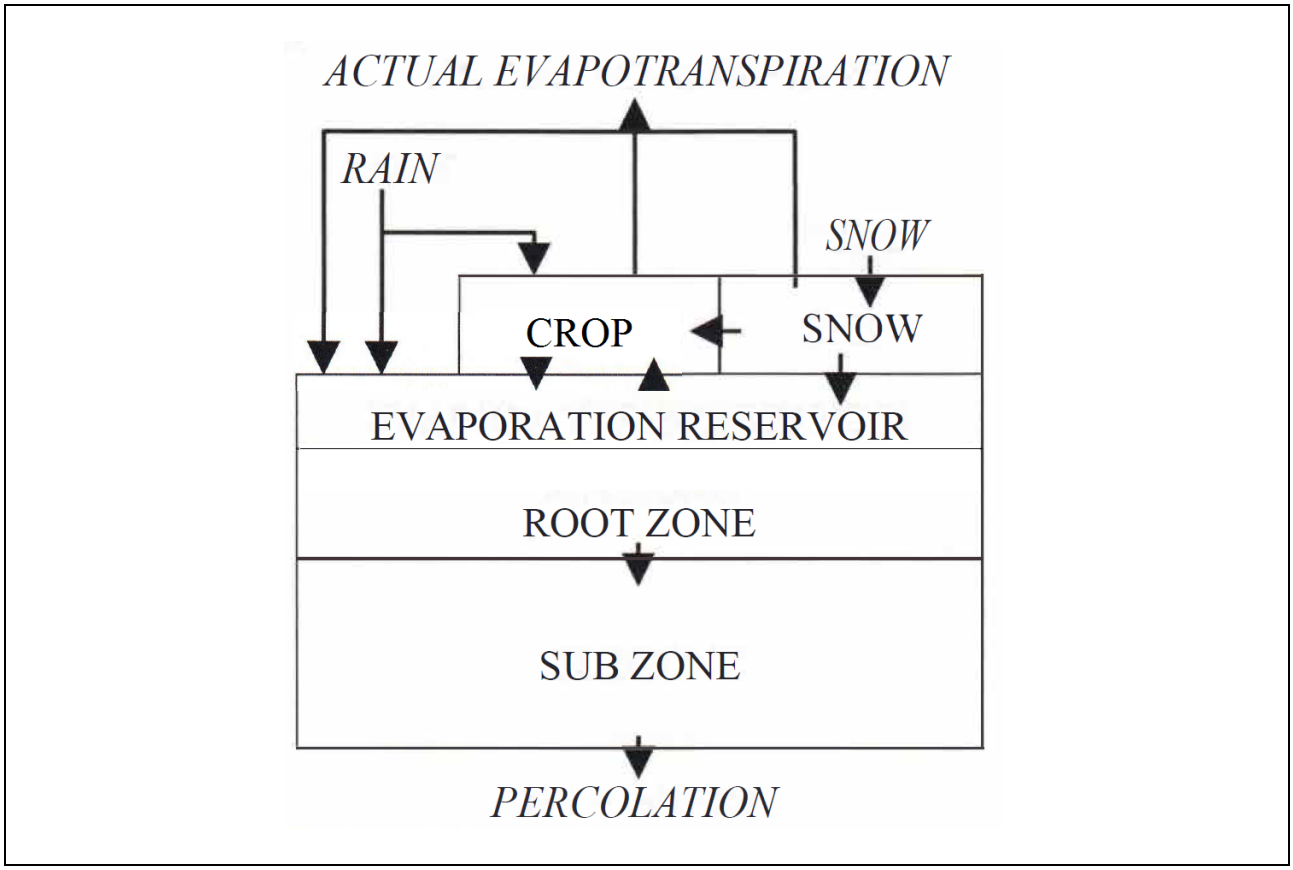

Figure 1 The Evacrop conceptualization of evapotranspiration and percolation (drainage) from a field with a crop. (Modified from Olesen and Heidmann, 2002.) The soil profile has depth z_{max}, which is divided into four subintervals of equal thickness but different soil parameters. The depth of the root zone changes over the growing season.

3.1 Model element 1: Precipitation falling as rain or snow

The model includes a snow reservoir that simulates the accumulation and melting of snow. The model does not impose a limit on snow accumulation, allowing for continuous buildup in appropriate conditions. When the daily mean temperature, T, is below a threshold, T_m, the daily precipitation is accumulated as snow. When the daily mean temperature is above the threshold, snow melts at a rate proportional to the difference between T and T_m. The proportionality factor, c_m, is called the degree-day factor for snow melt. It expresses the millimeters water-equivalent of snowmelt per degree temperature per day [mm w. e. $°C^{-1}$ day^{-1}] and is often set to 2 mm $°C^{-1}$ day^{-1}.

The daily actual evaporation from the snow reservoir, E_{as}, is limited by both the water content of the snow reservoir, V_s, and the daily potential evaporation, E_p. This is expressed by

$$E_{as} = \min\left[V_{s,0} + P_s, E_p\right] \quad (2)$$

where the snow accumulation, P_s, is first calculated as

$$P_s = \begin{cases} P; \text{ for } T \le T_m \\ 0; \text{ for } T > T_m \end{cases} \quad (3)$$

In (2), $V_{s,0}$ is the amount of snow accumulated at the beginning of the day (i.e. at the end of the previous day).

Snowmelt, P_m, is calculated as

$$P_m = \begin{cases} 0 & ; \text{ for } T \le T_m \\ \min\left[V_s - E_{as}, c_m\left(T - T_m\right)\right]; & \text{for } T > T_m \end{cases} \quad (4)$$

The water stored in the snow reservoir at the end of the day is calculated as

$$V_s = V_{s,0} + P_s - P_m - E_{as} \quad (5)$$

The amount of precipitation falling as rain during the day is calculated as

$$P_r = \begin{cases} 0; \text{ for } T \le T_m \\ P; \text{ for } T > T_m \end{cases} \quad (6)$$

3.2 Model element 2: Distribution of potential evaporation between vegetation and soil

Energy supplied from the atmosphere drives evapotranspiration. Only part of the energy reaches the ground when the ground is shadowed by vegetation; the leaves (and branches) of the vegetation attenuate a proportion of the energy. Potential evapotranspiration quantifies the energy available for evapotranspiration from a vegetation growing on a soil. Inspired by Beer's law for attenuation of radiation passing through a homogeneous medium, the potential evaporation ("energy") not attenuated by the vegetation is simulated as

$$E_{pe} = \left(E_p - E_{as}\right)\exp\left(-k_p L\right) \quad (7)$$

where k_p is the extinction coefficient (also called attenuation coefficient), and L is the leaf area index (an index quantifying the density of the leaves of the vegetation) which will be discussed later. The extinction coefficient k_p can for example be set to 0.6 (Olesen and Heidmann, 2002; Aslyng and Hansen, 1982, p. 58). The amount of potential evapotranspiration attenuated by the vegetation will thus be

$$E_{pc} = \left(E_p - E_{as}\right)\left(1 - \exp\left(-k_p L\right)\right) \quad (8)$$

When part of the leaf area, L_g, is "green" (meaning "alive") while the rest is "yellow" (meaning "dead" or "wilted"), E_{pc} is divided into

$$E_{pcg} = \left(E_p - E_{as}\right)\left(1 - \exp\left(-k_p L_g\right)\right) \quad (9)$$

and

$$E_{pcy} = E_{pc} - E_{pcg} \quad (10)$$

E_{pcg} is used for evaporation and transpiration from the green leaves. E_{pcy} is used for evaporation from the yellow leaves, which do not contribute to transpiration.

3.3 Model element 3: Interception and its evaporation

Rain and snowmelt will be intercepted by the vegetation. The intercepted water makes a thin film on the leaves (and branches), but the vegetation only has a certain capacity to intercept water. When the capacity is exceeded, the exceeding water drops to the ground and infiltrates.

The interception capacity, C_I, depends on the density of the vegetation and is calculated as

$$C_I = C_{I,\min} + c_i L \tag{11}$$

where $C_{I,\min}$ [mm] is a minimum capacity, c_i is a capacity constant [mm], and L is the leaf area index. For example, the capacity constant can be set to 0.5 mm (Olesen and Heidmann, 2002; Aslyng and Hansen, 1982, p. 58). The minimum capacity can be set to zero for crops. For forests, a value larger than zero can be used, as water can be intercepted on stems and branches even when the trees have no leaves. The amount of intercepted water, and the amount that evaporates during the day, is calculated as follows.

First, the amount of water stored by interception is calculated as

$$V_I^* = \min[C_I, V_I^0 + P_r + P_m + I] \tag{12}$$

V_I^0 is the amount of intercepted water at the beginning of the day, and I is applied irrigation. Equation (12) means that interception cannot exceed the interception capacity, C_I. If $V_I^0 + P_r + P_m + I$ exceeds C_I, the remainder will infiltrate. The infiltration is calculated from

$$P_I = P_r + P_m + I - \left(V_I^* - V_I^0\right) \tag{13}$$

If water is intercepted, the actual evaporation from green leaf area is calculated as

$$E_{alg} = \begin{cases} \min\left[\dfrac{V_I^*}{C_I} c_i L_g, E_{pcg}\right] & ;\ \text{for } L > 0 \\ 0 & ;\ \text{for } L = 0 \end{cases} \tag{14}$$

while the actual evaporation from yellow leaf area (and branches and stems) is calculated as

$$E_{aly} = \begin{cases} \min\left[\dfrac{V_I^*}{C_I}\left(C_{I,\min} + c_i L_y\right), E_{pcy}\right] & ;\ \text{for } L > 0 \\ 0 & ;\ \text{for } L = 0 \end{cases} \tag{15}$$

In (14) and (15), L_g and L_y is the green and yellow leaf area, respectively.

The total actual evaporation of intercepted water is then

$$E_{aI} = E_{alg} + E_{aly} \tag{16}$$

Finally calculate the amount of water intercepted at the end of the day (i.e. after evaporation) as

$$V_I = V_I^* - E_{aI} \tag{17}$$

If E_{pcg} exceeds E_{alg}, then the exceeding E_{pcg} is available for transpiration from the stomata of the green leaf area. The available potential transpiration for the day is thus calculated as

$$E_{pT} = E_{pcg} - E_{alg} \tag{18}$$

3.4 Model element 4: Infiltration and evaporation from the soil

Rain and snowmelt that reach the ground surface will infiltrate into the soil. The daily infiltration, P_I, can be calculated from (13) as explained above.

Infiltrated water can evaporate when potential evaporation reaches the ground surface beneath the vegetation. This potential evaporation, E_{pe}, is calculated by (7) above. The actual evaporation depends on the availability of water from various depths of the soil profile. If sufficient water is available just below the ground surface, it can evaporate at the potential rate, E_{pe}. If insufficient water is available close to the surface, capillary rise can supply water from deeper parts of the soil profile that will evaporate. Capillary rise balances with actual soil evaporation but occurs slowly, resulting in actual evaporation often being lower than E_{pe}.

To model the evaporation from a soil, Olesen and Heidmann (2002) consider a soil profile of a certain depth, $z_{\max}$, subdivided into four parts (horizons) of equal thickness. Part or the whole of this profile contains roots of the vegetation. The root zone contain roots, while the deeper part of the profile, without roots, is the subzone. The root zone has a capacity to hold water. When the water content exceeds the capacity of the root zone, the remainder will drain to the deeper subzone. Similarly, when the water content of the subzone exceeds its capacity to hold water, the remainder will drain from the subzone (to the unsaturated zone beneath the subzone). Chapter 3.6 describes simulation of the drainage processes. Edcrop does not simulate flow through the unsaturated zone.

The uppermost part of the root zone, the top soil, constitutes the evaporation zone. The water content of the evaporation zone, V_e, is available for direct evaporation. The evaporation zone has a capacity, C_e, to store water; V_e cannot exceed C_e. The evaporation zone capacity is typically set to a constant value of $C_e = 10$ mm (Olesen and Heidmann, 2002, p. 11; Aslyng and Hansen, 1982, p. 59). The root zone capacity is calculated as

$$C_r = \max\left[C_e, \theta_F z_r\right] \tag{19}$$

where θ_F is the plant available water content of the soil, and z_r is the depth of the root zone. (The plant available water content is the difference between water content at field capacity and water content at permanent wilting point.) Similarly, the subzone beneath the root zone has a capacity to hold water that is calculated as

$$C_b = \theta_F \left(z_{\max} - z_r\right) \tag{20}$$

Because the root zone depth may change from day to day during the life cycle of the vegetation, C_r and C_b need to be updated on a daily basis according to (19) and (20), respectively. Similarly, the water content of the root zone and subzone, respectively, must also be updated when the root depth changes. The water content in the root zone, before accounting for infiltration, is calculated as

$$V_r^* = \begin{cases} V_r^0 + \left(C_r - C_r^0\right) V_r^0 \Big/ C_r^0 & ; C_r \le C_r^0 \\ V_r^0 + \left(C_r - C_r^0\right) V_b^0 \Big/ C_b^0 & ; C_r > C_r^0 \end{cases} \tag{21}$$

The first expression of (21) is for a shrinking root zone (decreasing root depth), while the second expression is for an expanding root zone (increasing root depth). In (21), V_r^0 and V_b^0 are the water contents of the root zone and the subzone at the end of the previous day, and C_r^0 and C_b^0 are the root zone and subzone capacities from the previous day. The correspondingly updated water content of the subzone is calculated as

$$V_b^* = V_b^0 - \left(V_r^* - V_r^0\right) \tag{22}$$

When water infiltrates, it first wets the evaporation zone (top soil), changing the water content of the evaporation zone to

$$V_e^* = V_e^0 + P_I \tag{23}$$

where V_e^0 is the water content of the evaporation zone at the end of the previous day. Since the evaporation zone constitutes the uppermost part of the root zone, the root zone receives the infiltration and its water content will increase to

$$V_r^* = V_r^* + P_I \tag{24}$$

With these updated water contents, Olesen and Heidmann (2002) and Edcrop calculates the actual evaporation from the soil as

$$E_{ae} = \begin{cases} E_{pe} & ; E_{pe} \le V_e^* \\ c_e E_{pe} & ; E_{pe} \le V_r^* + V_b^* \\ 0 & ; E_{pe} > V_r^* + V_b^* \end{cases} \tag{25}$$

The first expression in (25) conceptualizes that when infiltration is high (exceeds potential evaporation), sufficient water is available in the soil to evaporate at potential rate. The second expression conceptualizes that when water content in the evaporation zone is insufficient, but water content within the root and sub zones exceeds the potential evaporation, capillary forces will drag water towards the surface at a rate that is sufficient to feed evaporation constituting a fraction of potential evaporation; in Edcrop, the constant c_e is set to 0.15 (Olesen and Heidmann, 2002, p. 19; Aslyng and Hansen, 1982, p. 59). The third expression in (25) conceptualizes that when potential evaporation is high relative to the water content of the root and sub zones, capillary forces will be insufficient to feed water for evaporation; as an approximation, the actual evaporation is set to zero.

After infiltration and evaporation, exceeding water will percolate from the evaporation zone deeper into the root zone. When actual evaporation is high, the evaporation reservoir can dry out. To mimic this, the water content of the evaporation zone at the end of the day is calculated as

$$V_e = \min\left[C_e, \max\left[0, V_e^* - E_{ae}\right]\right] \tag{26}$$

The water contents after evaporation from the root and sub zones are then updated:

$$V_r = \max\left[0, V_r^* - E_{ae}\right] \tag{27}$$

$$V_b = \max\left[0, V_b^* - E_{ae} + V_r^* - V_r\right] \tag{28}$$

With regard to the soil profile in Edcrop, it has a capacity to hold water that vary between 25 cm depth intervals. The root zone capacity and the subzone capacity is therefore calculated as

$$C_r = \max\left[C_e, \left(\sum_{i=1}^{I-1} \theta_{Fi}\Delta z\right) + \theta_{FI}\left(z_r - (I-1)\Delta z\right)\right] \tag{29}$$

$$C_b = \left(\sum_{i=1}^{nz} \theta_{Fi}\Delta z\right) - C_r \tag{30}$$

where nz is the number of soil layers within the soil profile, Δz is the thickness of each soil layer (here assumed to be constant), I is the number of the deepest layer containing roots, and θ_{Fi} is the plant available water content of soil layer number i. (In Edcrop, $nz = 4$, but this could be changed by changing ModelParameters.ndz, SoilParameters.thf, and SoilParameters.MvG_soilhorizons. The depth of the soil profile, $z_{\max}$, could be changed from the default, 1.0 m, in the input cf. Chapter 7.1. Edcrop internally computes $\Delta z = z_{\max}/nz$.)

3.5 Model element 5: Transpiration from vegetation

Plant roots absorb water from the soil by osmosis, a process where water moves across a semi-permeable membrane. The rate of absorption depends on the soil's hydraulic conductivity and water content. The plant uses a small percentage of the absorbed water for growth and metabolism, while the large part is lost from the leaves by transpiration. The transpiration happens through the stomata (small pores) of the leaves that can open and close to allow diffusion of carbon dioxide from the air to the plant for its photosynthesis. The cost of opening stomata is loss of water by transpiration (meaning evaporation when the water potential in the ambient air is lower than the water potential in the stomata). Water potential differences and capillary forces drive the water movement from roots to leaves. The plant also benefits from the transpiration. Transpiration cools the plant, changes the osmotic pressure of its cells, and enables mass flow of mineral nutrients and water from the roots to the shoots.

To model transpiration, Olesen and Heidmann (2002) do the following. First they assume that there is a relationship between relative transpiration and relative root zone water content as shown in Figure 2. When the water content V_r of the root zone exceeds a fraction c_b of the root zone capacity C_r, it is assumed that the plant can absorb water from the soil and transpire at a rate corresponding to potential transpiration. For root zone water content falling below $c_b C_r$, it is assumed that water absorption and transpiration decrease linearly with the water content. Olesen and Heidmann (2002) furthermore assume that c_b depends on crop type and varies during the year; it is small during winter, when the potential evapotranspiration is small, and large during summer, when the potential evapotranspiration is large. The parameter c_b is called the break point of the transpiration function.

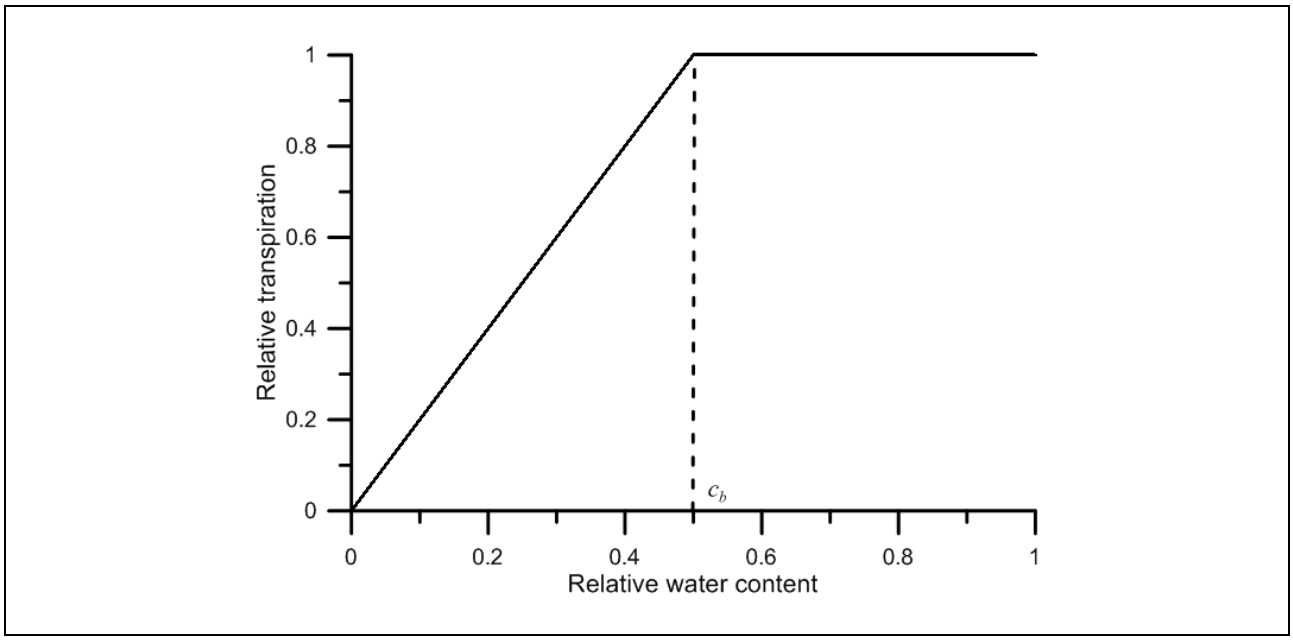

Figure 2 Relationship between water content in soil and relative transpiration from vegetation. The breakpoint is at relative water content c_b.

Secondly, Olesen and Heidmann (2002) divide the root zone into an upper part and a deeper part. The upper root zone is considered only when the water content of the entire root zone is insufficient to allow transpiration from the plant at potential rate. In this case, infiltrating water will fill the upper root zone to near field capacity, and the vegetation will use this water to transpire at potential rate. In the other case, when the infiltration is large enough to raise the water content of the entire root zone, V_r, to exceed $c_b C_r$, the upper root zone vanishes (merges with the deeper part) and transpiration will be taken from the entire root zone. The capacity of the upper root zone, C_u, therefore varies dynamically. The reasoning for introducing the upper root zone to the model is actually unclear to the author of this report; experimentation has shown that it has only very little effect on simulation results for actual transpiration. However, the upper root zone model is kept here to be in accordance with Olesen and Heidmann (2002).

The dynamics of the upper root zone is modelled by the following set of equations. First, if the water content of the entire root zone is sufficient to cover transpiration at potential rate, the upper root zone vanishes; that is, when $V_r \geq c_b \cdot C_r$ then

$$V_u = 0 \tag{31}$$

$$C_u = 0 \tag{32}$$

In the other case, when $V_r < c_b \cdot C_r$ then add the amount of infiltrating water to the upper root zone:

$$V_u = V_u + P_l - E_{ae} \tag{33}$$

$$C_u = \min\left[C_r, C_u + \max\left[0, (P_l - E_{ae}\right]\right] \tag{34}$$

Finally, if the water content of the upper root zone resulting from (33) is insufficient to cover transpiration at potential rate, the upper root zone vanishes. That is, if $V_u < c_b C_u$ or $V_u < E_{pT}$ then update V_u and C_u by (31) and (32), respectively.

The actual transpiration is then calculated by

$$E_{aT} = \begin{cases} E_{pT} & ; \text{for } V_u > 0 \text{ or } V_r \geq c_b C_r \\ E_{pT} \cdot \dfrac{V_r}{c_b \cdot C_r} & ; \text{for } V_u = 0 \text{ and } 0 < V_r < c_b C_r \\ 0 & ; \text{for } V_u = 0 \text{ and } V_r = 0 \end{cases} \tag{35}$$

checking that it does not exceed the water content of the entire root zone:

$$E_{aT} = \min\left[V_r, E_{aT}\right] \tag{36}$$

Finally, the actual evapotranspiration is calculated as

$$E_a = E_{as} + E_{ae} + E_{al} + E_{aT} \tag{37}$$

and the root zone water contents must be updated by subtracting actual transpiration:

$$V_r = V_r - E_{aT} \tag{38}$$

$$V_u = \max\left[0, V_u - E_{aT}\right] \tag{39}$$

3.6 Model element 6: Drainage from root zone and subzone

When soil water content exceeds root zone capacity, the excess water drains to the subzone at a rate of

$$D_r = \left[k_{qr} + \left(1 - k_{qr}\right) \frac{z_{max} - z_r}{z_{max}} \right] \cdot \max\left[0, V_r - C_r\right] \tag{40}$$

where k_{qr} is a drainage constant, z_{max} is the maximum soil depth modelled (for example 1 meter), and z_r is the actual root zone depth. Root zone depth varies during the year as described in chapter 3.7.

Similarly, when the water content of the subzone exceeds its capacity, excess water drains from the subzone. The drainage rate from the subzone is calculated as

$$D_b = \left[k_{qb} + \left(1 - k_{qb}\right) \frac{z_r}{z_x} \right] \cdot \max\left[0, V_b + D_r - C_b\right] \tag{41}$$

where k_{qb} is a drainage constant for the subzone. The water contents of the root zone and subzone are finally updated by subtracting the respective drainage:

$$V_r = V_r - D_r \tag{42}$$

$$V_b = V_b + D_r - D_b \tag{43}$$

3.7 Model element 7: Crop or forest growth

A crop has a life cycle that typically lasts less than a year. In Denmark, crops are sown in the fall or spring and harvested in the summer or fall. A crop's life cycle is mainly determined by the supply of solar energy during the growing season. The life cycle can thus be conceptualized as when certain energy supply thresholds are reached, the crop will begin to grow, reach its maximum of growth, or mature.

The supply of energy can for example be "measured" by growing degree-days, which is obtained by summing the daily temperatures since the day of sowing (Olesen and Heidmann, 2002):

$$S_s^d = \sum_{t_s}^{t_d} T_i \tag{44}$$

where t_s is the day of sowing, t_d is day *d* after sowing, and T_i is the mean temperature during day *i*. Edcrop uses the temperature sum (which is also called the growing degree-day) to govern plant growth as exemplified in the following.

Say spring barley is sown on day t_s. From the time of sowing, the seed requires a certain supply of energy from the atmosphere until it sprouts. This requirement is met on day t_o when the temperature sum S_s^d reaches a threshold value symbolized by S_o. On this day, the seed sprouts and the plant begins to grow. The roots and leaves will grow until reaching their maximum, which happens on the day that the temperature sum reaches another threshold value, S_f. From then on, the leaves grow no further. After some time the plant begins to mature on day t_r, when reaching the temperature threshold value S_r. Maturing lasts until day t_m, when reaching temperature threshold value S_m. During maturing the leaf area shrinks and gradually turns yellow (wilting). The plant finally disappears when harvested at day t_h.

The development and wilting of leaves for spring barley and other spring crops (like spring rape, pea, and potato) is modelled in Edcrop this way (Olesen and Heidmann, 2002):

$$L = \begin{cases} 0 & ;t < t_o \\ \min[L_m, L_m\left(\exp\left(2.4\left(S_s^d - S_o\right)/\left(S_f - S_o\right)\right) - 1\right)/10] & ;t_o \le t < t_r \\ L_m - \left(L_m - L_{ym}\right)\left(S_s^d - S_r\right)/\left(S_m - S_r\right) & ;t_r \le t < \min\left[t_m, t_h\right] \\ L_{ym} & ;t_m \le t < t_h \\ 0 & ;t_h \le t \end{cases} \tag{45}$$

where L_m is maximum leaf area, and L_{ym} is the yellow leaf area when the plant is fully matured.

Root growth for a spring crop is modelled as constant from the day of sprouting until it reaches its maximum depth. From then on root depth remains constant until the plant is either fully matured or harvested. Development of root depth is thus modelled by

$$z_r = \begin{cases} 0 & ;t < t_0 \\ \min\left[z_{\max}, c_r\left(t - t_0\right)\right] & ;t_0 \le t < \min\left[t_m, t_h\right] \\ 0 & ;\min\left[t_m, t_h\right] \le t \end{cases} \tag{46}$$

Figure 3 illustrates leaf development and root development as described by (45) and (46).

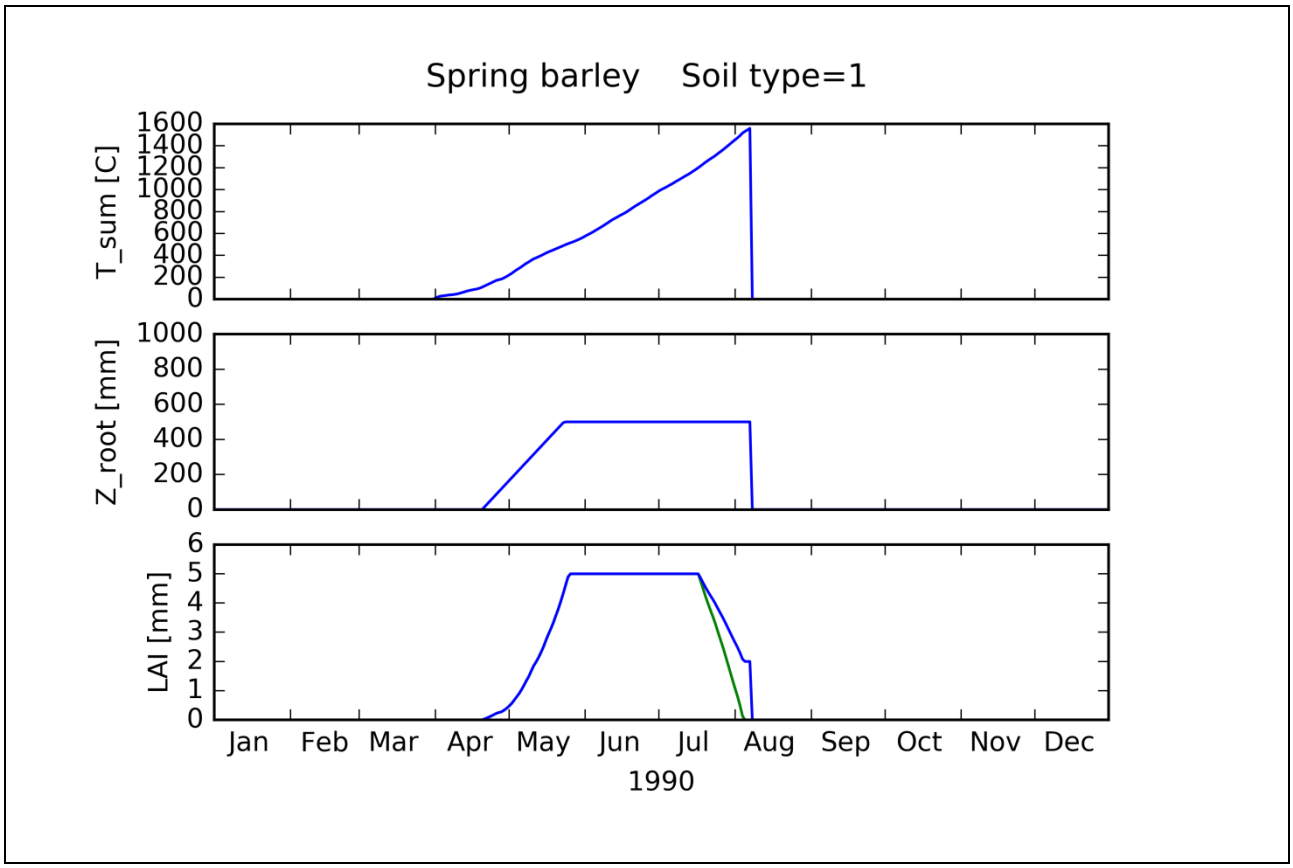

Figure 3 Temperature sum and development of root depth and leaf area index (LAI) for spring barely on coarse sandy soil. Green line indicates green LAI.

In Edcrop, the growth of other crops is modelled similarly; for example, Figure 4 shows growth of winter wheat in Denmark.

For crops, as default, for every year the time of harvest, t_h, is set to be at a fixed month and month-day. However, this default date can be combined with automatic setting: the automatic date is determined by Edcrop as being seven days later than the day when the green leaf area becomes less than 0.001. Harvest will then happen at the earlier of the default date and the automatic date.

It is noteworthy that this modelling, used in Edcrop, considers root growth and leaf development to be independent of water balance and plant transpiration. This is considered to work well for areas or times with sufficient rainfall and/or irrigation, but will work less well for areas or times with water shortage restricting plant growth.

According to Murray et al. (1989) and Foley et al. (1996), the annual cycle of leaf display in deciduous trees, and leaf activity in evergreen trees, also depends on degree-day requirements as well as on length of chilling period. However, the requirements and chilling period dependency varies considerably among tree species. Therefore, and because a forest usually consists of a blend of type of trees, Edcrop models growth of deciduous forest or needle tree forest as purely calendar dependent.

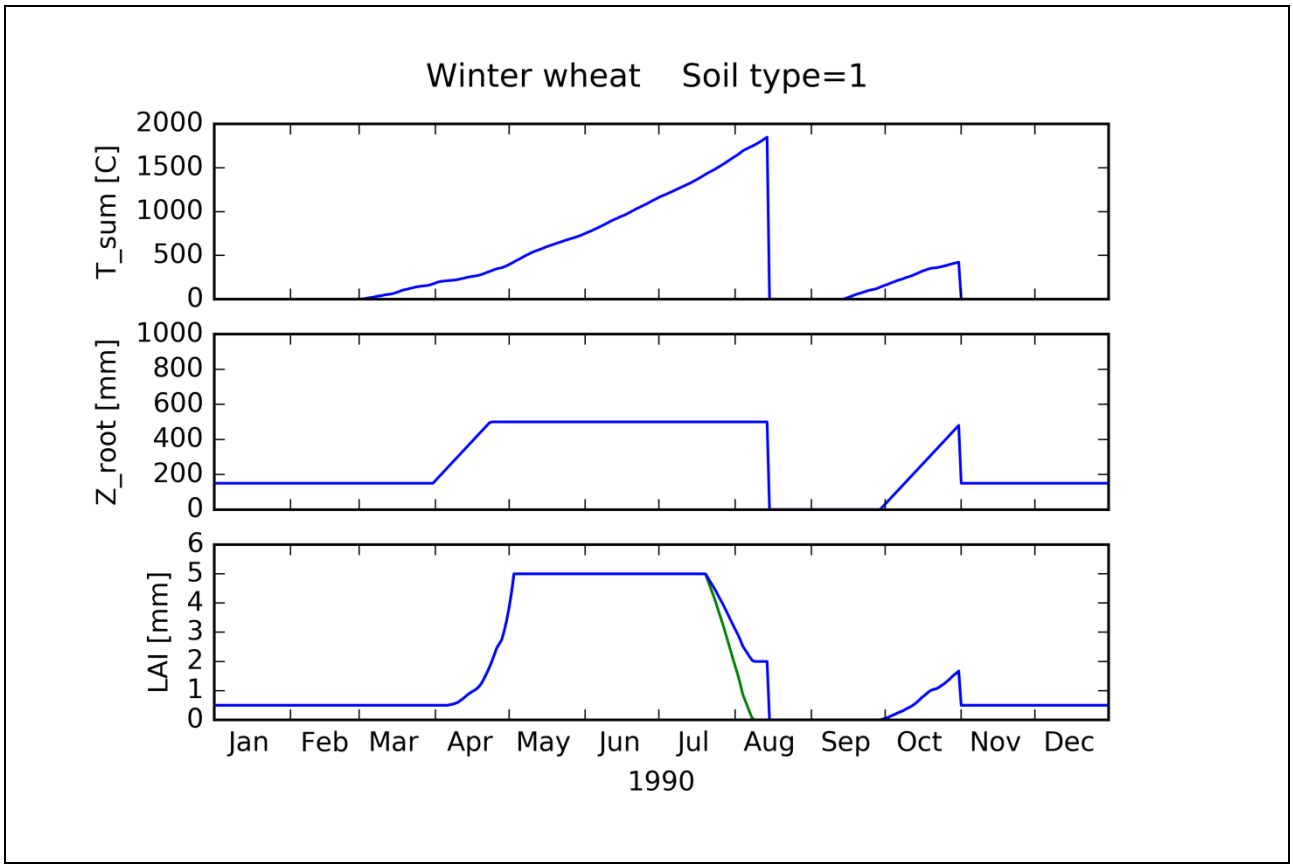

Figure 4 Temperature sum and development of root depth and leaf area index for winter wheat on coarse sandy soil. Green line indicates green LAI. Sowing of the crop happens in the fall, where it begins to grow. The plant goes in dormancy during winter where growth is set back. Growth begins again in the spring when the spring temperature sum exceeds a threshold. Harvest happens mid to late summer (here in August).

3.8 Modified models for wetland and wet meadow

In Edcrop, a wetland has a vegetation similar to permanent grass except the root zone permanently extends to full depth (equals z_{max}). Furthermore, the root zone is assumed to remain fully saturated year round. As a result, actual soil evaporation and actual evapotranspiration from the vegetation always equal their potential equivalents. The drainage rate from the root zone is computed as

$$D_r = P_l - E_a \tag{47}$$

Thus, drainage is modelled as instantaneous. It is positive when precipitation and snowmelt exceed actual evapotranspiration (which equals potential evapotranspiration). Conversely, it is negative when actual evapotranspiration is the higher value, where groundwater discharge, for example, compensates for the lack of precipitation. Full saturation of the root zone is simulated by maintaining

$$V_r = C_{r,sat} \tag{48}$$

and simulation of full saturation of the evaporation zone is maintained by

$$V_e = C_e \tag{49}$$

In (48), $C_{r,sat}$ is the water content within the entire soil profile when fully saturated.

In Edcrop, a wet meadow is similar to a wetland, except that the soil evaporation zone can dry out. When this occurs, actual soil evaporation becomes less than potential soil evaporation. The wet meadow is simulated like the wetland, but without applying equation (49).

3.9 Computation of potential evapotranspiration from reference evapotranspiration

As input, Edcrop uses a time series of reference evapotranspiration to compute potential evapotranspiration as

$$E_p = k_c \cdot E_{ref} \tag{50}$$

where k_c is a crop coefficient. The crop coefficient varies from crop to crop, depending on factors such as resistance to transpiration, crop height, roughness, reflection, and ground cover (Allen et al., 1998). Allen et al. (1998) give two methods of calculating the crop coefficient. Edcrop uses the simpler single coefficient method, where k_c varies with the development of the leaf area:

$$k_c = k_{c,\min} + \left(k_{c,\max} - k_{c,\min}\right) \cdot L_g / L_m \tag{51}$$

Here $k_{c,\min}$ and $k_{c,\max}$ are minimum and maximum values, respectively. For a sown and harvested crop, $k_{c,\min}$ should equal the evaporation coefficient for bare soil; for a permanent crop (e.g. grass) or forest, it should be the k_c winter value. Most of the default $k_{c,\max}$ values in Edcrop are from Allen et al. (1998). However, for forests the values are from Refsgaard et al. (2011).

3.10 Irrigation

In Edcrop, irrigation can be simulated either by specifying day and amount (forced irrigation), or through automatic irrigation.

For automatic simulation of irrigation, Edcrop uses a modified version of the procedure proposed by Aslyng and Hansen (1982). Following the modified procedure, the amount of irrigation on day i is:

$$I_i = \begin{cases} V_{irr} & ; V_{r,i-1} < c_{\lim} \cdot c_b \cdot C_{r,i-1} \wedge \sum_{j=i}^{i+2} P_j < P_{\lim} \wedge I_{count} > t_{freq} \wedge t_i < t_r - t_{\lim} \wedge t_i \in \text{irrigation season} \\ 0 & ; \text{Otherwise} \end{cases} \tag{52}$$

This means that irrigation $\left(I_i = V_{irr}\right)$ only happens when meeting all of five conditions. (i) The water content on the preceding day is less than a fraction of the root zone capacity; $c_{\lim}$ is the irrigation limit factor having a value between 0 and 1. (ii) The precipitation cumulated from day i and the following two days is less than $P_{\lim}$; this condition is based on the assumption that a farmer will know from the weather forecast when sufficient rain, $P_{\lim}$, can be expected within the next three days. (iii) The number of days since last

irrigation, I_{count}, must exceed t_{freq} days. (iv) Irrigation does not happen less than $t_{\lim}$ days from the end of the growing season, t_r (the day when maturing begins); this condition only works for maturing crops. Finally, (v) irrigation only happens within the irrigation season (which in Denmark would be May to July or August).

The amount of irrigation in (52) computes as

$$V_{irr} = \begin{cases} I_{\max} & ; C_{r,i-1} - V_{r,i-1} \geq I_{\max} \\ C_{r,i-1} - V_{r,i-1} & ; I_{\min} < C_{r,i-1} - V_{r,i-1} < I_{\max} \\ I_{\min} & ; C_{r,i-1} - V_{r,i-1} \leq I_{\min} \end{cases} \tag{53}$$

$V_{irr} = C_{r,i-1} - V_{r,i-1}$ means that water content in the root zone is restored to field capacity.

Aslyng and Hansen (1982) found that (52) and (53) give close to optimal crop production for Danish conditions with $c_{\lim} = 0.8$, $P_{\lim} = 5$ mm, $t_{\lim} = 20$ days, $I_{\min} = 20$ mm, and $I_{\max} = 50$ mm. However, common Danish practice is to rather use $I_{\min} = 25$ mm, $I_{\max} = 35$ mm, and $t_{freq} = 5$ days (Christian Thirup, Danish agronomist, personal communication).

In Edcrop, forced irrigation works for both crops, bare soil, and forests. Automatic irrigation only works for crops, not for bare soil and forests. Automatic and forced irrigation can be used simultaneously for crops. If automatic irrigation happens on the same day where forced irrigation is to take place, the automatic irrigation rate will overrule the forced rate.

3.11 Macro-pore drainage

The original Evacrop model by Olesen and Heidmann (2002) does not simulate macro pore drainage. However, Edcrop includes a modification that allows for it, using the following procedure.

Gravity drainage through macro pores, D_{mp}, occurs when the water content of the evaporation zone exceeds a threshold value:

$$D_{mp} = \begin{cases} \min\left(K_{mp}, V_e - C_{mp}\right) & ; V_e > C_{mp} \wedge V_{rel} < V_{mp-rel} \\ 0 & ; V_e \leq C_{mp} \vee V_{rel} \geq V_{mp-rel} \end{cases} \tag{54}$$

In (54), K_{mp} defines the maximum rate of macro pore drainage; the capacity C_{mp} defines the water content threshold at which macro-pore flow initiates; and V_e is the water content of the evaporation zone after evaporation, i.e. after (26). Further,

$$V_{rel} = \frac{V_r^* + V_b^*}{C_r + C_b} \tag{55}$$

is the relative water content of the entire soil profile prior to infiltration, and V_{mp-rel} is a relative water content threshold below which macro-pore drainage can occur. The water content threshold can be used to imitate the situation where macro pores vanish when the soil gets wet and forms when the soil gets dry. For example, if $V_{mp-rel} = 0.5$ then macro-pore drainage will only occur when the water content of the soil profile is less than fifty per cent of the soil profile's field capacity ($C_r + C_b$).

Macro-pore drainage is not simulated for wetlands or wet meadows; in these cases, (54) is substituted by $D_{mp} = 0.0$.

The simulation assumes that the macro pores are continuous through the soil profile, so the drainage occurs from the evaporation zone directly through the soil column. The sum of drainage from the soil profile is therefore

$$D_{sum} = D_b + D_{mp}, \tag{56}$$

After (54), the water content of the evaporation zone is reduced to

$$V_e = \min\left[C_e, V_e - D_{mp}\right] \tag{57}$$

and, instead of using (27) and (28), the water content of the root and sub zones are updated by:

$$V_r = \max\left[0, V_r^* - E_{ae} - D_{mp}\right] \tag{58}$$

$$V_b = \max\left[0, V_b^* - E_{ae} - D_{mp} + V_r^* - V_r\right] \tag{59}$$

Further, (33) and (34) are substituted by:

$$V_u = V_u + P_l - E_{ae} - D_{mp} \tag{60}$$

$$C_u = \min\left[C_r, C_u + \max\left[0, (P_l - E_{ae} - D_{mp}\right]\right] \tag{61}$$

4 Description of the alternative conceptualization for evapotranspiration and drainage

The Evacrop conceptualization separates the soil into only the root zone and the subzone. It uses daily time steps to compute evaporation, transpiration, and drainage, and employs a linear reservoir formulation to simulate the drainage. Edcrop also includes a modified conceptualization for soil evaporation, transpiration, and drainage, which routes water between four defined soil horizons. It uses either a Mualem – van Genuchten function or a linear model for the routing. To improve numerical accuracy, the simulations can use shorter than daily time steps to compute soil evaporation, transpiration, and drainage. The modified conceptualization can also simulate macro-pore flow and infiltration loss due to surface runoff. Simulation of all other elements of the water balance and of vegetation growth is as described in Chapter 3 and proceeds in daily time steps.

Each day can be subdivided into N_{step} equal-sized time steps. Similarly, in the following P_I, E_{pe}, and E_{pT} represents infiltration, potential evaporation, and potential transpiration, respectively, during the time step. The time step values are computed from their corresponding daily value by division with N_{step}. All other symbols shown in the following also apply for the actual time step.

The simulation for each time step proceeds as follows.

4.1 Evaporation from the soil

The water content of the evaporation zone is calculated as

$$V_e^* = V_e^0 + P_I \tag{62}$$

where V_e^0 is the water content of the evaporation zone at the end of the previous time step.

The actual evaporation from the soil is calculated as

$$E_{ae} = \begin{cases} E_{pe} & ; E_{pe} \le V_e^* \\ c_e E_{pe} & ; E_{pe} \le V_{soil}^* \\ 0 & ; E_{pe} > V_{soil}^* \end{cases} \tag{63}$$

where

$$V_{soil}^* = \sum_{i=1}^{nz} V_i^0 \tag{64}$$

and V_i^0 is the water content of layer (soil horizon) i at the beginning of the time step. This calculation is essentially the same as that used by the Evacrop (described in chapter 3.4).

Having calculated E_{ae}, the water content of the soil layers is updated as follows:

$$V_i^1 = \begin{cases} V_i^0 & ; E_{pe} > V_{soil} \\ V_i^0 - E_{ae} \times V_i^0 / V_{soil} & ; V_{soil} \ge E_{pe} > V_e^* \\ V_i^0 - E_{pe} & ; V_e^* \ge E_{pe} \wedge i = 1 \\ V_i^0 & ; V_e^* \ge E_{pe} \wedge i > 1 \end{cases} \tag{65}$$

This calculation is the same as for Evacrop except for the second condition, when $V_{soil} \geq E_{pe} > V_e^*$. In Evacrop, evaporation is taking as much evaporation as possible from the root zone, with the remainder from the subzone. However, in equation (65), the relative amount of evaporation taken from each layer depends on that layer's water content relative to the entire soil profile. In (65), V_i^1 symbolizes the water content in layer i after soil evaporation.

The water content of the evaporation zone is changed to

$$V_e = \begin{cases} V_e^* & ; E_{pe} > V_{soil} \\ V_e^* \left(1 - E_{ae} / V_{soil}\right) & ; V_{soil} \geq E_{pe} > V_e^* \\ V_e^* - E_{ae} & ; V_e^* \geq E_{pe} \end{cases} \tag{66}$$

4.2 Transpiration from vegetation

The water content of the root zone is calculated as:

$$V_r = \left(\sum_{i=1}^{I-1} V_i^1 \right) + V_I^1 \cdot \left(z_r - (I-1)\Delta z \right) / \Delta z \tag{67}$$

while the root zone capacity, C_r, is calculated as for Evacrop by (29).

The actual transpiration, E_{aT}, is also calculated as for Evacrop by (35) and (36).

The actual transpiration is taken from the root zone layers by

$$V_i^2 = \begin{cases} V_i^1 - E_{aT} \cdot V_i^1 / V_r & ; i < I \\ V_i^1 - E_{aT} \cdot \left(V_i^1 / V_r \right) \cdot \left(z_r - (I-1)\Delta z \right) / \Delta z & ; i = I \end{cases} \tag{68}$$

and from the evaporation zone by

$$V_e = \max\left[0, V_e \left(1 - \left(V_{i=1}^1 - V_{i=1}^2 \right) / V_{i=1}^1 \right) \right] \tag{69}$$

Now, V_i^2 symbolizes the water content of layer i after transpiration.

4.3 Drainage from soil layers

For crop or forest cover, the water content of a soil layer after infiltration, evaporation, transpiration, and drainage is computed as

$$V_i^3 = \begin{cases} V_i^2 - D_i & ; i = 1 \\ V_i^2 - D_i + D_{i-1} & ; i > 1 \end{cases} \tag{70}$$

where D_i is the drainage from layer i to the layer below (layer $i+1$), and V_i^3 symbolizes the water content of layer i after transpiration. For the alternative conceptualization, there are two alternatives to compute drainage.

The first alternative is to use a linear reservoir model, similar to that in Evacrop

$$D_i = \begin{cases} k_{qr} \cdot \left(V_i^2 - C_i \right) & ;\ V_i > C_i \\ 0 & ;\ V_i \le C_i \end{cases}, \tag{71}$$

where k_{qr} is a drainage constant, and C_i is the capacity of layer i to withhold water from gravity drainage, computed as

$$C_i = \theta_{Fi} \Delta z \tag{72}$$

In eqs. (71) and (72), Edcrop uses the same values for k_{qr} and θ_{Fi} as used in its Evacrop conceptualization (40) and (29), where the default values are taken from Olesen and Heidmann (2002).

The second alternative is to use the Mualem – van Genuchten hydraulic conductivity function (Schaap and van Genuchten, 2006) to compute drainage

$$D_i = \begin{cases} K_{s,i} \cdot \left[V_{i,r} \right]^{l_i} \cdot \left[1 - \left(1 - \left[V_{i,r} \right]^{1/m_i} \right)^{m_i} \right]^2 & ;\ V_{i,r} \le 1.0 \\ K_{s,i} & ;\ V_{i,r} > 1.0 \end{cases} \tag{73}$$

where subscript i refers to soil layer i, $V_{i,r} = V_i^2 / \theta_{s,i}$, $\theta_{s,i}$ is the water content at saturation, $K_{s,i}$ is the saturated hydraulic conductivity, l_i is a pore-connectivity parameter, $m_i = 1 - 1/n_i$, and n_i is a pore-size distribution parameter. The van Genuchten equation for soil water retention is (Schaap and van Genuchten, 2006)

$$\theta_h = \begin{cases} \theta_r + \dfrac{\theta_s - \theta_r}{\left[1 + \left| \alpha h \right|^n \right]^m} & ;\ h \le 0 \\ \theta_s & ;\ h > 0 \end{cases} \tag{74}$$

where h is pressure head, and θ_r is residual water content.

For eqs. (73) and (74), Edcrop contains default values for K_s, θ_s, θ_r, α, I, and n, taken from the Daisy version 5.93 input file "dk-horizon.dai" (Abrahamsen, 2020). Furthermore, Edcrop uses (74) to compute plant available water

$$\theta_{Fi} = \theta_{h=100\ \mathrm{cm}} - \theta_{h=16000\ \mathrm{cm}} = \theta_{pF2.0} - \theta_{pF4.2} \tag{75}$$

This is a standard method for estimating plant available water, which Madsen and Holst (1987) also used.

Finally, the drainage from the soil profile is

$$D_b = D_{i=nz}, \tag{76}$$

and the water content of the evaporation zone is adjusted to

$$V_e = \min\left[C_e, \max\left[0, V_e \left(1 - \left(V_{i=1}^2 - V_{i=1}^3 \right) / V_{i=1}^2 \right) \right] \right] \tag{77}$$

4.4 Macro-pore drainage

Gravity drainage, D_{mp}, through macro pores can occur when the water content of the top soil layer exceeds a threshold value:

$$D_{mp} = \begin{cases} \min\left(K_{mp}, V_1^3 - C_{mp}\right) & ;\ V_1^3 > C_{mp} \wedge V_{rel} < V_{mp-rel} \\ 0 & ;V_1^3 \le C_{mp} \vee\ V_{rel} \ge V_{mp-rel} \end{cases} \tag{78}$$

where K_{mp} defines the maximum rate of macro pore drainage, and the capacity C_{mp} is the water content threshold of the top soil layer at which macro-pore drainage initiates. Further,

$$V_{rel} = \frac{\sum_{i=1}^{nz} V_i}{C_r + C_b} \tag{79}$$

is the relative water content of the entire soil profile prior to macro-pore drainage, and V_{mp-rel} is a relative water content threshold below which macro-pore drainage can occur. This threshold can simulate the behavior of macro pores vanishing when the soil becomes wet and reforming as the soil dries. For example, if $V_{mp-rel} = 0.5$ then macro-pore drainage will only occur when the water content of the soil profile is less than fifty per cent of the soil profile's field capacity ($C_r + C_b$).

Macro pore drainage is not simulated for a wetland or a wet meadow. In these cases (78) is substituted by $D_{mp} = 0.0$.

The simulation assumes that macro pores are continuous throughout the soil profile, allowing drainage to occur from the top layer directly through the soil column. The sum of drainage from the soil profile is therefore

$$D_{sum} = D_b + D_{mp}, \tag{80}$$

The water content of the evaporation zone is reduced by

$$V_e = \min\left[C_e, \max\left[0, V_e\left(1 - D_{mp} / V_{i=1}^3\right)\right]\right] \tag{81}$$

before the water content of the top layer is reduced by

$$V_1^3 = V_1^3 - D_{mp} \tag{82}$$

After the simulation of macro pore drainage, Edcrop proceeds to compute the next time step.

4.5 Surface runoff

After soil and macro-pore drainage, loss of infiltration due to surface runoff, Q_{ro}, can be simulated to occur when the water content of the top soil layer is above saturation:

$$Q_{ro} = \begin{cases} 0 & ;\ V_1^3 \le C_{sat} \\ \min\left(K_{ro} \cdot \left(V_1^3 - C_{sat}\right), V_1^3 - C_{sat}\right) & ;\ V_1^3 > C_{sat} \end{cases} \tag{83}$$

where K_{ro} is the surface-runoff constant, and the capacity

$$C_{sat} = \left(\theta_{s,1} - \theta_{r,1}\right)\Delta z \tag{84}$$

defines the water content in the top soil layer at saturation. In (84), the saturation and residual water contents, $\theta_{s,1}$ and $\theta_{r,1}$, are for the top soil layer. The water content of the top layer is reduced by the amount of surface runoff

$$V_1^3 = V_1^3 - Q_{ro} \tag{85}$$

Surface runoff is not simulated for a wetland or a wet meadow. In these cases (83) is substituted by $Q_{ro} = 0.0$.

5 Input instructions

To use Edcrop, it is crucial to use consistent units for all input. Since all the default values set in Edcrop use millimeter for length and day for time, it is recommended to use these units to align with the default settings. If other units are used, all required data and parameters must be explicitly defined in the input to avoid the use of default values set in the code.

All input to Edcrop, except the climate time series, is given in a required text file named edcrop.yaml. The file is named with the extension .yaml because it follows the YAML format, a human-readable data serialization standard, and is processed using the Python YAML module. The edcrop.yaml file must be available in the working directory when Edcrop is run.

Chapter 5.1 describes the structure and content of edcrop.yaml. Chapter 5.2 briefly describes how Edcrop uses the edcrop.yaml input during execution. Chapters 5.3-5.6 go more into depth regarding the content of the edcrop.yaml file. Chapter 5.4 also describes the climate-time-series file.

5.1 Structure of input file edcrop.yaml

The edcrop.yaml file is a text file that consists of blank lines, comments, keys, key-value pairs, or value continuations that span multiple lines if needed.

A comment line begins with the hash character,"#".

A key is a text string which is succeeded by a colon, ":". The colon is not part of the key.

Values can take several forms, including the following:

- A text string – e.g. *JB1*.
- An integer number – e.g. *23*.
- A floating point number – e.g. *7.831* (with period as the decimal separator), or *3.52e-7* for an exponential number.
- A date in the format %Y-%m-%d – e.g. *1998-08-31*.
- A Boolean – i.e. either *True* or *False*.
- A list of either of the above – e.g. a list of three real numbers *[3.2, 7.32, 7.5e-3]*.
- A dictionary – e.g. a dictionary with two entries *{'key_1': 1.3, 'key_2': 2.7}*, where an entry has a key (a text string) and a value (here a real number, but it could be of any type).

A list begins with "[", and ends with "]". Entries are separated by commas.

A dictionary begins with "{", and ends with "}". A colon followed by a space separates the key and value, while commas separate entries.

The file information is further structured by line indentation. **It is a requirement to carry out line indentation by use of space characters, not by use of tabs!** Using tabs will create an error message and program failure

Indentation is used to define a block of information; indented lines contain information belonging to the block. Further indentation defines information belonging to a sub block.

Table 1 shows an example of an input block named “Models”. The block begins with the key “Models”, which define the block name. This line is succeeded by an indented line with the key “M1” defining a sub block of information to “Models”. The following line with key “M2” defines a new sub block to “Models”.

It is required to use exact same indentation for all sub blocks! Otherwise, the file loading fails with an error message! The error message is likely to point to the place in edcrop.yaml causing the loading problem.

In Table 1, the “M2” sub block contains a sub-sub-block with the key “wbfunc” and the value “evacrop”.

Table 1 Example of input block in edcrop.yaml file.

```
# This is a comment – beginning with the hash character

Models:
   M1:
   M2:
      wbfunc: evacrop
```

5.2 How Edcrop uses the edcrop.yaml input during execution

If Edcrop loads the “Models” block from Table 1, it will execute two model runs. First, Edcrop will run the model named 'M1.' Since there is no sub-block under 'M1', Edcrop will use the default setup and parameters; as default, Edcrop uses the conceptualization for evapotranspiration and drainage described in Chapter 4. Second, Edcrop will run the model named 'M2.' Since this sub-block contains the key 'wbfunc' with the value 'evacrop', Edcrop will use the Evacrop conceptualization for evapotranspiration and drainage, as described in Chapter 3.

Table 2 shows another example of an edcrop.yaml file containing three blocks, each with one or more sub blocks.

The first block is named “Climates”. Its sub block has a key named “C1”, and this has its own sub block with key “filename” and value (text string) “climate.dat”. This instructs Edcrop, when it executes, to read the required climate data from a file named “climate.dat”. “C1” would typically be a short name that identifies the climate station.

The second block is named “Soils”. It has two sub blocks named “JB1” and “JB7”, respectively. Neither of the sub blocks have its own sub block(s). This instructs Edcrop to execute sequential simulation for each of two default soil types named JB1 and JB2, respectively, without changing any soil parameter value from its default value. Soil descriptions are found below in Chapters 5.5 and 7.2.

The third block is named “Crops”, having two sub blocks named “SB” and “WW”, respectively. These sub blocks also do not have own sub blocks. This instructs Edcrop to execute simulation for two default crop types named SB (spring barley) and WW (winter wheat), respectively, without changing any crop parameter value from its default value. Crop descriptions are found below in Chapters 5.6 and 7.3.

The edcrop.yaml file in Table 2 thus instructs Edcrop to execute in total four simulations; one simulation for each combination of two soils (JB1 and JB7) and two crops (SB and WW).

Table 2 Example with the three mandatory input blocks of edcrop.yaml file.

```
# This is an edcrop.yaml file with the following three mandatory blocks

# Block defining file with climate data set
Climates:
   C1:
      filename: climate.dat

# Block defining soil types to be simulated
Soils:
   JB1:
   JB7:

# Block defining crops to be simulated
Crops:
   SB:
   WW:
```

If the edcrop.yaml file contained both the “Models” block in Table 1 and the three blocks in Table 2, Edcrop would execute in total eight simulations; a simulation for each combination of two model setups, two soils, and two crops.

It is mandatory that the edcrop.yaml file contains a block for “Climates”, “Soils”, and “Crops”, respectively, like in Table 2.

If the “Models” block is missing, Edcrop uses the default model setup when executing simulations.

In Edcrop, the edcrop.yaml file is loaded in the function named read_inp_file(…). This also stores the “Models”, “Climates”, “Soils”, and “Crops” information in respective Python dictionaries. These dictionaries are used by Edcrop to change settings and parameter values from their default as explained below.

The following sub-chapters give more information about the input regarding model setup, climate data, soils and crops. The sub-chapter titles and content follow the block names in the edcrop.yaml file, i.e. “Models”, “Climates”, “Soils”, and “Crops”.

5.3 “Models” input

The user should use the “Models” block of the edcrop.yaml file to specify model setup and model parameter values only to the extent they need to deviate from their default. If the “Models” block is not present, Edcrop uses the default model setup when executing simulations.

As illustrated in Table 1 and explained in Chapter 5.1, each sub block (indented block) of “Models” specifies a model setup that will be executed when running Edcrop with the respective edcrop.yaml input file. The key of the sub block gives the name of that model execution (e.g. M1 in Table 1), which is also used as model case short name in the naming of output files from this execution (explained in Chapter 6).

A sub block to a sub block (twice-indented block) specifies the desired setting or value by a key and a value, as for key “wbfunc” in Table 1. There can be several of such sub blocks to a sub block, if more than one setting or parameter needs to deviate from its default.

In Chapter 7.1, Table 6 lists all the model settings or model parameter values that can be changed by the edcrop.yaml “Models” input. The Table also gives the respective key and type of value to be specified in the edcrop.yaml file. Table 7 shows a second example of “Models” block input of an edcrop.yaml file.

In Edcrop, default model settings and parameter values are set in ModelParameters.initialize(…). Changing default settings or values by using the edcrop.yaml input happens in ModelParameters.read_initialize(…).

As default, Edcrop uses “wbfunc : ed”, the alternative water balance function described in chapter 4. To use the water balance function from Evacrop instead, set key and value “wbfunc: evacrop”.

5.4 “Climates” input

The mandatory 'Climates' block in the edcrop.yaml file specifies the file(s) from which to read the time series of climate data, including daily temperature, precipitation, and reference evapotranspiration. If more than one file is specified, Edcrop will execute simulation using each data set sequentially.

As illustrated in Table 3, each sub block (indented block) of “Climates” specifies a climate data input that will be used to execute a simulation when running Edcrop. The key of the sub block gives the climate case short name of that model execution (e.g. Clim_1 in Table 3), which is also used to name the output files from this execution (explained in Chapter 6). The short name typically identifies the corresponding climate station.

A sub block to the sub block (twice-indented block) MUST specify a set of climate data input by using the key “filename” with value being a string, where the string is a valid filename containing the climate data time series.

Another sub block may specify by key “dtformat” a value being a string defining the format of dates to be read from the climate data file. In Table 3, Clim_2 contains this sub block. This instructs Edcrop that dates in the climate file are in the format '%Y%m%d', which could for example be 20200128.

As for Clim_1 in Table 3, if a sub block with key “dtformat” is not specified, Edcrop will read the climate data file using the default format '%Y-%m-%d', which could for example be 2020-01-28.

The climate data file is a CSV file, which is read by Edcrop using pandas.read_csv(…).

Edcrop skips reading the first line of the file. This is because such a line often just names the input of the column beneath. Edcrop uses its own internal naming.

The file must contain four columns of daily data in this order: date, temperature, precipitation, and reference evapotranspiration.

During reading, the date column data are transformed by pandas.to_datetime(…) using the date format mentioned above.

Edcrop checks that the succeeding dates in the file increase by 1 day. If this is not the case, Edcrop will not use the file; instead, it writes a message to the screen and in the edcrop.log file, and continues with the next “Climates” input block.

The beginning and end dates of the climate data determine the simulation period.

In Edcrop, a climate data file is read by TimeSeries.read(…).

Table 3 Example of “Climates” input block of edcrop.yaml file.

```
Climates:
   Clim_1:
      filename: climate_station_1.dat
   Clim_2:
      filename: climate_station_2.dat
      dtformat: '%Y%m%d'
```

Table 4 Example of “Soils” input block of edcrop.yaml file. JB1a is a new soil type defined from one of the default soil types, JB1.

```
Soils:
   JB1:
   JB2:
      kqr: 0.2
   JB1a:
      soiltype: JB1
      name: Very coarse sandy
      thf: [.12, .06, .06, .06]
      kqr: 1.2
```

5.5 “Soils” input

The mandatory “Soils” block of the edcrop.yaml file specifies the soil types simulated during Edcrop execution.

Table 4 provides an example of a 'Soils' block, where three soil types – JB1, JB2, and JB1a – are specified using the first level of indented lines. JB1 and JB2 are two of seven predefined soil types in Edcrop (see Chapter 7.2), while JB1a is a new soil type defined for the actual simulation. The user is free to choose any name (key) for a new soil type. During execution, Edcrop uses JB1, JB2, or JB1a, respectively, as soil case short name in the naming of output files from that simulation (see Chapter 6).

Because there are no second level indented lines following the JB1-line in Table 4, the JB1 soil is simulated using entirely default parameter values set in Edcrop.

For the JB2 soil, a second-level indented line follows the JB2-line, having key “kqr” and value equal to 0.2. This instructs Edcrop to simulate the JB2 soil with the drainage rate k_{qr} set equal to 0.2; for all remaining soil parameters Edcrop uses default values.

A new soil type needs to be defined from one of the predefined soil types. In Table 4, a new soil type, short-named JB1a, is defined from the predefined JB1 soil, which is a coarse sandy type (see Chapter 7.2). The second-level indented line following the JB1a line, using “soiltype” as key and the string “JB1” as value, instructs this. The following second-level indented lines instruct Edcrop to change some soil parameter values from the default values of the JB1 soil.

As an alternative, a new soil type can be coded into Edcrop. This is done in SoilParameters.initialize(…). Thereby the soil type will be “predefined” in Edcrop.

In Chapter 7.2, Table 9 lists all the soil parameter values that can be changed by the edcrop.yaml “Soils” input. The Table also gives the respective key and type of value to be specified in the edcrop.yaml file.

Table 10 shows a second example of “Soils” block input of an edcrop.yaml file.

As default, Edcrop uses the linear drainage model for all soil types. However, if “wbfunc: ed” in the “Models” block, the nonlinear drainage model can be chosen instead by setting “soilmodel: mvg” as illustrated in Table 10 (Chapter 7.2). Also, as default Edcrop uses $K_{mp} = 0$, which means there is no macro pore drainage.

5.6 “Crops” input

The mandatory “Crops” block of the edcrop.yaml file specifies the vegetation types simulated during Edcrop execution. The “Crops” block is structured, and used by Edcrop, as described above for the “Soils” block.

Table 5 provides an example of a 'Crops' block, where three vegetation types –SB (spring barley), DF (deciduous forest), and WR (winter rape) – are specified using the first level of indented lines. SB and DF are two of thirteen predefined vegetation types in Edcrop (see Chapter 7.3), while WR is a new type defined for the actual simulation. (The user is free to choose nay name (key) for a new vegetation type.) For SB and DF, during execution they will have some of their parameter values or settings changed from the default. WR is defined from the predefined type, WW, which is winter wheat, but dates of sow and harvest are changed. During execution, Edcrop uses SB, DF, or WR, respectively, as vegetation case short name in the naming of output files from that simulation (see Chapter 6).

In Table 5, notice that the year is 1900 in the dates for sow and harvest of WR and for “leaflife” of DF. Edcrop does not use the year; it only uses month and day for every simulated year. Therefore, an arbitrary year can be used for the value of these dates.

For permanent use, a new vegetation type can be coded into Edcrop and thereby be “predefined”. This is done in CropParameters.initialize(…).

In Chapter 7.3, Table 12 lists all the vegetation parameter values that can be changed by the edcrop.yaml “Crops” input. The Table also gives the respective key and type of value to be specified in the edcrop.yaml file.

In Edcrop, default vegetation parameter values and growth models are set in CropParameters.initialize(…). Changing default settings or values by using the edcrop.yaml input happens in CropParameters.read_initialize(…).

Table 5 Example of “Crops” input block of edcrop.yaml file. SB and DF are predefined vegetation types for which Edcrop during execution changes some parameter values from the default. WR is a new crop type defined from one of the predefined types, WW.

```
Crops:
   SB:
      cb: [0.1, 0.1, 0.2, 0.4, 0.5, 0.5, 0.5, 0.5, 0.4, 0.3, 0.2, 0.1]
      cr: 12.
      autoharvest: true
   DF:
      leaflife: [1900-04-15, 1900-06-01, 1900-09-15, 1900-10-15]
   WR:
      croptype: WW
      name: Winter rape
      sowdate: 1900-09-01
      harvestdate: 1900-07-20
```

5.7 Winter season, irrigation season, and use for southern hemisphere conditions

Edcrop assumes there is no plant growth during winter. By default, winter begins at the end of October and ends at the end of February. The beginning and end of winter can be changed in the “Models” input using the key “winterperiod” (see Chapter 7.1).

Similarly, the irrigation season can be specified by using the “Models” input key “irrigationperiod” (see Chapter 7.1). By default, Edcrop uses end of April as the beginning, and the end of August as the end of the season.

To use Edcrop for Southern Hemisphere conditions, it is obviously necessary to change the winter and irrigation seasons from their default. It will also be necessary to change the dates for the sowing and harvesting of crops as well as the ‘leaflife’ of deciduous forest (see Chapter 7.3). For more permanent use of Edcrop for southern hemisphere conditions, it will be more expediently to change these default values inside the Edcrop code: for default winter and irrigation periods, they should be changed in ModelParameters.initialize; for the crop and forest related parameters, they should be changed in CropParameters.initialize.

6 Output files

By default, Edcrop generates a log file and two output files for each simulation, containing daily and yearly water balance results. The user is free to choose which results are printed. Users can also choose to plot some input data and simulation results and save them as PNG files.

6.1 The log file – edcrop.log

Edcrop always generates a log file named edcrop.log. The log file summarizes the input of all the simulations that Edcrop loops through during execution, and it lists all the error messages and warnings that Edcrop sends.

6.2 Print files

For each simulation, daily and yearly water balance results are saved in two files; one with daily results, the other with yearly results. The name of each file consists of five parts separated by underscore,"_":

%1_%2_%3_%4_%5

The first part (%1) represents the climate case short name, the second part (%2) is the soil case short name, the third part (%3) is the vegetation case short name, and the fourth part (%4) is the model case short name. The fifth part (%5) is '_wb.out' for daily output or '_y_wb.out' for yearly output. The short names are taken from the edcrop.yaml input file as explained above in Chapters 5.3 to 5.6. Hereby each filename becomes unique, and a file with results from a specific simulation is identifiable from the filename.

The variables that can be printed are listed and explained in Table 13. By default, the printed variables are

T P Ep I Ea Dsum

For daily output, the date is also printed; for yearly output, only the year is also printed.

It is easy to change the list of output variables by, in the right place of the "Models" block of the edcrop.yaml file, using the key "prlistd" for the daily output, or "prlisty" for the yearly output. For example, putting this line in the right place of the "Models" block

prlistd: P Ea Dsum

instructs Edcrop to only print daily precipitation, actual evapotranspiration, and drainage from the sub zone. During reading of the value list, only names recognized from Table 13 in Chapter 7.4 as valid variables will be used to print output. If there is no recognition of valid variables from this list, or if 'Date' is the only valid variable in the list, no daily output is printed.

The same rules apply for the key "prlisty" and the yearly output. For yearly output, 'Date' is not valid.

For the daily output, there is an alternative to setting the output variables by use of key "prlistd". The alternative is to use the key named "iprnd" in the "Models" block. This key can take any of four integer values: 1, 2, 3, or 4. For increasing value, more variables will be printed: for "iprnd : 1", only the default variables are printed (in which case there is no need to set "iprnd"); for "iprnd : 4", all variables will be printed.

6.3 Plot file

In the 'Models' block of the edcrop.yaml file, the user can set

plotseries: true

to generate two PNG files. These files will contain plots of certain input and simulated variables. The first file plots precipitation (or sum of precipitation and irrigation), actual evapotranspiration, and drainage from the subzone (named P, Ea, and Db in Table 13, respectively). The other file plots temperature, root depth, leaf area, and crop coefficient (named T, zr, L, and kc in Table 13, respectively).

The name of the two PNG files has five parts separated by underscore,"_":

%1_%2_%3_%4_%5

The first part, *%1*, is the climate case short name; *%2* is the soil case short name; *%3* is the vegetation case short name; *%4* is the model case short name; and *%5* is "_P_Ea_Db.png" and "_T_zr_L_kc.png" for the two files, respectively.

Figure 5 shows an example of plots from Edcrop.

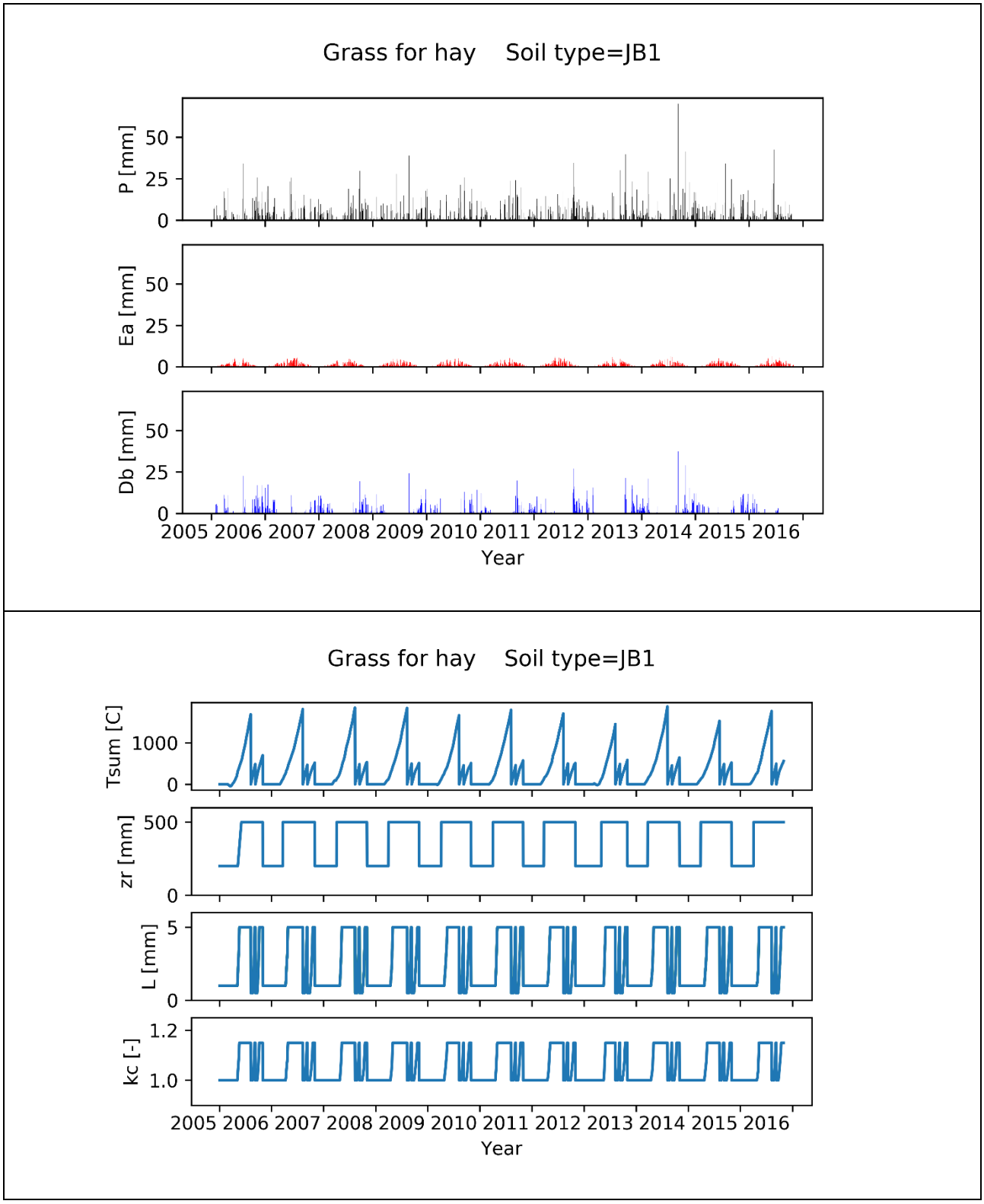

Figure 5 Plots produced by Edcrop when *plotseries* = yes.

7 Tables of input and output

7.1 Model parameters

The following table supplements the model parameter input description in chapter 5.3.

Table 6 Model parameters for which 'value' can be specified by use of 'key' in the "Models" block of the edcrop.yaml input file. Note that 'key' is a case sensitive name. FPN is short for 'floating point number'.

'key'	Model parameter c.f. Ch. 1	Description
wbfunc		A string specifying which water balance function to use. 'value' must be *ed* or *evacrop*; default is *ed*.
stepsperday		Defines the number of time steps per day when wbfunc = ed. The value must be an integer greater than 0.
Cr	C_r	Initial capacity of root zone to hold water. *FPN*.
Cb	C_b	Initial capacity of subzone to hold water. *FPN*.
Cu	C_u	Initial capacity of upper root zone. *FPN*.
Vs	V_s	Initial water content of snow reservoir. *FPN*.
Vr	V_r	Initial water content of root zone. *FPN*.
Vb	V_b	Initial water content of subzone. *FPN*.
Ve	V_e	Initial water content of evaporation zone. *FPN*.
Vu	V_u	Initial water content of upper root zone. *FPN*.
Vl	V_l	Initial content of intercepted water. *FPN*.
ci	c_i	Interception capacity constant. *FPN*.
cm	c_m	Degree day factor for snow melt. *FPN*.
Tm	T_m	Threshold temperature for snow melt. *FPN*.
ce	c_e	Evaporation factor for dry soil. *FPN*.
kp	k_p	Extinction coefficient. *FPN*.
zmax	$z_{\max}$	Depth of simulated soil profile. *FPN*.
winterperiod		A list of two dates defining the beginning and end of winter, respectively. For format, see [1)] and [2)].
irrigationperiod		A list of two dates defining the beginning and end of irrigation season, respectively. For format, see [1)] and [2)].
irrigationdate		A list containing dates of forced irrigation. See [1)] and [2)].
irrigation		The rate for forced irrigation. *FPN*.
autoirrigate		A boolean specifying whether to simulate irrigation automatically. 'value' must be *false* or *true*; default is *false*.
tlim	t_{lim}	The irrigation time limit in days with respect to maturing. An integer value.
tfreq	t_{freq}	Minimum number of days between automatic irrigation. 'value' must be positive integer.

clim	c_{lim}	Factor limiting water content requiring automatic irrigation. *FPN*.
Plim	P_{lim}	Precipitation limit for automatic irrigation. *FPN*.
Imin	I_{min}	Minimum amount of automatic irrigation. *FPN*.
Imax	I_{max}	Maximum amount of automatic irrigation. *FPN*.
iprnd		'value' must be integer 1, 2, 3, or 4; default is 1. Specifies predefined list of time series variables for writing daily and yearly outputs to print files. The higher number, the more variables are output.
prlistd		Text string of time series variables for writing of daily output to print file. In the string, a space character must separate two variable names.
prlisty		Text string of time series variables for writing of yearly output to print file. In the string, a space character must separate two variable names.
plotseries		Specifies whether to make plots of simulation output. 'value' must be *yes* or *no*; default is *no*.

[1)] A date is given in the format %Y-%m-%d; an example using this format is 2019-12-31, where first four digits give year (2019), next two digits give month (12), and last two digits give day (31).
[2)] Only day and month is used, every simulated year.

Table 7 A second example of "Models" block input in edcrop.yaml file. Two model runs, named M1 and M2, will be executed.

```
Models:
 M1:
  wbfunc: evacrop
  winterperiod: [1900-11-02, 1901-03-02]
  irrigationperiod: [1900-04-02, 1900-09-03]
  irrigationdate: [1900-07-07, 1900-07-17, 1900-08-07,
                   1900-08-17]
  irrigation: 16.
  iprnd: 2
  plotseries: true

# M2 implicitly uses the default, which is wbfunc: ed
 M2:
  stepsperday: 4
  autoirrigate: true
  tlim: 20
  tfreq: 5
  clim: .9
  Plim: 7.
  Imin: 25.
  Imax: 35.
  prlistd: Ep Ea Db
```

7.2 Soil types and parameters

The following tables supplement the soil input description in chapter 5.5.

Table 8 Predefined soil types, with oral and quantitative characterization. Valid 'key' type names are JB1 to JB7. Notice that 'key' is a case sensitive name. These soil types are based on Olesen and Heidmann (2002), who used the classification system developed by Madsen and Holst (1987).

Soil type 'key'	Soil description	Weight %			
		Clay < 2 µm	Silt 2-20 µm	Fine sand 20-200 µm	Total sand 20-2000 µm
JB1	Coarse sandy	0-5	0-20	0-50	75-100
JB2	Fine sandy			50-100	
JB3	Coarse sandy with clay	5-10	0-25	0-40	65-95
JB4	Fine sandy with mix of clay			40-95	
JB5	Clayey with coarse sand	10-15	0-30	0-40	55-90
JB6	Clayey with fine sand			40-90	
JB7	Clayey	15-25	0-35		40-85

Table 9 Soil parameters for which 'value' can be specified by use of 'key' in the "Soils" block of the edcrop.yaml input file. Notice that 'key' is a case sensitive name. FPN is short for 'floating point number'.

'key'	Model parameter c.f. Ch. 1	Description
soiltype		A predefined soil type 'key' chosen from Table 8. A *string*.
name		A descriptive name for the soil. A *string*.
thf	θ_F	Plant available water content of the soil. List of four *FPN*'s,; one value for each of four 25 cm depth intervals. (Total soil depth is 100 cm – unless the model parameter value $z_{\max}$ is changed.)
Ce	C_e	Capacity of evaporation zone. *FPN*.
kqr	k_{qr}	Drainage constant for root zone. *FPN*.
kqb	k_{qb}	Drainage constant for sub zone. *FPN*.
Kmp	K_{mp}	Maximum rate of macro-pore drainage. *FPN*. Only used when wbfunc = *ed* (see Table 6).
Cmp	C_{mp}	Water content threshold at which macro-pore drainage is initiated. *FPN*.
Vmprel	V_{mp-rel}	Relative water content threshold below which macro-pore drainage can occur. *FPN*.
Kro	K_{ro}	Maximum rate of surface runoff. *FPN*. Only used when wbfunc = *ed* (see Table 6).
thsat	$\theta_{s,1} - \theta_{r,1}$	Water content at which macro-pore flow is initiated. *FPN*. Only used when both wbfunc = *ed* (see Table 6) and soilmodel = *lin*. For soilmodel = *mvg*, thsat is calculated internally from water content values given for the top soil horizon.
horizon	name θ_s θ_r α n K_s l	Definition of soil horizon, which is a dictionary with key being name given to horizon (*string*) and value being a list of six Mualem – van Genuchten parameter values (*FPN*). Only used when both wbfunc = *ed* (see Table 6) and soilmodel = *mvg*. Definition of default horizons is in Edcrop's SoilParameters.horizon dictionary.
soilhorizons		Name of four horizons making the soil profile. List of four *strings*, where each string is the name of a soil horizon Each horizon must either be given as input or be found in Edcrop's SoilParameters.horizon dictionary. Definition of default soil horizons for the soil types in Table 8 is found in Edcrop's SoilParameters.MvG_soilhorizons dictionary.
soilmodel		A string specifying which soil drainage model to use, linear or Mualem – van Genuchten. 'value' must be *lin* or *mvg*; default is *lin*. Only used when wbfunc = *ed* (see Table 6).

Table 10 A second example of “Soils” block input in edcrop.yaml file. JB1 is a predefined soil to be simulated using the Mualem- van Genuchten drainage model. JB10 is a new soil type with two horizons changed from the default of the JB1 soil.

```
Soils:
   JB1:
      soilmodel: mvg
   JB10:
      soiltype: JB1
      # The following defines two new horizons
      horizon: {B_JB10: [0.3, 0.0, 0.06, 1.445, 800., -0.3],
                C_JB10: [0.25, 0.0, 0.06, 1.6, 1000., 1.3]}
      # The following defines the horizons of the JB10 soil
      soilhorizons: [A_JB1, B_JB1, B_JB10, C_JB10]
      soilmodel: mvg
```

7.3 Vegetation types and parameters

The following tables supplement the crop input description in chapter 5.6.

Table 11 Predefined crop or vegetation types, with characterization. Notice that ‘key’ is a case sensitive name. The first ten types are taken from Olesen and Heidmann (2002).

Type ‘key’	Crop or vegetation description (the ‘name’)
BS	Bare soil
G1	Grass with grazing
G2	Grass for hay
WW	Winter wheat
SB	Spring barley
POT	Potato
FB	Fodder beet
SR	Spring rape
PEA	Pea
SBG	Spring barley with grass
MZ	Maize
DF	Deciduous forest
SF	Spruce forest
WL	Wetland
WM	Wet meadow

Table 12 Crop or vegetation parameters for which 'value' can be specified by use of 'key' in the "Crops" block of the edcrop.yaml input file. Notice that 'key' is a case sensitive name. FPN is short for 'floating point number'.

'key'	Model parameter c.f. Ch. 1	Description
name		A descriptive name for the crop. A string.
cb	c_b	Bend point for relative transpiration function. List of twelve *FPN*'s, one for each month.
cr	c_r	Root growth velocity during spring season. *FPN*.
cre	c_{re}	Root growth velocity during fall season. *FPN*.
zrv	z_{rv}	Root depth during winter season (for winter crop). *FPN*.
zrx	z_{max}	Maximum root depth. *FPN*.
Lm	L_m	Maximum leaf area. *FPN*.
Lv	L_v	Leaf area during winter (for winter crop). *FPN*.
Lc	L_c	Leaf area after cutting (for grass). *FPN*.
Lym	L_{ym}	Yellow leaf area at maturity. *FPN*.
So	S_o	Temperature sum for sprouting. *FPN*.
Sf	S_f	Temperature sum for full leaf area. *FPN*.
Sr	S_r	Temperature sum for beginning of maturing. *FPN*.
Sm	S_m	Temperature sum for end of maturing. *FPN*.
Soe	S_{oe}	Temperature sum for sprouting during fall (for winter crop). *FPN*.
Sfe	S_{fe}	Temperature sum for full leaf area during fall (for winter crop). *FPN*.
kcmin	$k_{c,\min}$	Minimum crop coefficient. *FPN*.
kcmax	$k_{c,\max}$	Maximum crop coefficient. *FPN*.
sowdate		Date of sowing. Give date as explained in [1), 2)]. Not used for grass crops since they are permanent.
harvestdate		Date of harvesting. Give date as explained in [1), 2)]. For crop G1, grass for grazing, this is not used since it is a permanent crop. For crop G2, grass for hay, this must be a list containing dates of cutting. For crop SBG, spring barley with grass, this must be a list with date for barley harvest, followed by date(s) of grass cutting.
autoharvest		Only for crops: a boolean specifying whether to simulate harvest automatically. Harvest will occur 7 days after Lg < 0.001. 'value' must be *trues* or *false*; default is *false*.
leaflife		Only for deciduous forest: list of four dates for leaf_out_begin, leaf_out_end, leaf_loss_begin, and leaf_loss_end, respectively. Give each date as explained in [1), 2)].

[1)] A date is given in the format %Y-%m-%d; an example using this format is 2019-12-31, where first four digits give year (2019), next two digits give month (12), and last two digits give day (31).

[2)] Only day and month are used, every simulated year.

7.4 Possible output variables

The following table supplements the output description in chapter 6.2.

Table 13 Possible variables that can be daily or yearly output in respective print files. Notice that 'key' is a case sensitive name.

'key'	Model variable c.f. Ch. 1	Description
Date		Date.
T	T	Temperature.
P	P	Precipitation, total.
Pr	P_r	Precipitation falling as rain.
Ps	P_s	Precipitation falling as snow.
Pm	P_m	Water made available by snow melt.
Er	E_{ref}	Reference evapotranspiration.
Ep	E_p	Potential evapotranspiration.
Ea	E_a	Actual evapotranspiration.
Eas	E_{as}	Actual evaporation from snow reservoir.
Eae	E_{ae}	Actual evaporation from soil.
Eai	E_{al}	Actual evaporation of intercepted water.
Eaig	E_{alg}	Actual evaporation of water intercepted on green leaves.
Eaiy	E_{aly}	Actual evaporation of water intercepted on yellow leaves.
Eat	E_{aT}	Actual transpiration.
Ept	E_{pT}	Potential transpiration.
Epe	E_{pe}	Potential evaporation not attenuated by vegetation.
Epc	E_{pc}	Potential evaporation attenuated by vegetation.
Epcg	E_{pcg}	Potential evaporation attenuated by green vegetation.
Epcy	E_{pcy}	Potential evaporation attenuated by yellow vegetation.
Dr	D_r	Drainage from root zone to sub zone.
Db	D_b	Drainage from sub zone.
Dmp	D_{mp}	Drainage from macro pores.
Dsum	$D_{sum} = D_b + D_{mp}$	Sum of drainage from subzone and macro pores.
Qro	Q_{ro}	Surface runoff.
I		Irrigation.
Tsum	S_s	Temperature sum driving plant growth.
L	L	Leaf area
Lg	L_g	Green leaf area.
Ly	L_y	Yellow leaf area.
zr	z_r	Root depth
kc	k_c	Crop coefficient.
Vsum		Sum of stored water (equal to $V_s + V_l + V_{soil}$).
Vdel		Change in stored water.
Vs	V_s	Water stored in snow reservoir.

Vi	V_I	Water stored by interception.
Ve	V_e	Water stored in evaporation zone.
Vu	V_u	Water stored in upper root zone.
Vr	V_r	Water stored in root zone.
Vb	V_b	Water stored in sub zone.
Vsoil	V_{soil}	Water stored in soil profile (equal to $V_r + V_b$) .
Cu	C_u	Capacity of upper root zone to store water.
Cr	C_r	Capacity of root zone to store water.
Cb	C_b	Capacity of sub zone to store water.

8 References

Abrahamsen, P. (2020). Daisy Program Reference Manual. Department of Plant and Environmental Sciences, University of Copenhagen, February 3, 2020. 624 pp. https://daisy.ku.dk/.

Ahuja, L.R., Rojas, K.W., Hanson, J.D., Shaffer, M.J., and Ma, L. (2000): Root Zone Water Quality Model – Modeling management effects on water quality and crop production. Water Resources Publications, LLC. Colorado, USA, 372 pp.

Allen, R.G., Pereira, L.S., Raes, D., and Smith, M. (1998). Crop Evapotranspiration—Guidelines for Computing Crop Water Requirements. FAO Irrigation and drainage paper 56. Rome, Italy: Food and Agriculture Organization of the United Nations. ISBN 92-5-104219-5. http://www.fao.org/3/X0490E/x0490e00.htm.

Aslyng, H.C. & Hansen, S. (1982). Water Balance and Crop Production Simulation. Hydrotechnical Laboratory. The Royal Veterinary and Agricultural University. Copenhagen. 200 pp.

Doherty, J. (2010). PEST – Model-Independent Parameter Estimation, User Manual, 5th Edition. Watermark Numerical Computing. 336 pp. https://pesthomepage.org/documentation.

Foley, J.A., Prentice, I.C., Ramankutty, N., Levis, S., Pollard, D., Sitch, S., and Haxeltime, A. (1996). An integrated biosphere model of land surface processes, terrestrial carbon balance, and vegetation dynamics. Global Biogeochemical Cycles, 10(4), p. 603-628.

Hansen, S., Jensen, H.E., Nielsen, N.E., and Svendsen, H. (1990). DAISY – Soil Plant Atmosphere System Model. NPo-forskning fra Miljøstyrelsen, Nr. A10. Miljøstyrelsen, København. 272 pp.

Healey, R.W. (2010). Estimating groundwater recharge. Cambridge University Press, 245 pages.

Madsen, H.B., and Holst, K. (1987). Potentielle marginaljorder - Landsdækkende kortlægning af jordbundsfysiske og kemiske forhold, der har indflydelse på jordens dyrkning. Marginaljorder og miljøinteresser. Miljøministeriets projektundersøgelser 1986, Teknikerrapport nr. 1.

McDonald, M.G., and Harbaugh, A.W. (1988). A Modular Three-Dimensional Finite-Difference Ground-Water Flow Model. U.S. Geological Survey, Techniques of Water-Resources Investigations, Book 6, Chapter A1.

Murray, M.B., Cannell, M.G.R., and Smith, R.I. (1989). Date of budburst of fifteen tree species in Britain following climatic warming. Journal of Applied Ecology, 26, p. 603-700.

Olesen, J.E., and Heidmann, T. (2002). EVACROP – Et program til beregning af aktuel fordampning og afstrømning fra rodzonen, Version 1.01. Afdeling for Plantevækst og Jord og Afdeling for Jordbrugssystemer, Forskningscenter Foulum, Tjele, Danmark.

Refsgaard, J.C., Stisen, S., Højbjerg, A.L., Olsen, M., Henriksen, H.J., Børgesen, C.D., Vejen, F., Kern-Hansen, C., and Blicher-Mathiesen, G. (2011). Vandbalance i Danmark – Vejledning i opgørelse af vandbalance ud fra hydrologiske data for perioden 1990-2010. Danmarks og Grønlands Geologiske Undersøgelse, Rapport 2011/77.

Schaap, M.G., and van Genuchten, M.T. (2006). A Modified Mualem–van Genuchten Formulation for Improved Description of the Hydraulic Conductivity Near Saturation. Vadose Zone Journal 5, p. 27–34, doi:10.2136/vzj2005.0005.

Stisen, S., Sonnenborg, T.O., Højbjerg, A.L., Troldborg, L., Refsgaard, J.C. (2011): Evaluation of Climate Input Biases and Water Balance Issues Using a Coupled Surface–Subsurface Model. Vadose Zone Journal, 10, p. 37–53, doi:10.2136/vzj2010.0001.

Appendix A: Comparison of Edcrop results

This appendix compares results obtained by using the two water balance functions of Edcrop with alternative settings. The comparisons use a base example with spring barley (SB) growing on a JB1 (sandy) soil. Winter wheat (WW) and clayey (JB7) soil were also simulated to supplement the base example. The climate data cover the ten years 1990-1999 measured at Taastrup, Denmark (in file dk-taastrup.dwf, downloaded with the Daisy version 5.93 code from https://daisy.ku.dk/download/windows/).

The compared simulation results were obtained using either of four water balance function setups:

(i) *wbfunc: evacrop*;
(ii) *wbfunc: ed*; with linear soil drainage model, using one time step per day of simulation;
(iii) *wbfunc: ed*; with linear soil drainage model, using six time steps per day of simulation;
(iv) *wbfunc: ed*; with Mualem – van Genuchten (MvG) soil drainage model, using six time steps per day of simulation; and
(v) *wbfunc: ed*; with linear soil drainage model, and macropore drainage.

For a given combination of soil type and crop type, simulation of potential evapotranspiration, as well as simulation of plant growth, are identical for cases (i) to (v).

A.1. Case (i) versus case (ii) – "evacrop" versus "ed"

To directly compare the 'evacrop' and 'ed' water balance functions, the 'ed' simulations used a daily time step and a linear soil drainage model, similar to the 'evacrop' simulations. The soil parameters used in 'ed' and 'evacrop' were identical in this comparison.

Figure A.1 shows a comparison of simulated actual evapotranspiration (E_a) for spring barley (SB) growing on coarse sandy (JB1) soil. It is noticed that E_a simulated by "ed" tends to exceed the simulation by "evacrop", and the tendency increases with E_a. Figure A.2 shows the opposite tendency for drainage from the soil column (D_b). For daily drainage, it is noticed that "evacrop" simulates high drainage during several days whereas "ed" does not. Figure A.3 shows that "ed" simulates smoother drainage with lower peaks than "evacrop". The smoother drainage and lower peaks occur because 'ed' simulates transpiration and drainage using four equally thick soil horizons, while 'evacrop' uses only two zones: the root zone and the subzone. Later, it is shown that incorporating macro-pore drainage in 'ed' makes the total drainage simulation more similar to that of 'evacrop.

The smoothing and lower peak tendencies in simulation of E_a and D_b become even clearer for spring barley grown on clayey soil (JB7), and there is more scatter in the results. This is illustrated for D_b in Figure A.4. It is also observed that on several summer days, 'evacrop' simulates drainage, whereas 'ed' does not. Similar results were obtained for winter wheat growing on JB1 and JB7 soils.

The comparison shows that, other things equal, "evacrop" simulates faster and peakier flow through the soil than "ed" (unless, as shown later, the latter simulation includes macro pore drainage), and it simulates less actual evapotranspiration and larger drainage than "ed".

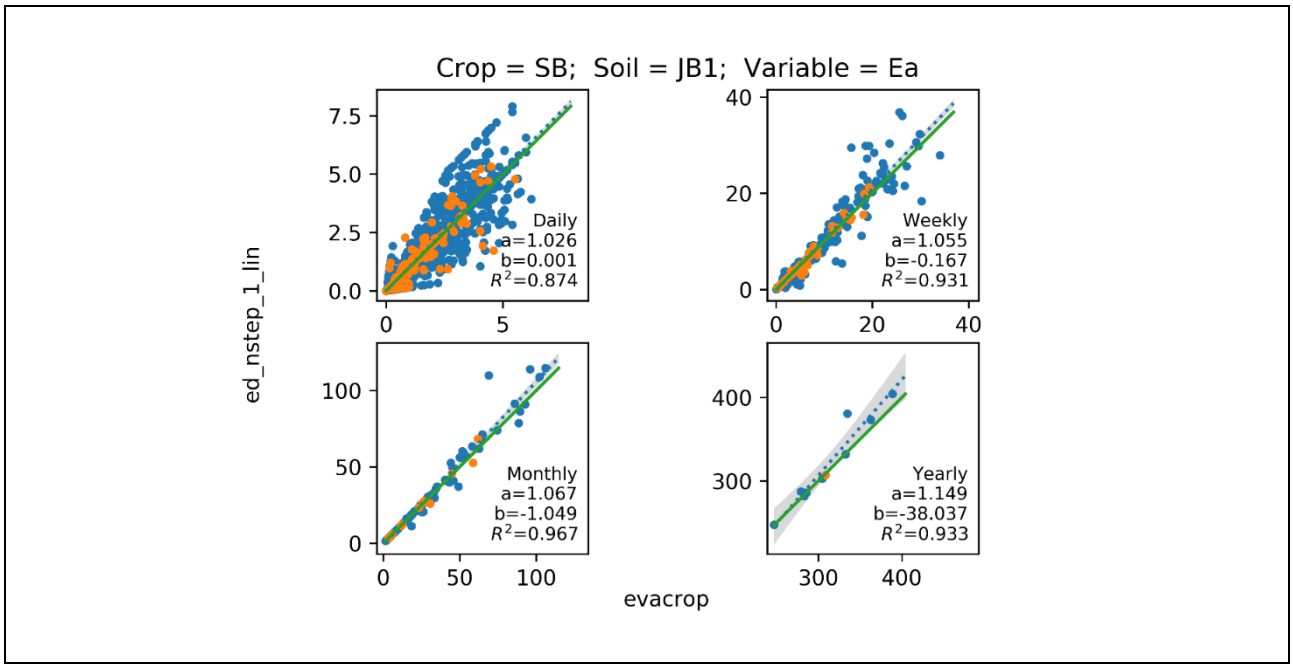

Figure A.1 Actual evapotranspiration for "evacrop" (case i) versus "ed" (case ii) for the period 1990-1999. Orange points are for 1990, blue points for the remaining years. Blue dashed line is line fitted to the blue points, and the grey shaded area is the 95% confidence band for the fitted line. The green line is the identity (1:1) line. The text says whether it is daily, weekly, monthly, or yearly results; a is the slope, b the intercept, and R^2 the coefficient of determination of the fitted line.

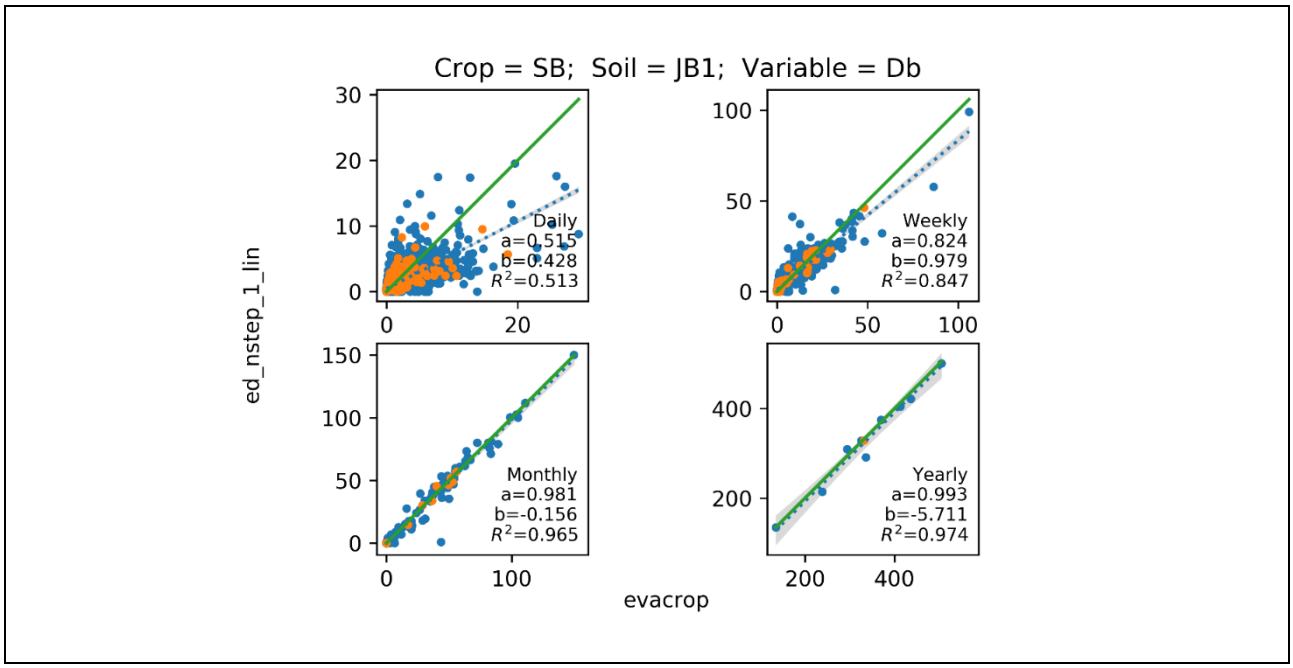

Figure A.2 Drainage from the soil profile for "evacrop" (case i) versus "ed" (case ii) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

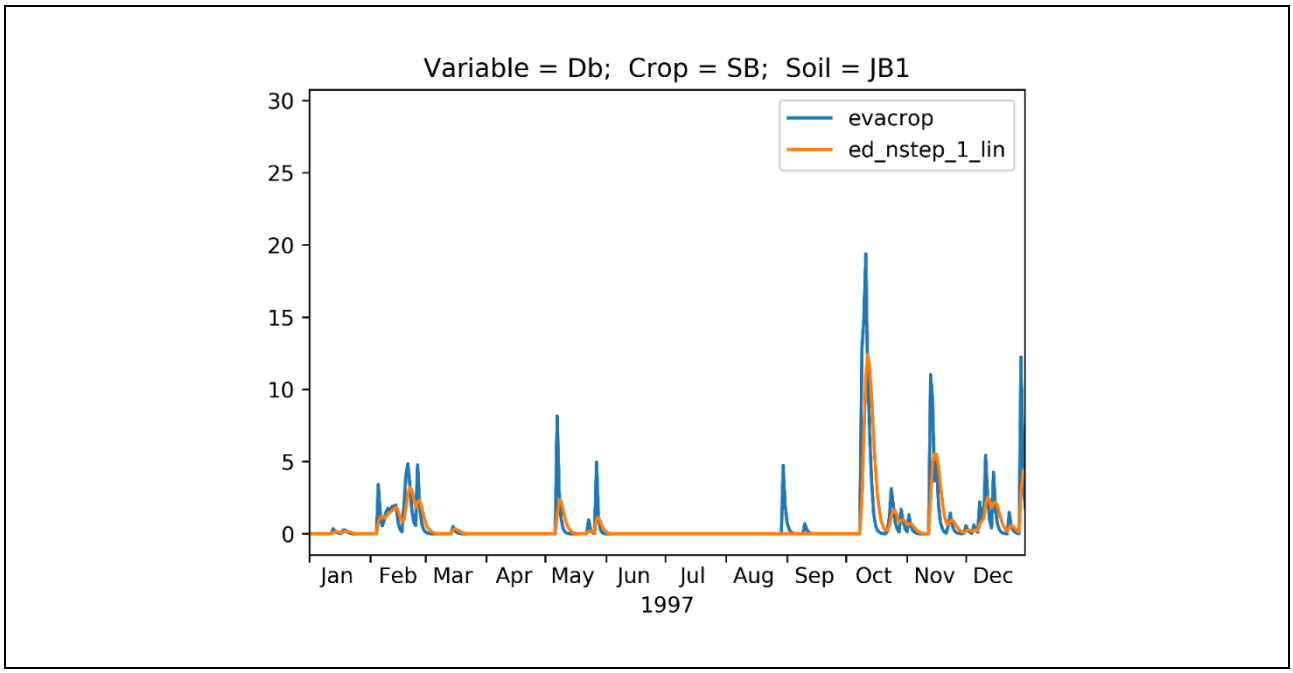

Figure A.3 A year of simulated daily drainage for “evacrop” (case i) versus “ed” (case ii).

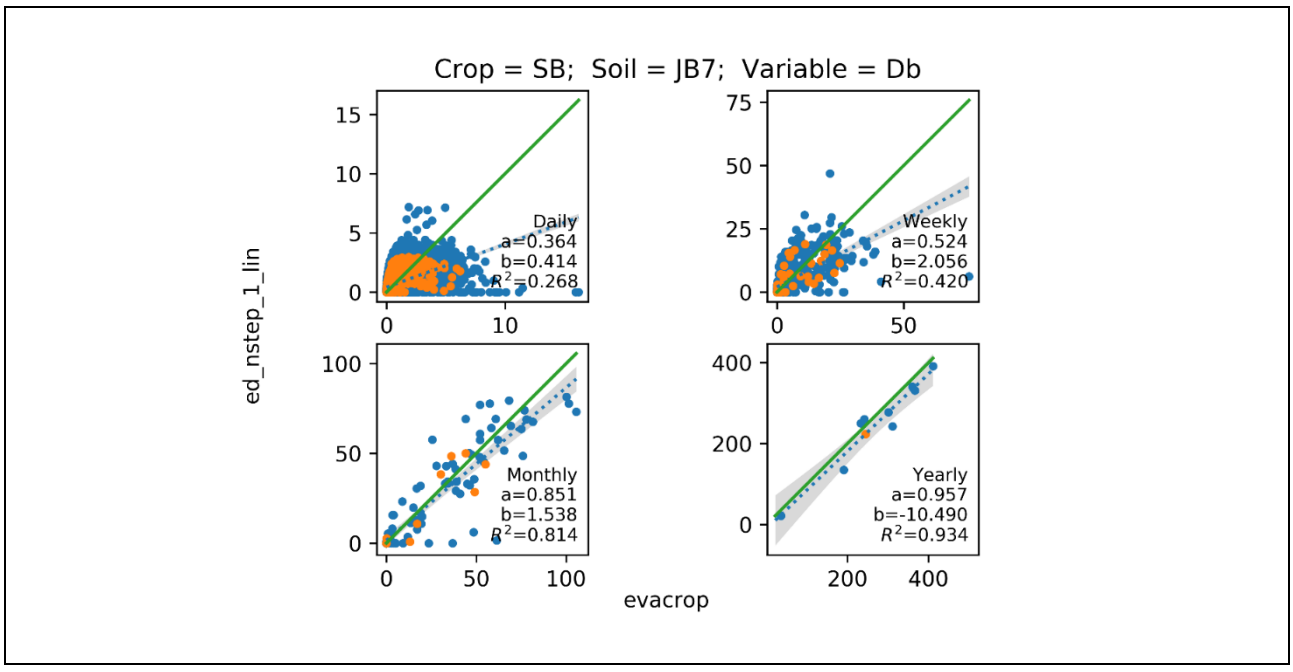

Figure A.4 Drainage from the soil profile for “evacrop” (case i) versus “ed” (case ii) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

A.2. Case (ii) versus case (iii) – dependence on time steps per day for "ed"

These cases use the "ed" water balance function to simulate the growth of spring barley (SB). For case (iii), each day is subdivided into 6 (four hour long) time steps, while for case (ii) there is no subdivision of the daily steps. (Increasing to more than 6 time steps per day did not further change the results.)

Figure A.5 and Figure A.6 show simulated drainage from SB growing on JB1 soil. For yearly drainage, the results are almost unaffected by the subdivision of the daily time steps for simulation of transpiration and drainage. For weekly and daily drainage, there is significant scatter because drainage becomes slower and smoother when simulated using subdivision of the daily time steps. This causes the scatter of points to show somewhat elliptic patterns in Figure A.5, because of the shift in corresponding drainage curves (Figure A.6).

For SB grown on JB7 soil, there is reasonable consistency between results, regardless of whether time step subdivision is used (Figure A.7). The cause of scatter and its pattern is again that drainage becomes a bit slower and smoother when simulated using time step subdivision.

Similar results were obtained for winter wheat growing on JB1 and JB7 soils.

The comparison shows that, other things equal, "ed" with time step subdivision simulates slower and smoother drainage than "ed" without time step subdivision. This tendency is strongest for coarse soil with fast drainage. "'Ed' is set here by default to use 6 steps per day. If sharper or more dynamic results are desired, the number of time steps can be reduced to 1.

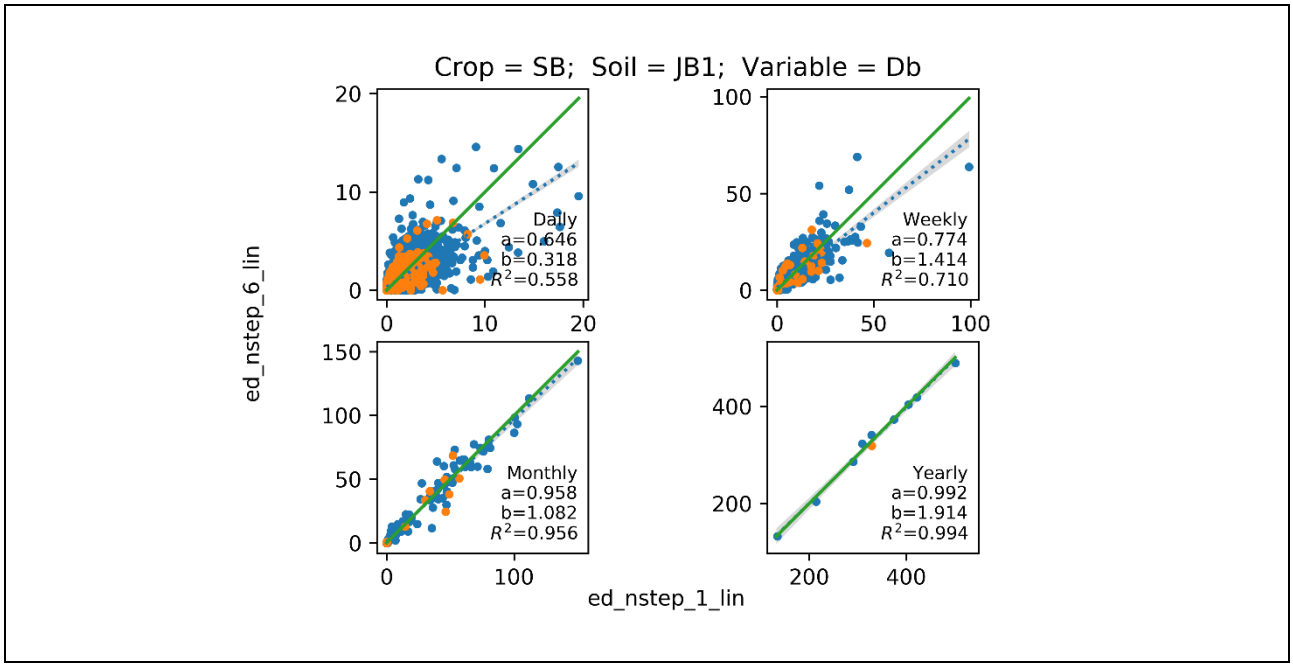

Figure A.5 Drainage from the soil profile for "ed_nstep_1_lin" (case ii) versus "ed_nstep_6_lin" (case iii) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

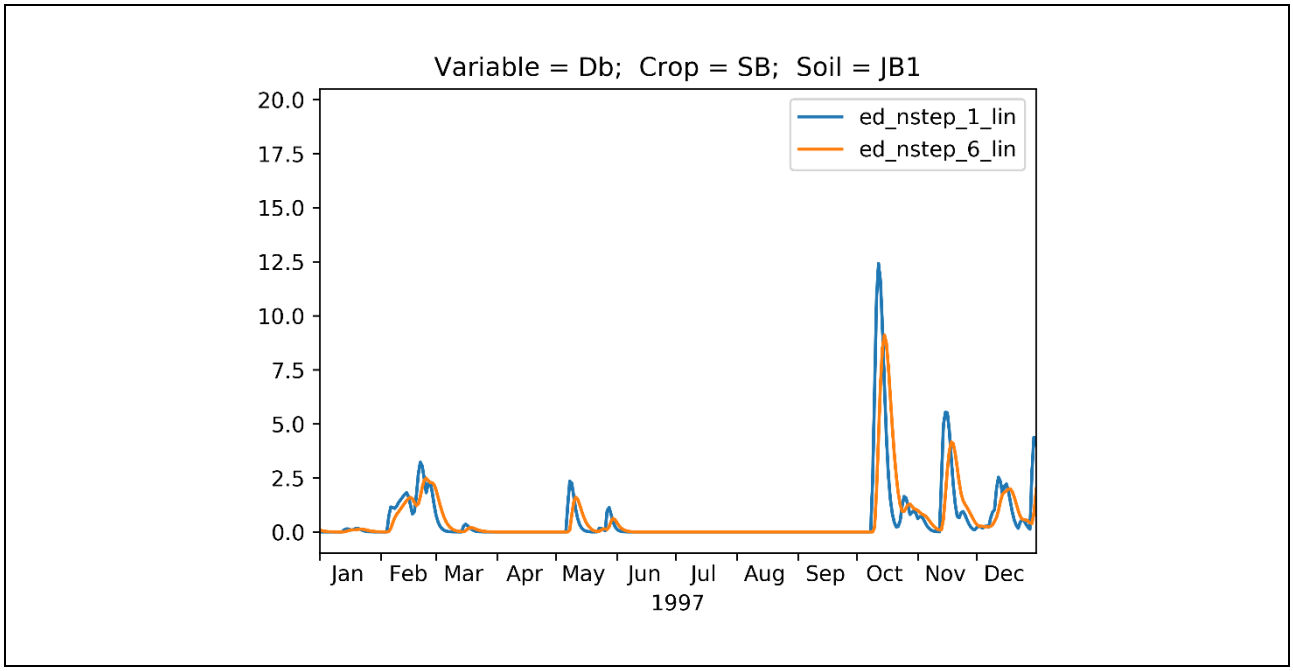

Figure A.6 A year of simulated daily drainage for “ed_nstep_1_lin” (case ii) and “ed_nstep_6_lin” (case iii).

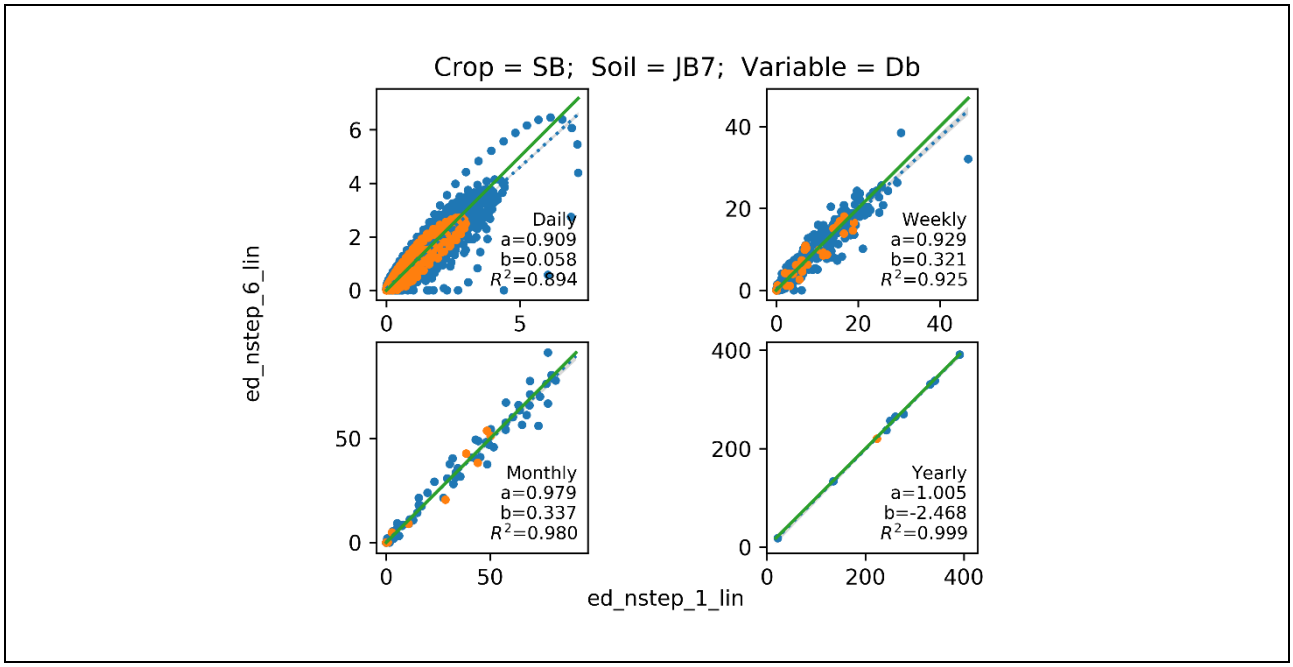

Figure A.7 Drainage from the soil profile for “ed_nstep_1_lin” (case ii) versus “ed_nstep_6_lin” (case iii) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

A.3. Case (iii) versus case (iv) – dependence of "ed" on soil drainage model

These cases use the "ed" water balance function to simulate growing of spring barley (SB). For case (iii), "ed" uses the linear soil drainage model with its default parameter values; for case (iv), it uses the MvG soil drainage model with the MvG default parameter values. Both cases use a 6-step subdivision of the daily time steps.

Figure A.8 and Figure A.9 show simulated drainage from SB growing on JB1 soil. For yearly drainage, the results are almost unaffected of choice of drainage model except for an offset of 10 mm per year. For monthly, weekly, or daily drainage, there is large scatter of points, and the fitted line fall far below the identity line (Figure A.8). This is because drainage is much slower and smoother when simulated using the MvG model instead of the linear model (Figure A.9). By increasing the saturated hydraulic conductivity of each soil horizon by a factor 2.7, the drainage simulated using the MvG model aligns more closely with that simulated using the linear drainage model (Figure A.10 and Figure A.11) although it is still much smoother and less dynamic than the linear model based drainage of case (iii). The factor value was estimated by least-squares fitting of weekly, monthly, and yearly drainage ($D_{sum} = D_b$) simulated by the MvG-based model to corresponding drainage simulated by the linear case (iii) model. The estimation used PEST (Doherty, 2010) with singular value decomposition (SVD).

The results are similar for SB growing on JB7 soil (Figure A.12), as well as for WW growing on JB1 and JB7 soils (results not shown here).

The comparison shows that, "ed" with the MvG soil drainage model and MvG default parameter values simulates much slower and smoother drainage than "ed" with the linear drainage model and default parameter values. Brief testing has shown that drainage simulation results of the two models can be made comparable by parameter adjustment, for example by increasing the saturated hydraulic conductivity of each soil horizon used as default by the MvG model by one order of magnitude for JB1 soil, or by two or three orders of magnitude for JB7 soil.

As default "ed" is set up to use the linear drainage model with its default parameter values. If much smoother results are desired, or if MvG soil drainage parameter values have been determined for a field site, "ed" can be instructed to use the MvG drainage model instead.

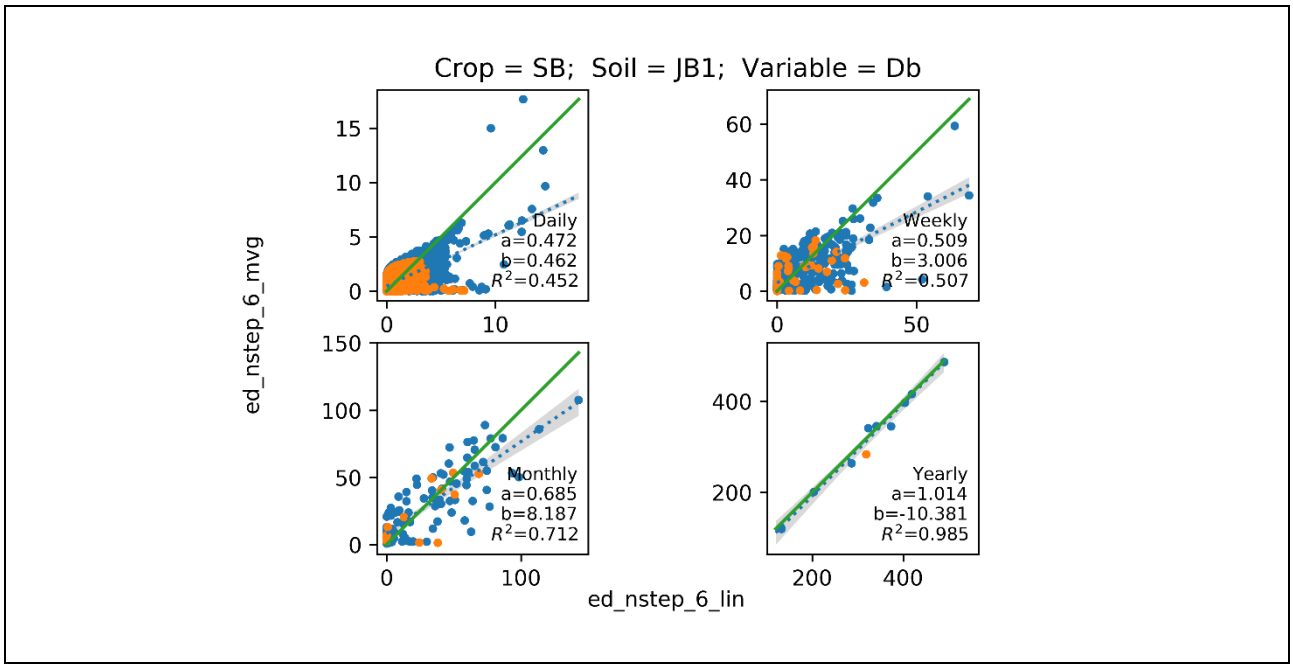

Figure A.8 Drainage from the soil profile for "ed_nstep_6_lin" (case iii) versus "ed_nstep_6_mvg" (case iv) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

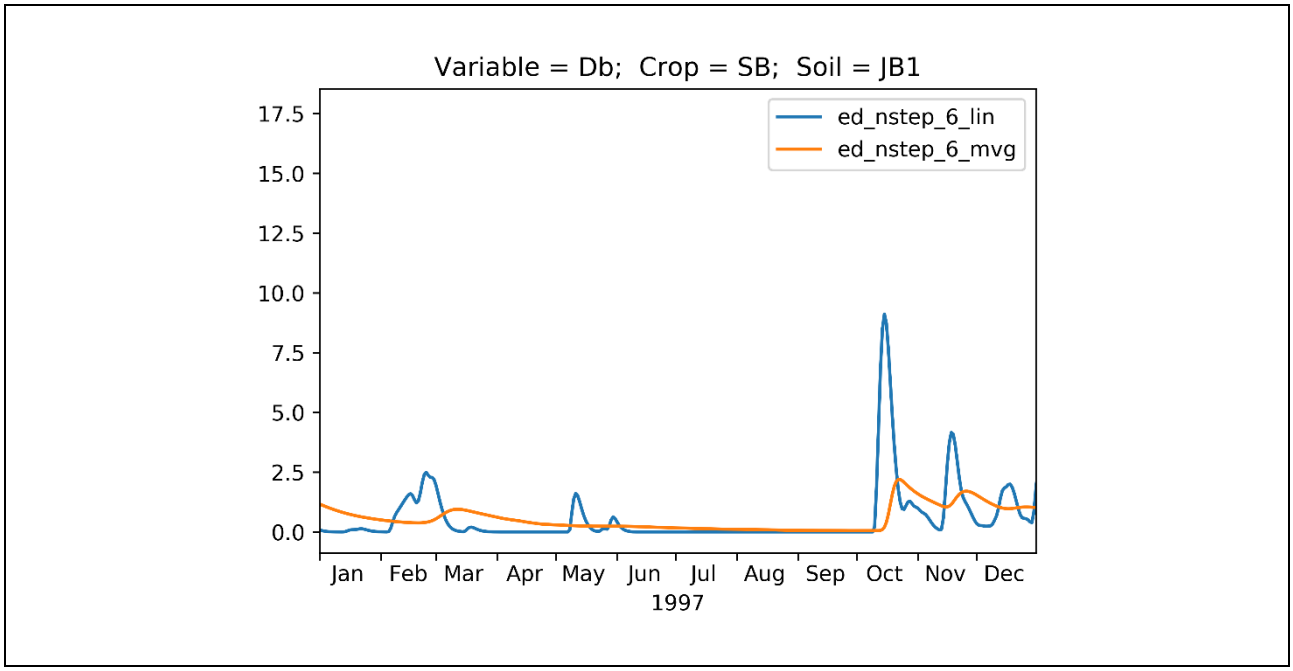

Figure A.9 A year of simulated daily drainage for "ed_nstep_6_lin" (case iii) and "ed_nstep_6_mvg" (case iv).

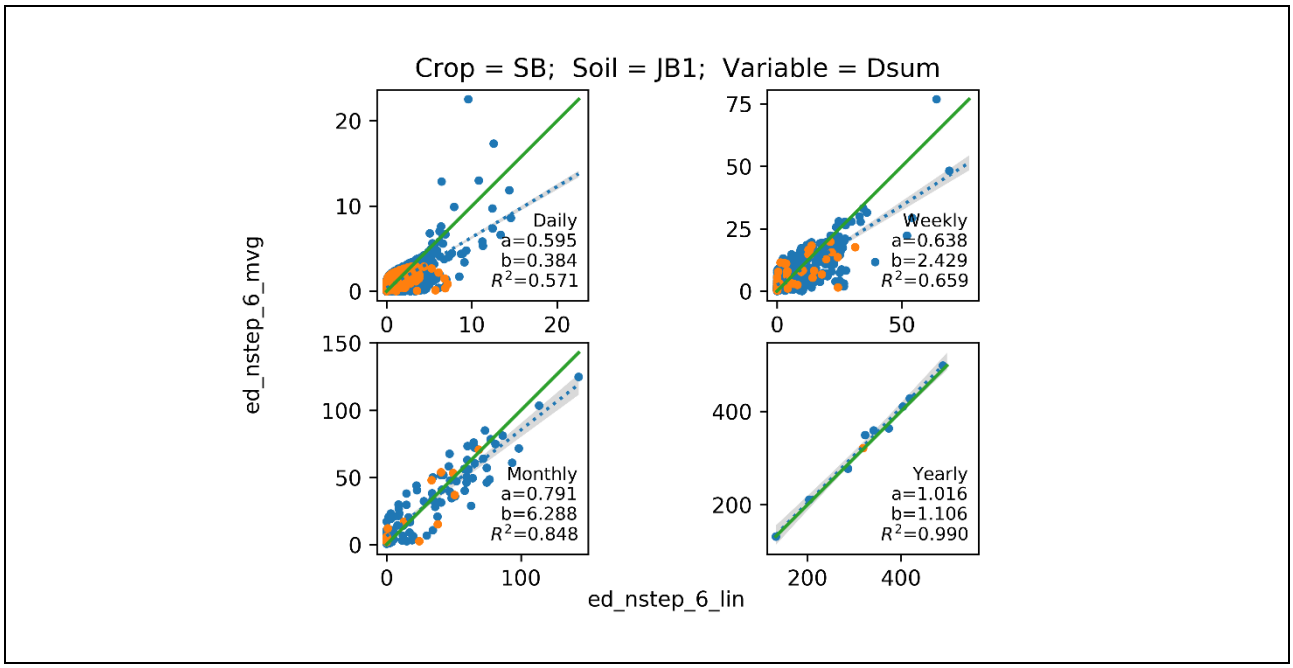

Figure A.10 Drainage from the soil profile for "ed_nstep_6_lin" (case iii) versus "ed_nstep_6_mvg" (case iv) for the period 1990-1999, when saturated hydraulic conductivity of each soil horizon of the MvG model is increased by a factor of 2.7. The meaning of points, lines, and text are as in Figure A.1.

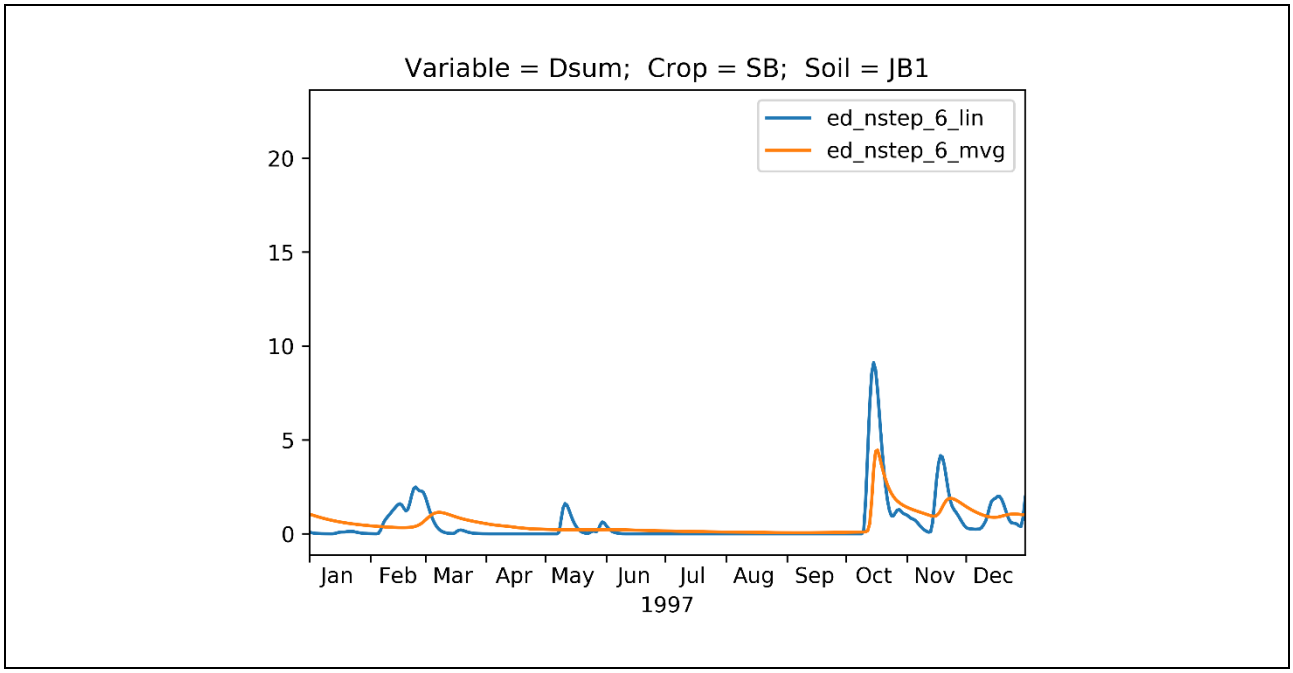

Figure A.11 A year of simulated daily drainage for "ed_nstep_6_lin" (case iii) and "ed_nstep_6_mvg" (case iv), when saturated hydraulic conductivity of each soil horizon of the MvG model is increased by a factor of 2.7.

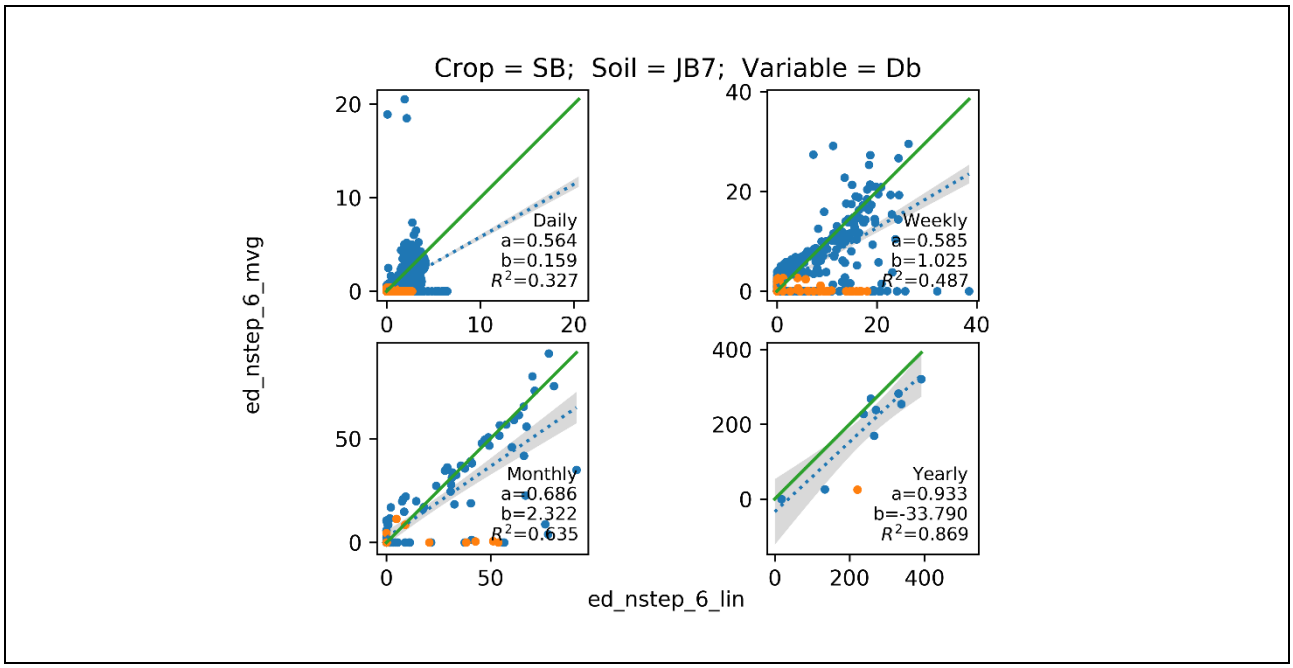

Figure A.12 Drainage from the soil profile for "ed_nstep_6_lin" (case iii) versus "ed_nstep_6_mvg" (case iv) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

A.4. Case (i) versus case (iii) or case (v) – "evacrop" versus "ed" with macro-pore drainage

Case (v) is similar to case (iii), except that it simulates macro-pore drainage along with linear soil drainage, whereas case (iii) does not include macro-pore drainage. Case (iii) is identical to case (ii) except that the former uses $n_{step} = 4$ while the latter uses $n_{step} = 1$ for soil drainage simulation.

For SB grown on JB1 soil, case (v) uses default values for all parameters except for the capacity of the macro pore drainage reservoir, C_{mp}, and the macro pore drainage constant, K_{mp}. K_{mp}, was fixed at a high value (99.0), while the value of C_{mp} was estimated by least-squares fitting of weekly, monthly, and yearly drainage simulated by "ed" (for which $D_{sum} = D_b + D_{mp}$) to corresponding drainage simulated by "evacrop" in case (i) (for which $D_{sum} = D_b$, because "evacrop" does not include macro-pore drainage). The estimation used PEST (Doherty, 2010) with SVD and gave the estimate $C_{mp} = 0.157$.

Figure A.13 and Figure A.14 show that drainage simulated by "ed" in case (iii) is smoother than that simulated by "evacrop". This is in agreement with the results already discussed and commented in sections A.1 and A.2.

Figure A.15 and Figure A.16 show that drainage simulated by "ed" in case (v), which includes macro pore drainage, is very similar to that simulated by "evacrop" in case (i). However, there is a clear tendency for "evacrop" to simulate higher peaks, and weakly higher drainage in general, than "ed" with macro pore drainage" (noticed from line slope, a, or intercept , b, in Figure A.15). There are also rather rare occurrences where "ed" simulates significant drainage whereas "evacrop" does not, and vice versa.

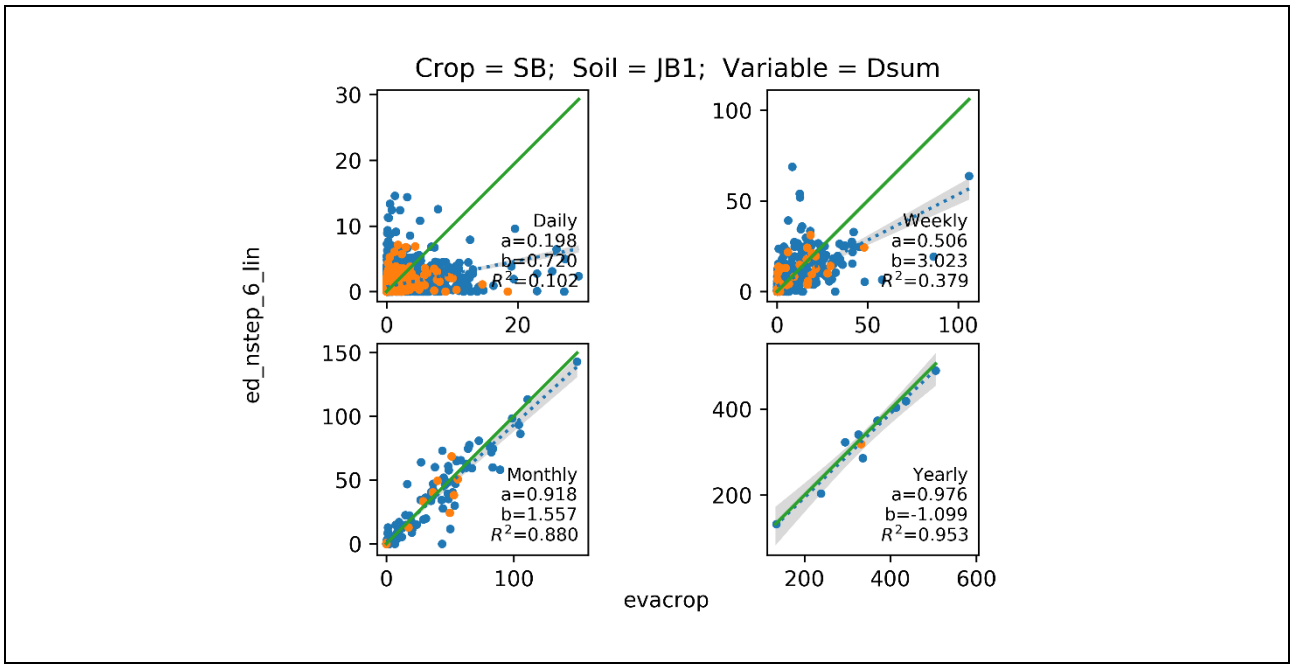

Figure A.13 Drainage from the soil profile for “evacrop” (case i) versus “ed_nstep_6_lin” (case iii) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

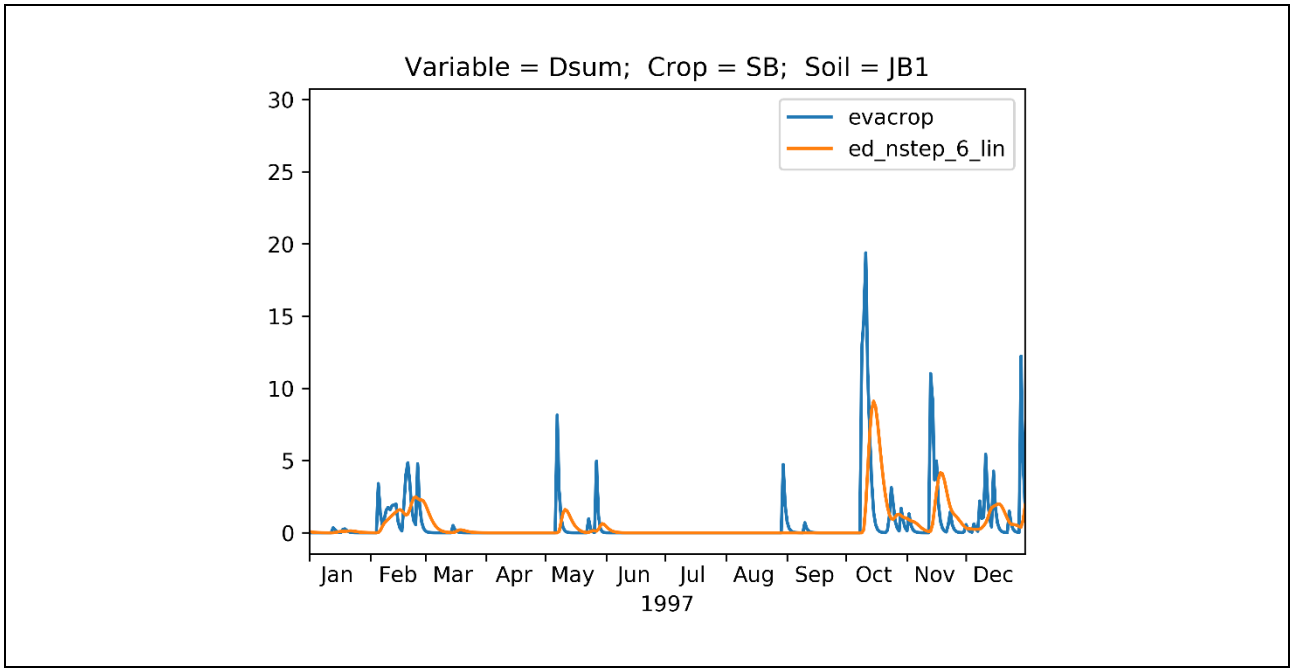

Figure A.14 A year of simulated daily drainage for “evacrop” (case i) and “ed_nstep_6_lin” (case iii).

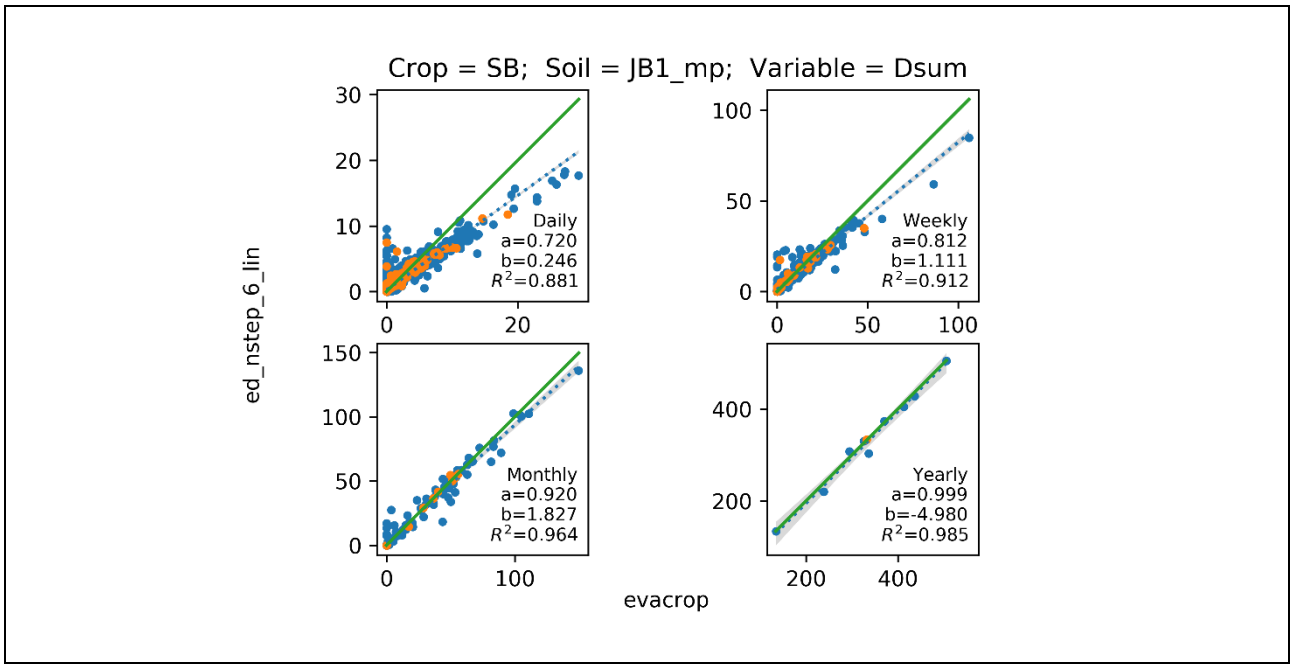

Figure A.15 Drainage from the soil profile for “evacrop” (case i) versus “ed_nstep_6_lin_mp” (case v) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

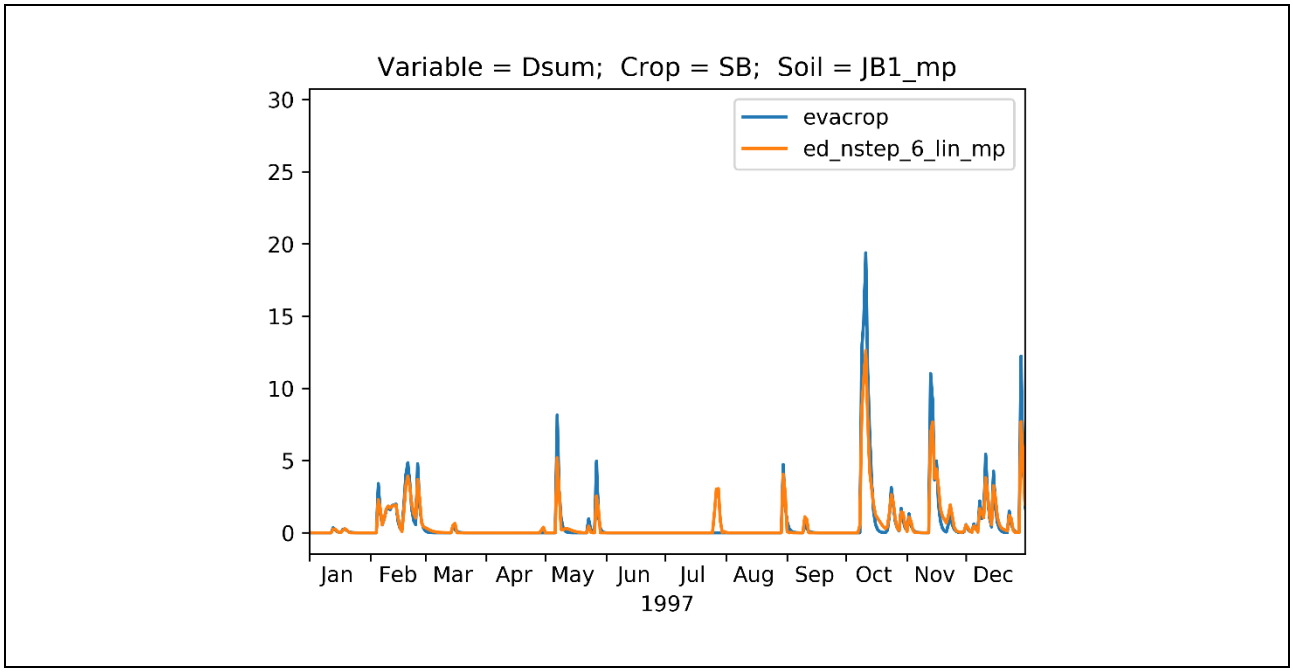

Figure A.16 A year of simulated daily drainage for “evacrop” (case i) and “ed_nstep_6_lin_mp” (case v).

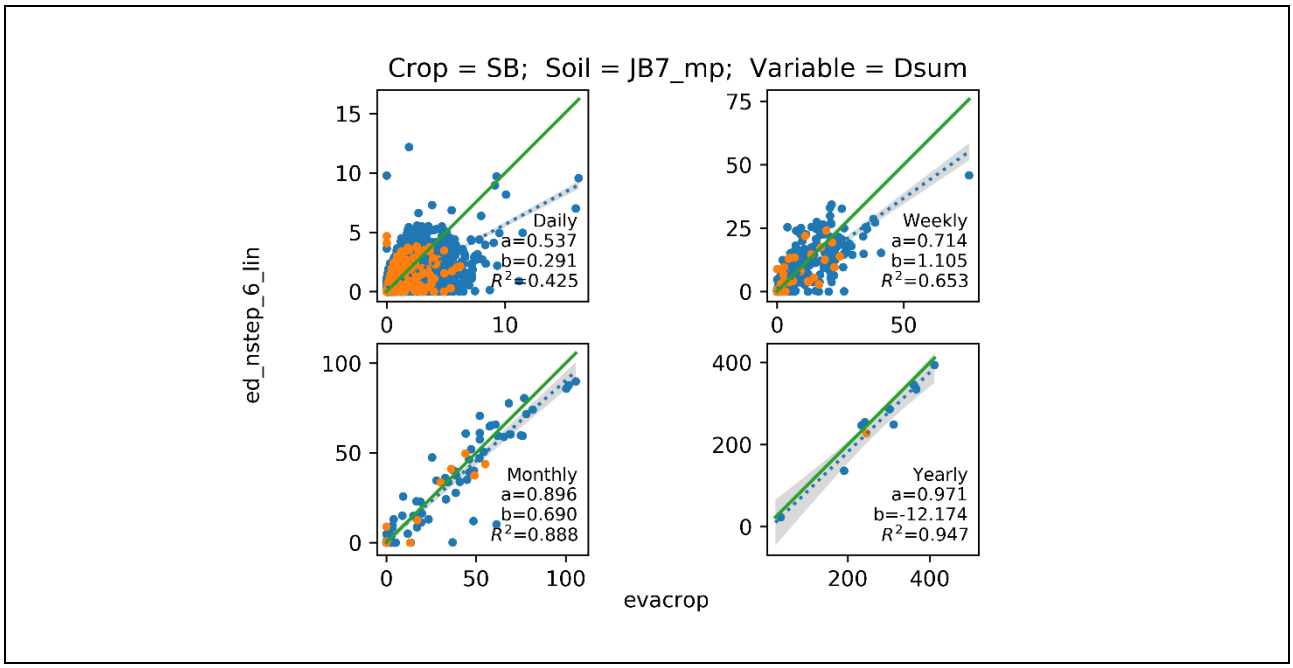

Figure A.17 Drainage from the soil profile for “evacrop” (case i) versus “ed_nstep_6_lin_mp” (case v) for the period 1990-1999. The meaning of points, lines, and text are as in Figure A.1.

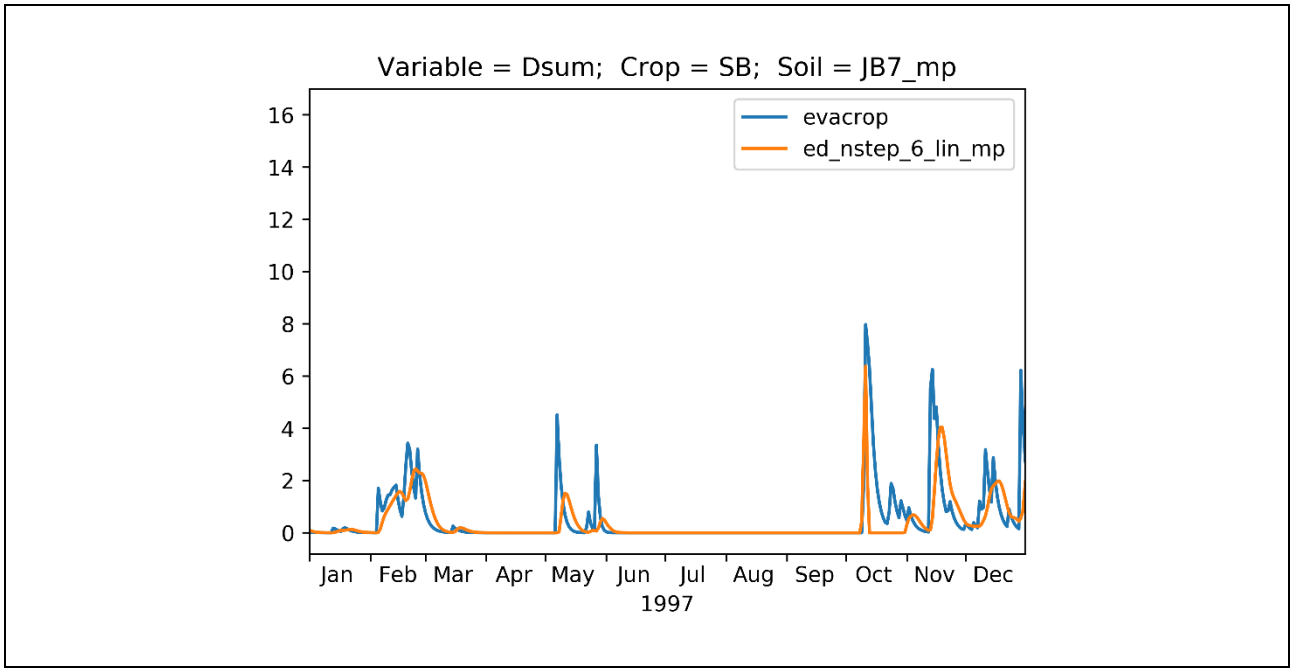

Figure A.18 A year of simulated daily drainage for “evacrop” (case i) and “ed_nstep_6_lin_mp” (case v).

In the case (v) simulation of SB grown on JB1 soil, for the entire simulation period 19% of the simulated total drainage is soil drainage and 81% is macro-pore drainage. For “ed_nstep_6_lin_mp” in Figure A.16, all spikes come from simulated macro pore drainage.

For SB grown on JB7 soil, case (v) uses default values for all parameters except for the soil drainage constant, k_{qr}, the capacity of the macro pore drainage reservoir, C_{mp}, and the macro pore drainage constant, K_{mp}. K_{mp} was fixed at a high value (99.0), while the value of k_{qr} and C_{mp}, respectively, were estimated by fitting of weekly, monthly, and yearly drainage simulated by “ed” to corresponding drainage simulated by “evacrop”. The estimation using PEST (Doherty, 2010) with SVD gave the $k_{qr} = 0.584$ and $C_{mp} = 0.256$. (In Edcrop, the default value of k_{qr} is 0.3; for C_{mp}, the default value is set to a more or less arbitrary value 0.15 because of lack of experience.) Using the estimated parameter values, 95% of simulated total drainage during the simulation period is soil drainage; only 5% is macro pore drainage.

Figure A.17 and Figure A.18 compares drainage simulated for SB grown on JB7 soil. Case (v) drainage is smoother and have lower peaks than case (i) drainage. However, the temporal variation of drainage simulated in case (v) is much closer to “evacrop” drainage than “ed” drainage simulated (but not shown here) by “ed” using default $k_{qr} = 0.3$ and default $K_{mp} = 0.0$. For “ed_nstep_6_lin_mp” in Figure A.18, only the drainage spike seen in the beginning of October comes from simulated macro pore drainage; all other drainage is simulated as soil drainage.

A.5. Conclusions

In Edcrop, users can choose between two alternative water balance functions. The first alternative, “evacrop”, is a straight copy of the function used in the original Evacrop code by Olesen and Heidmann (2002), simulating flow through the soil profile as flow through two linear reservoirs using daily time steps. The second alternative, “ed”, simulates flow through the soil profile as flow through four linear or nonlinear reservoirs using daily or sub-daily time steps. The analyses of the appendix show the following.

In a straight comparison between “evacrop” “ed”, “ed” simulation used daily time step and a linear soil drainage model. Both simulations used the same default soil parameter values. The comparison shows that “evacrop” simulates faster and peakier flow through the soil than “ed”, and it simulates less actual evapotranspiration and larger drainage than “ed”. Temporal variation of drainage is smoother when simulation is using “ed” instead of “evacrop”.

Use of “ed” with time step subdivision simulates slower and smoother drainage than use of “ed” without time step subdivision. This tendency is strongest for coarse soil with fast drainage.

Using (i) “ed” with the MvG soil drainage model and MvG default parameter values simulates much slower and smoother drainage than using (ii) “ed” with the linear drainage model and default parameter values. Brief testing indicates that drainage simulation results using either of the two drainage models can be made comparable by parameter adjustment.

For JB1 (sandy) soil, “ed” with macro-pore drainage simulates nearly identical total drainage (and actual evapotranspiration) to “evacrop”.

For JB7 soil, “ed” without macro pore drainage but with increased soil drainage constant simulates total drainage quite similar to that of “evacrop”.

Appendix B: Comparison of Edcrop and Daisy results

This appendix illustrates the similarities and differences in simulation results obtained by the simpler Edcrop code compared to the more advanced Daisy code (Hansen et al., 1990). We used Daisy version 5.93 downloaded from https://daisy.ku.dk/download/windows/

Daisy is an advanced soil plant atmosphere system model developed to "enable simulation of crop production, water dynamics and nitrogen dynamics in crop production at various agricultural management practices and strategies" (Hansen et al., 1990). Concerning crop growth and water balance, there are some remarkable differences between the more advanced Daisy and the simpler Edcrop.

Daisy contains a soil temperature model, which influences root growth (penetration rate). Experiments with Daisy done during the comparison with Edcrop has shown that in Daisy root growth also depends on available water. In Edcrop, root growth is constant. Root density is also constant in Edcrop, whereas it varies with depth in Daisy.

In Daisy, leaf (canopy) development depends on sum of ambient (air) temperature, accumulated top dry matter, photosynthesis and thereby soil water availability, respectively. In Edcrop, it only depends on sum of air temperature.

Daisy's soil water model solves the Richards equation using a finite difference scheme, where Edcrop simulates flow as taking place through a series of linear or nonlinear reservoirs. In Edcrop, simulated transpiration only depends on the water content within the entire root zone, whereas

Daisy simulated transpiration depends on the root density and the water content around the roots in the various soil depths. In Edcrop, simulated transpiration depends only on the water content within the entire root zone.

Daisy simulates ponding on the surface, which Edcrop does not. The latter contains an evaporation zone with a certain capacity, C_e, which feeds soil evaporation.

B.1. Base example – spring barley on sandy soil

The base example is for spring barley (SB) growing on a sandy (JB1) soil. The climate data covers the ten years 1990-1999 measured at Taastrup, Denmark (in file dk-taastrup.dwf, downloaded with the Daisy program version 5.93 from https://daisy.ku.dk/download/windows/).

The base example Daisy simulation used the input given in Table B.1. This includes simulation of flow using Mualem/van Genuchten functions.

In Edcrop, the simulation uses four nonlinear reservoirs, with drainage computed using equation (73) and default Mualem/van Genuchten parameter values, identical to those used in the Daisy simulation. Also, in this simulation Edcrop uses the same maximum root depth, $z_{max} = 1000$ mm, as used by Daisy. (This is twice of the default value for JB1 soil in Edcrop.) Sow date is April 5th, latest possible harvest date is August 20th, and *autoharvest* is set to *yes*.

Table B.1 Daisy input file used for base example.

```
;; Including external library-files
(input file "tillage.dai")
(input file "crop.dai")
(input file "dk-horizon.dai")
(input file "fertilizer.dai")
(input file "dk-management.dai")
(input file "log.dai")

;; Weather data
(weather default "dk-taastrup.dwf")

;;  Parameterisation of column
(defcolumn "JB1" default
     (Soil (MaxRootingDepth 100 [cm])
           (horizons (  -25 [cm] "Ap_JB1")
                     (  -75 [cm] "B_JB1")
                     ( -100 [cm] "C_JB1")))
     (Groundwater deep)
     (OrganicMatter original (init (input 3000 [kg C/ha/y]))))

;;  Selecting column
(column "JB1")

;; Start and end of simulation.
(time 1990 1 1)
(stop 2000 1 1)

;; Selecting management
(manager activity
  "SBarley w. MF" "SBarley w. MF" "SBarley w. MF" "SBarley w. MF" "SBarley w. MF"
  "SBarley w. MF" "SBarley w. MF" "SBarley w. MF" "SBarley w. MF" "SBarley w. MF"
)

;; Selecting output
;;  Description that will occur in all output files
(description "Spring Barley; Soil: JB1; Weather: Taastrup")

;(activate_output
;(log_prefix "Ex1/")
(output harvest
        ("Crop Production" (when daily))
;;      Water balance 0-100 cm
        ("Field water" (to -100 [cm])(when daily)
         (where "Daily_FWB.dlf"))
)
```

Figure B.1 compares the simulated potential evapotranspiration, E_p, which is the product of reference evapotranspiration and crop coefficient, c.f. (50). The two simulation codes, Daisy and Edcrop, use the same reference evapotranspiration but simulates the crop coefficient somewhat differently, as illustrated later. Figure B.1 shows strong correlation between the daily, weekly, and monthly E_p simulated by the two codes, respectively, excluding the 1993 results; an explanation of this exclusion follows later. For some days, weeks, or months, the plotted data deviate from the respective fitted line, which is close to the identity line, but the coefficient of determination, R^2, for the fit is close to 1.0, emphasizing that the Daisy and Edcrop simulations are very similar. For the yearly results, the fitted line deviate from the identity line, but the latter nearly falls inside the 95% confidence band of the former, and R^2 is 0.91. Excluding year 1993, the simulated average annual E_p is 483 mm/y for Edcrop and 493 mm/y for Daisy. This supports that there is nearly a 1:1 relationship between yearly E_p simulated by the two codes.

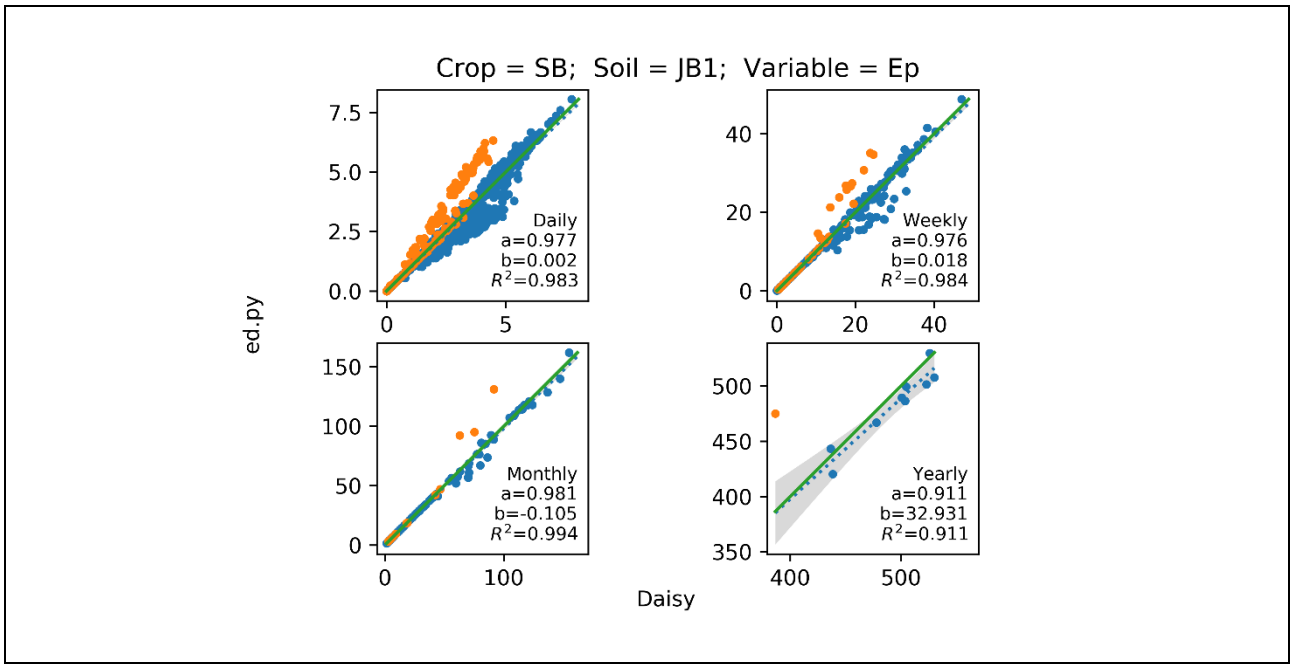

Figure B.1 Comparison of potential evapotranspiration, E_p, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for the period 1990-1999. Orange points are for 1993, blue points for the remaining years. Blue dashed line is line fitted to the blue points, and the grey shaded area is the 95% confidence band for the fitted line. The green line is the identity (1:1) line. The text says whether it is daily, weekly, monthly, or yearly results; a is the slope, b the intercept, and R^2 the coefficient of determination of the fitted line.

Figure B.2 compares actual evapotranspiration, E_a. The two sets of model results compare more or less as for E_p although more spread of points around the fitted line is seen for daily, weekly and monthly E_a; the larger spread quantifies by the lesser R^2 for E_a than for E_p. Excluding year 1993 (will be explained later), the average annual E_a simulated by Edcrop is 375 mm/y, while it is 378 mm/y for Daisy.

Figure B.3 compares drainage from the subzone, D_b. This indicates a 1:1 relationship between monthly, and yearly, drainage simulated by Edcrop and Daisy, respectively, with high values of R^2, and the identity line falling inside the confidence band of the fitted line. For daily and monthly results, the spread of points around the fitted line is larger, R^2 therefore less, and the slope of the fitted line is a little less than 1.0. Excluding year 1993, the average annual D_b simulated by Edcrop is 263 mm/y, while it is 258 mm/y for Daisy.

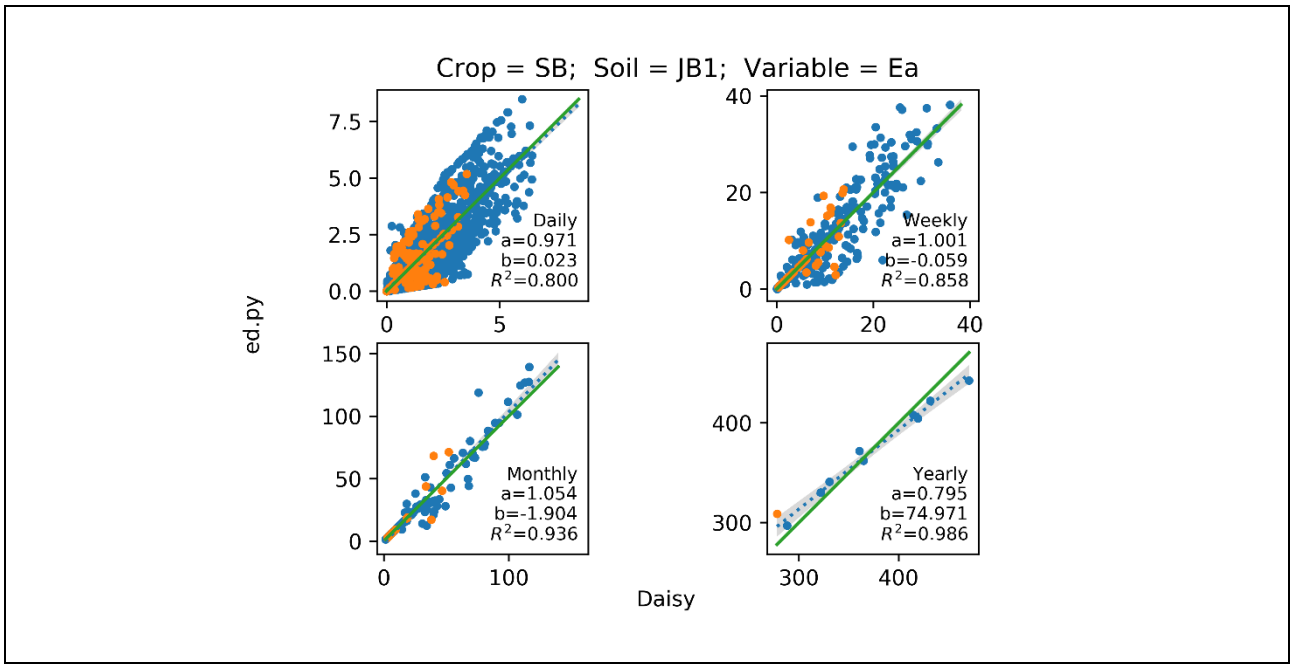

Figure B.2 Comparison of actual evapotranspiration, E_a, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for the period 1990-1999. The caption of Figure B.1 explains the meaning of points, lines, and text.

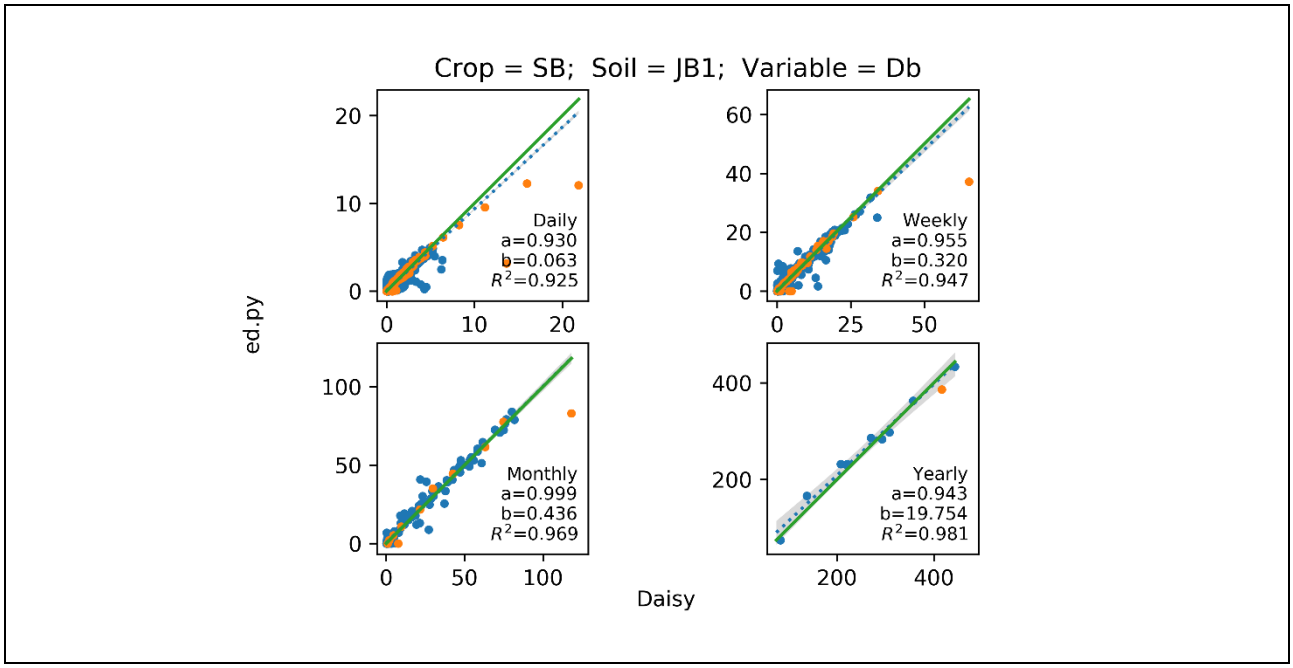

Figure B.3 Comparison of drainage, D_b, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for the period 1990-1999. The caption of Figure B.1 explains the meaning of points, lines, and text.

The plot of annual E_p in Figure B.1 shows that the 1993 value is an outlier. The Edcrop-simulated value is much larger than the Daisy value. The monthly E_p values for May, June and July are also outliers, with Edcrop-values being the larger. The explanation for this is difference between Edcrop and Daisy in simulation of plant development, leading to difference in simulation of crop coefficient and thus E_p. The difference between Daisy and Edcrop is illustrated in Figure B.4, which shows simulated leaf area and root depth for 1991 and 1993, respectively. During 1991, Daisy and Edcrop simulation of plant development is quite similar, while during 1993 the two simulations are very different. Analysis has shown that the main reason for the difference during 1993 is that this year had a very dry spring: total precipitation during March was 8.9 mm, during April 10.9 mm, and during May 6.6 mm. For the ten-year period 1990-1999, the average precipitation for the three months were 41.0 mm, 45.6 mm, and 38.9 mm, respectively. Very low spring precipitation, not compensated by irrigation, will stunt plant growth. This is simulated by Daisy, but not by Edcrop for which temperature is the only driving variable for plant growth. During the simulated decade, only 1993 had a precipitation pattern that stunted Daisy-simulated plant growth; for the other years, the two codes simulate quite similar plant development, as shown in Figure B.4 for 1991.

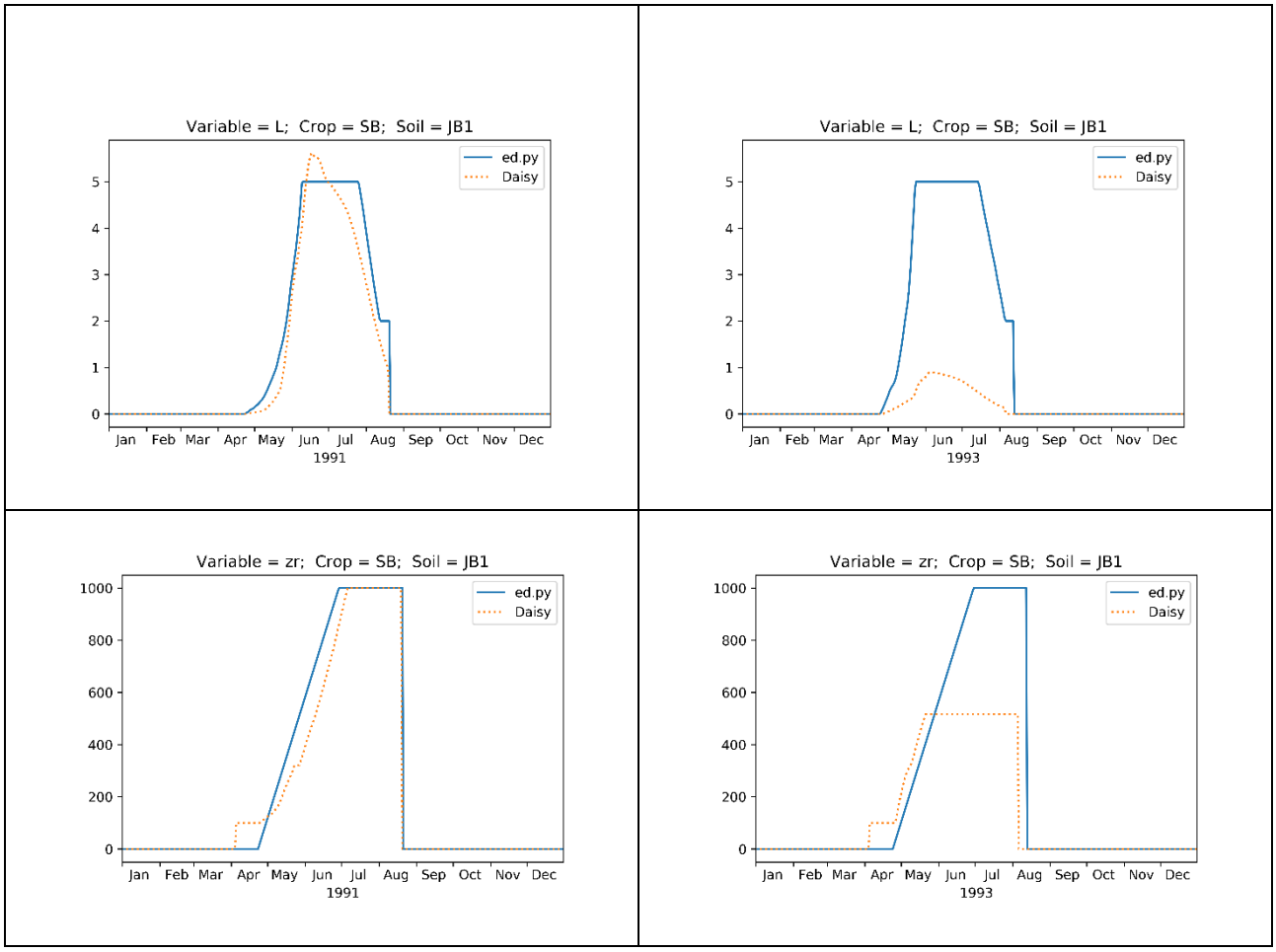

Figure B.4 Total leaf area index, L, and root depth, z_r, simulated for the base case by Daisy and Edcrop (named ed.py in plot) during 1991 and 1993, respectively.

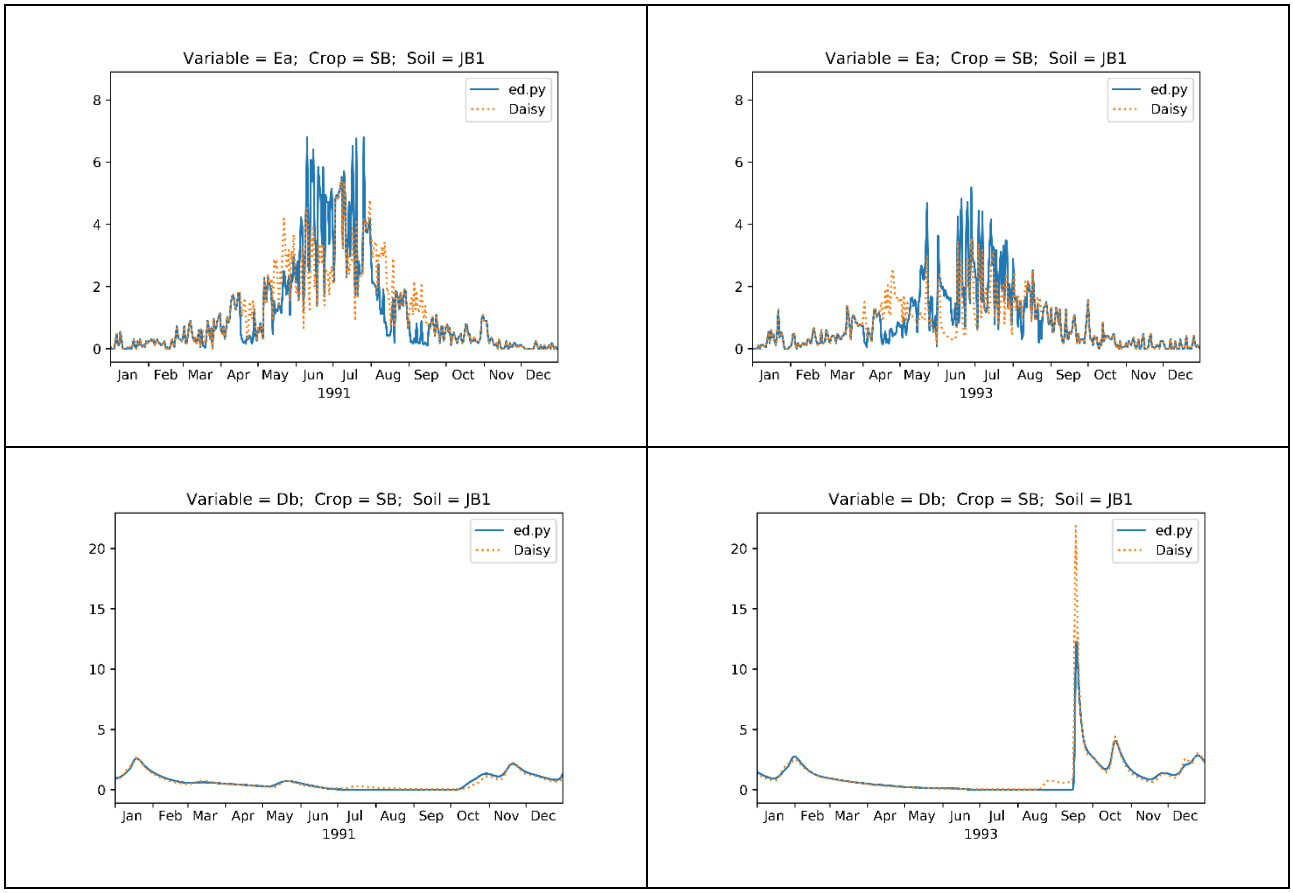

Figure B.5 Actual evapotranspiration, E_a, and drainage from subzone, D_b, simulated for the base case by Daisy and Edcrop (named ed.py in plot) during 1991 and 1993, respectively.

Figure B.5 shows simulated actual evapotranspiration and simulated drainage from the subzone for 1991 and 1993. Both Figure B.2 and Figure B.5 show that on daily and weekly basis there is difference in simulated actual evapotranspiration. For simulated drainage, the two sets of simulation results are quite similar. However, in 1991 some simulated rises and decreases are slightly offset to each other; in 1993, Daisy simulates an increase in mid-August, which is not simulated by Edcrop, and Daisy simulates a higher peak in drainage in mid-September than Edcrop. The difference in simulation of this peak is also noticed in Figure B.3 as the far right, too low orange point values of drainage in the daily and weekly plots. Similar but less remarkable differences can be seen during the other simulated years (not shown), but in general the Daisy and Edcrop simulated time series of drainage compares fairly well for the base case.

B.2. Winter wheat on sandy soil

This example uses the same input as the base example, except that the crop is winter wheat (WW) instead of spring barley. For winter wheat, sow date is September 10th, latest possible harvest date is September 1st, *autoharvest is set to yes*, and maximum root depth is set to 1 m.

Figure B.6 shows strong correlation between the daily, weekly, monthly, and yearly E_p simulated by the two codes, respectively. The same is the case for simulated actual evapotranspiration (Figure B.7) and simulated drainage (Figure B.8). Excluding year 1993, the simulated average annual E_p is 531 mm/y for Edcrop and 530 mm/y for Daisy. The simulated average annual E_a is 424 mm/y for Edcrop and 413 mm/y for Daisy. The simulated average annual D_b is 220 mm/y for Edcrop and 231 mm/y for Daisy.

Figure B.9 shows simulated leaf area, L, and root depth, z_r, for 1991 and 1993, respectively. It is noticed that Edcrop simulates that leaf area and root depth reduce during winter (from November through March), whereas Daisy does not simulate such reduction. Despite of this difference, there is little difference between the two codes in simulation of evapotranspiration during winter, because in Denmark the reference evapotranspiration is very small during winter. The difference in winter wheat growth models therefore has almost no influence on simulated water balance.

Figure B.9 also shows that during most of the 1993 growing season, the simulated root depth is at its maximum for both Daisy and Edcrop. Therefore, in this case potential evapotranspiration for 1993 is not an outlier in Figure B.6.

Figure B.9 finally shows that Daisy simulates harvest of WW to happen a couple of weeks earlier than Edcrop.

Figure B.10 shows simulated actual evapotranspiration and simulated drainage from the subzone for 1991 and 1993. There are the same similarities and differences between the simulations as was mentioned for the base case. In general, the Daisy and Edcrop simulated time series of drainage compares fairly well for this case, with Edcrop producing slightly smoother time series.

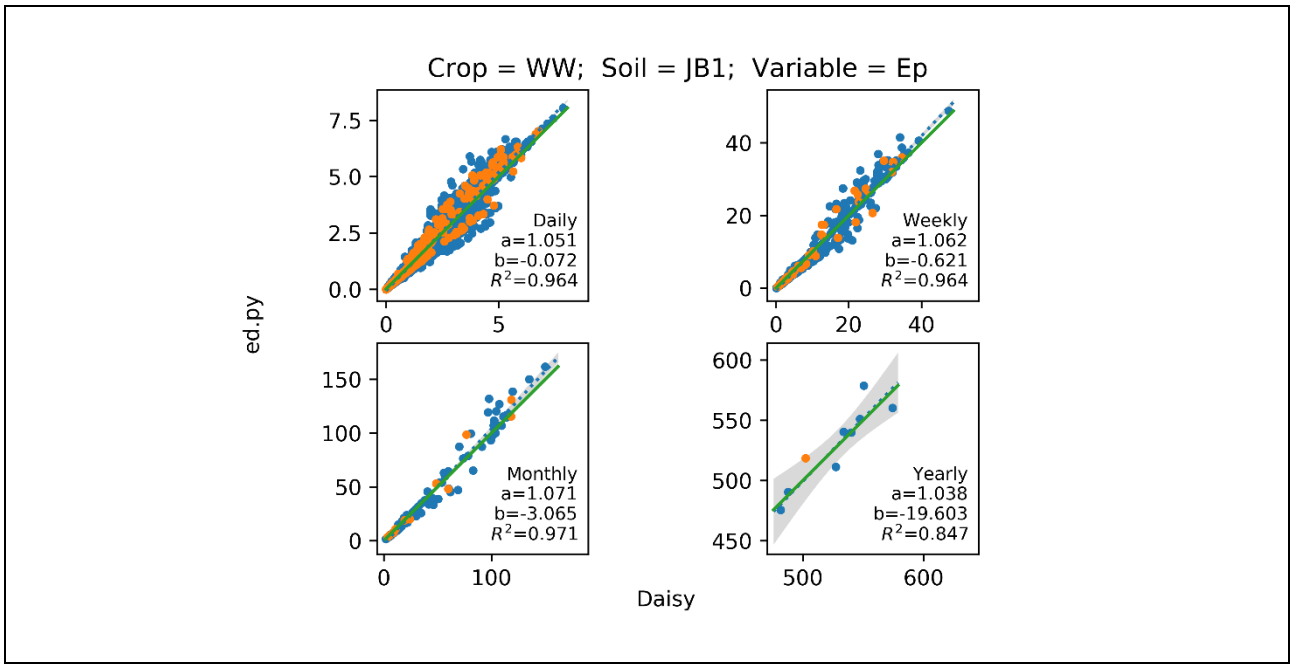

Figure B.6 Comparison of potential evapotranspiration, E_p, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for the period 1990-1999. . The caption of Figure B.1 explains the meaning of points, lines, and text.

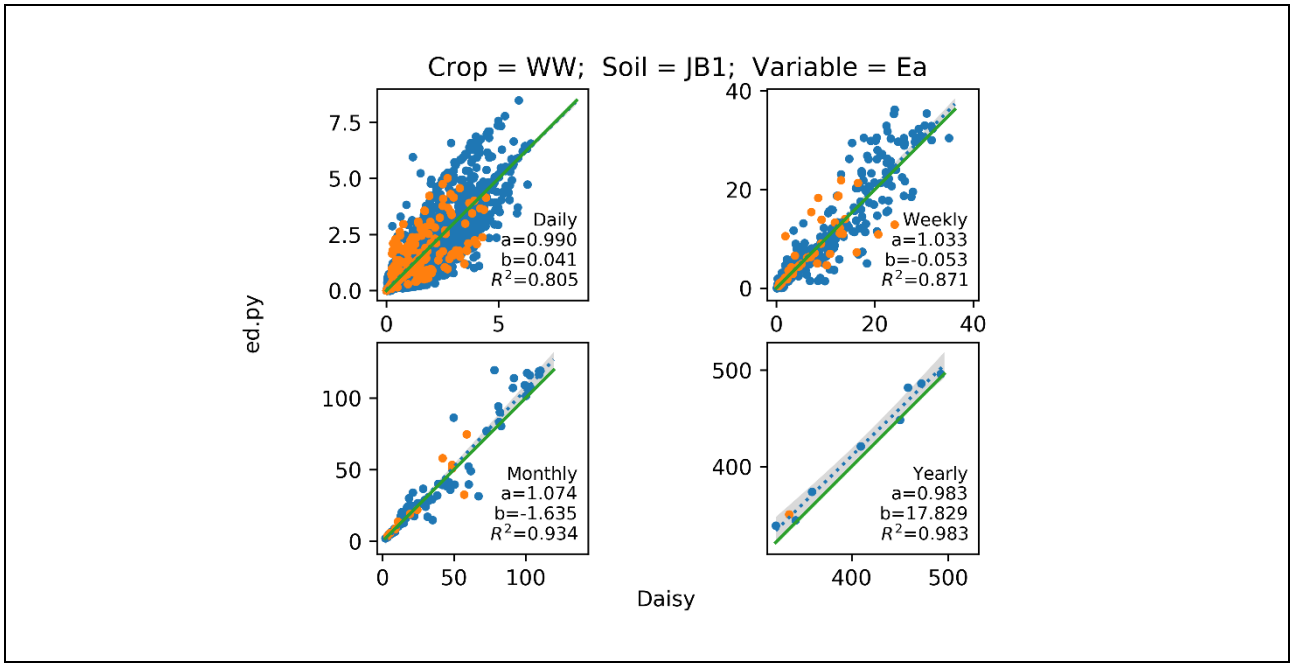

Figure B.7 Comparison of actual evapotranspiration, E_a, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for the period 1990-1999. The caption of Figure B.1 explains the meaning of points, lines, and text.

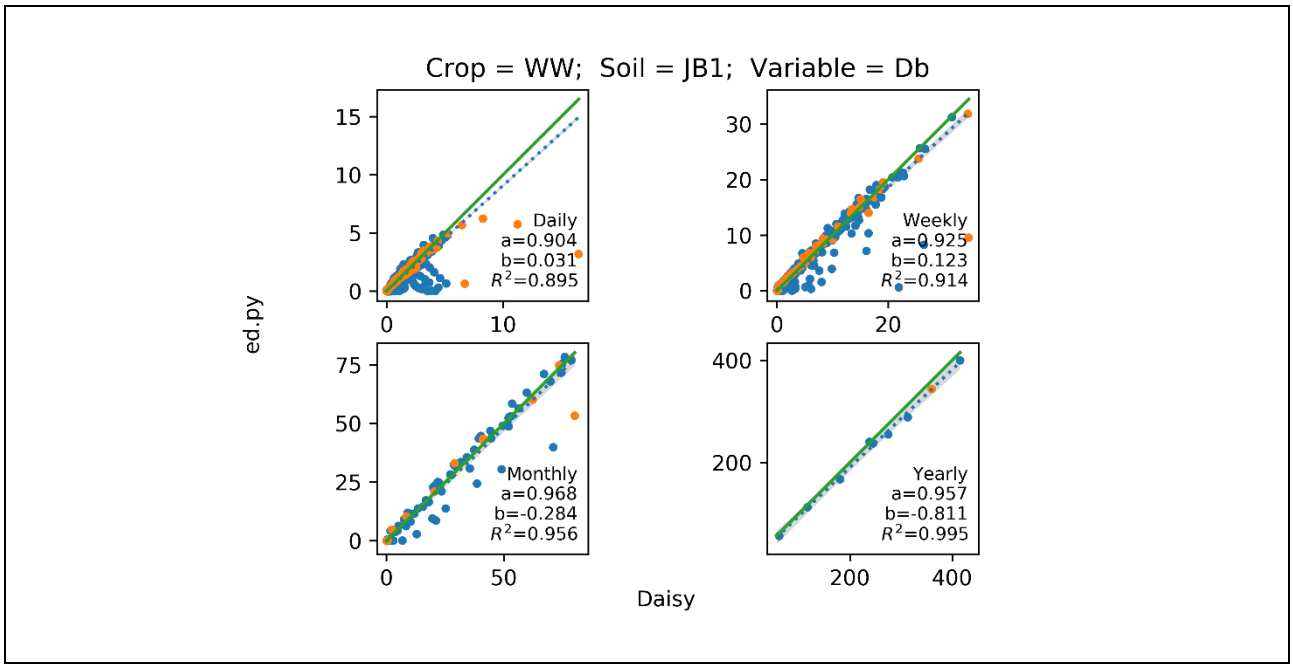

Figure B.8 Comparison of drainage, D_b, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for the period 1990-1999. The caption of Figure B.1 explains the meaning of points, lines, and text.

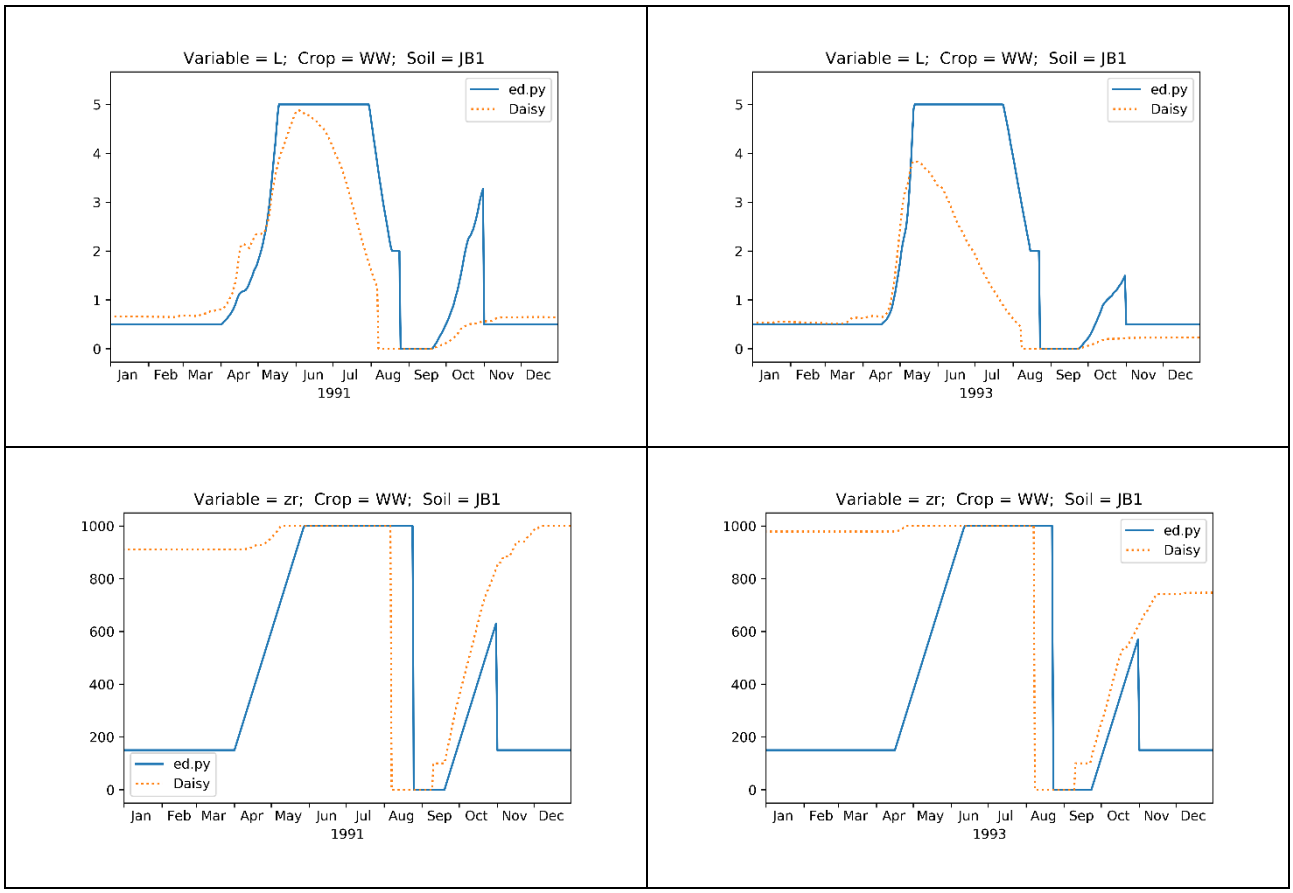

Figure B.9 Total leaf area index, L, and root depth, z_r, respectively, for winter wheat (WW) growing on sandy soil (JB1).

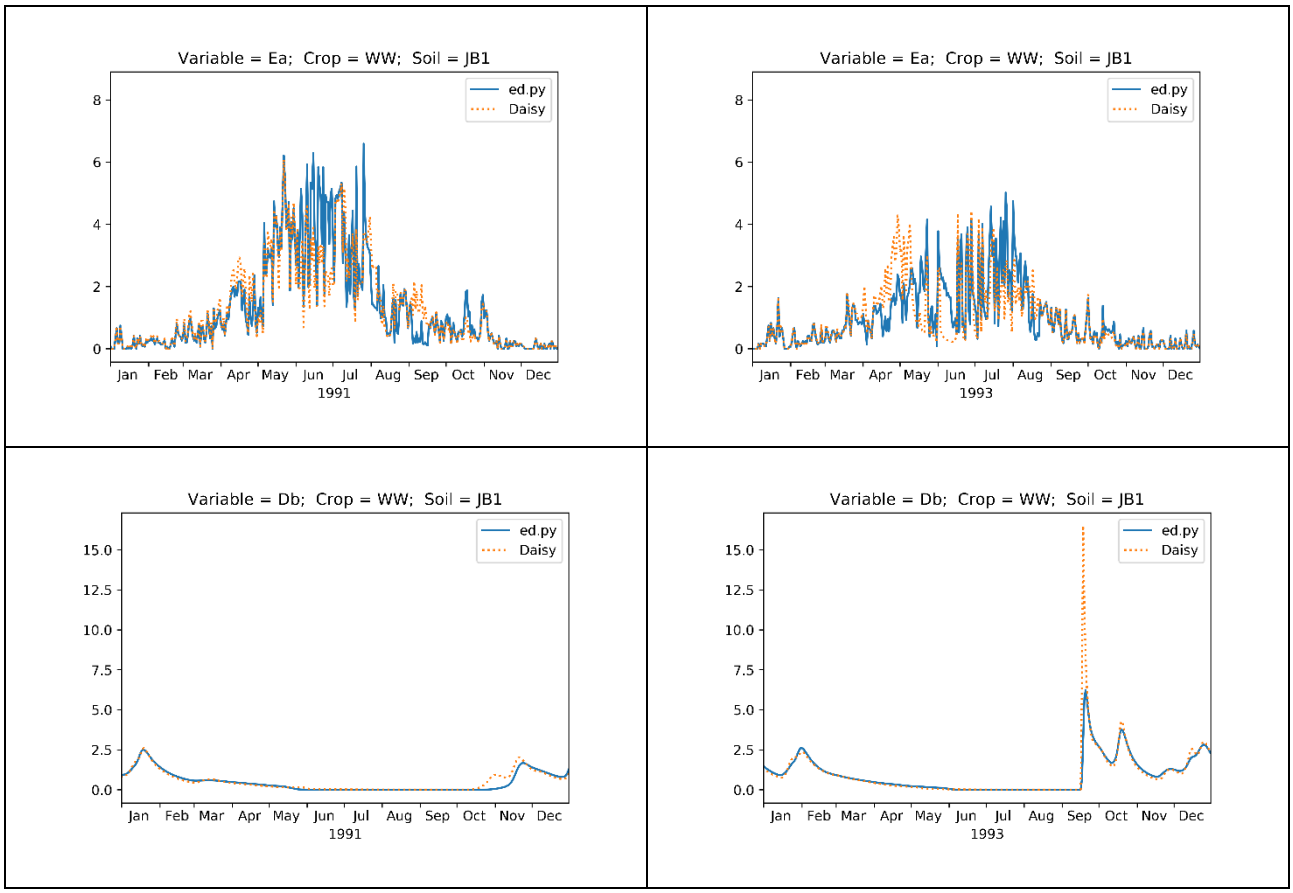

Figure B.10 Actual evapotranspiration, E_a, and drainage from subzone, D_b, simulated for the base case by Daisy and Edcrop (named ed.py in plot) during 1991 and 1993, respectively.

B.3. Spring barley on clayey soil

This example compares Daisy and Edcrop simulation results for spring barley (SB) growing on a clayey (JB7) soil. The example uses the same climate input data as for the base example. Since the temperature data are identical, Edcrop simulation of plant growth is identical for the present and the base example. Since plant available water also drives Daisy simulation of plant growth, changing from sandy to clayey soil make changes in Daisy simulation of plant growth from the base case.

Figure B.11 compares the simulated potential evapotranspiration, E_p. This shows strong correlation between the daily, weekly, and monthly E_p, also for 1993. For the yearly results, the fitted line deviates from the identity line, falling outside much of the 95% confidence band. Excluding years 1990 and 1993, the simulated average annual E_p is 481 mm/y for Edcrop and 494 mm/y for Daisy. Anyhow, as for SB on JB1 soil, there is nearly a 1:1 relationship between yearly E_p simulated by the two codes.

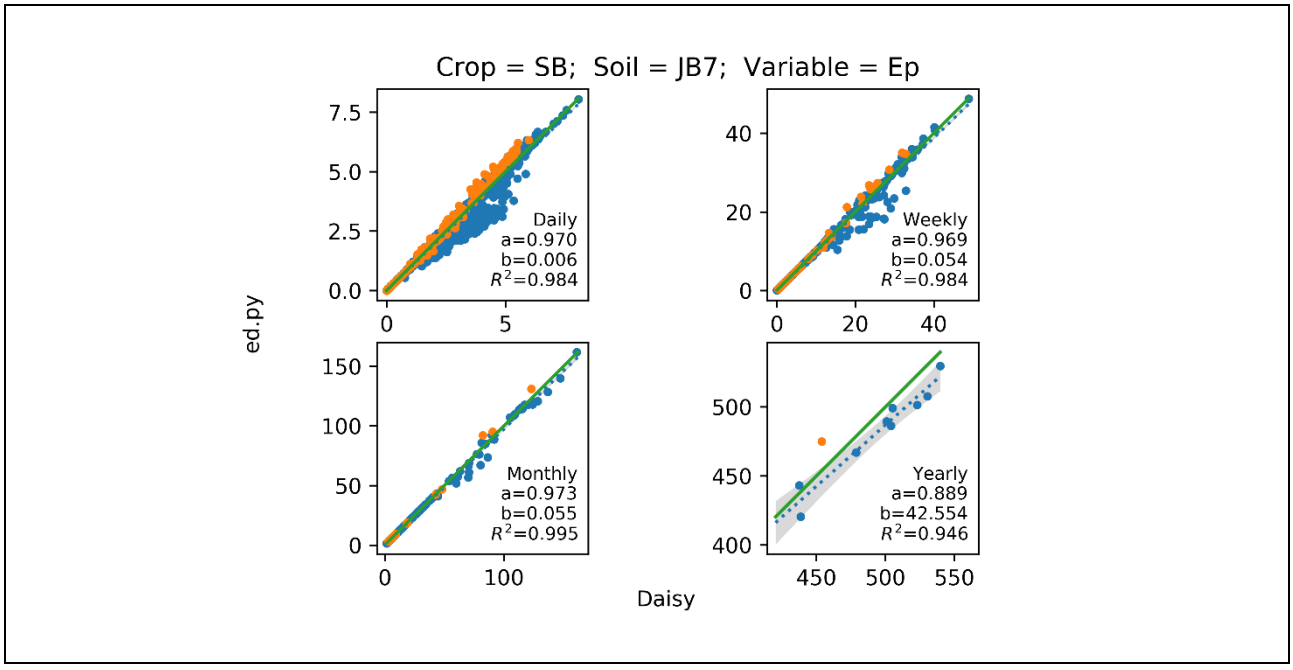

Figure B.11 Comparison of potential evapotranspiration, E_p, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for spring barley (SB) growing on clayey soil (JB7) from 1990 to 1999. The caption of Figure B.1 explains the meaning of points, lines, and text.

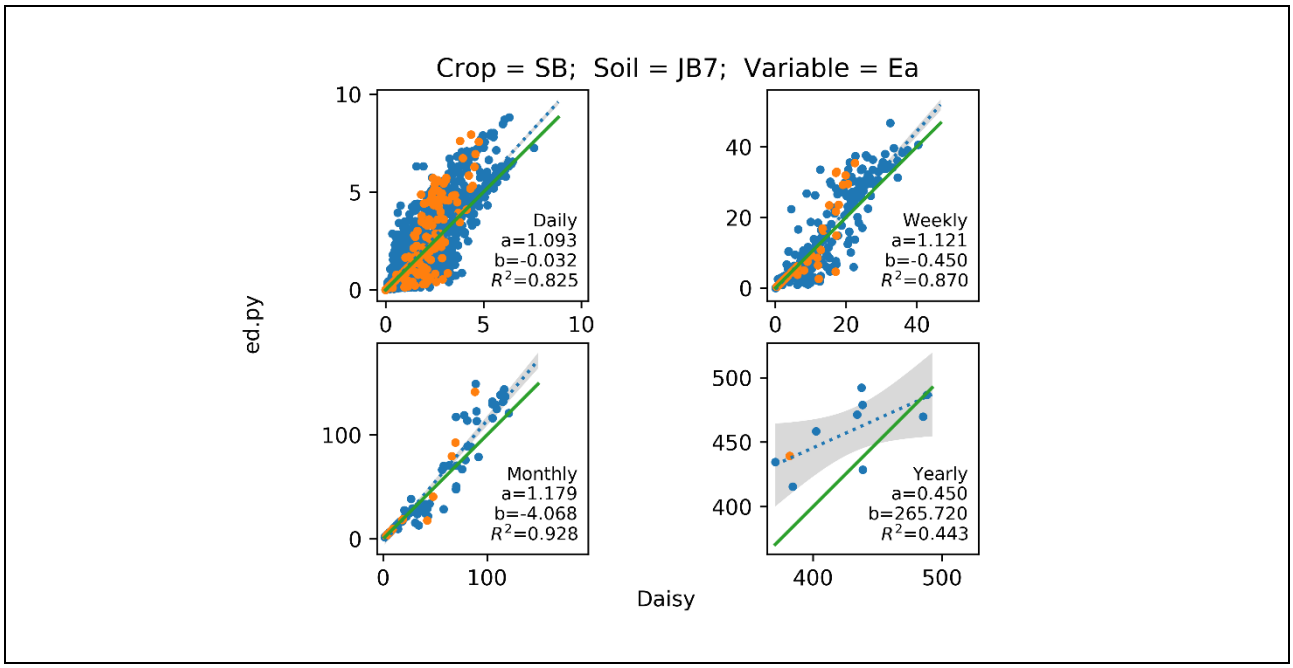

Figure B.12 Comparison of actual evapotranspiration, E_a, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for spring barley (SB) growing on clayey soil (JB7) from 1990 to 1999. The caption of Figure B.1 explains the meaning of points, lines, and text.

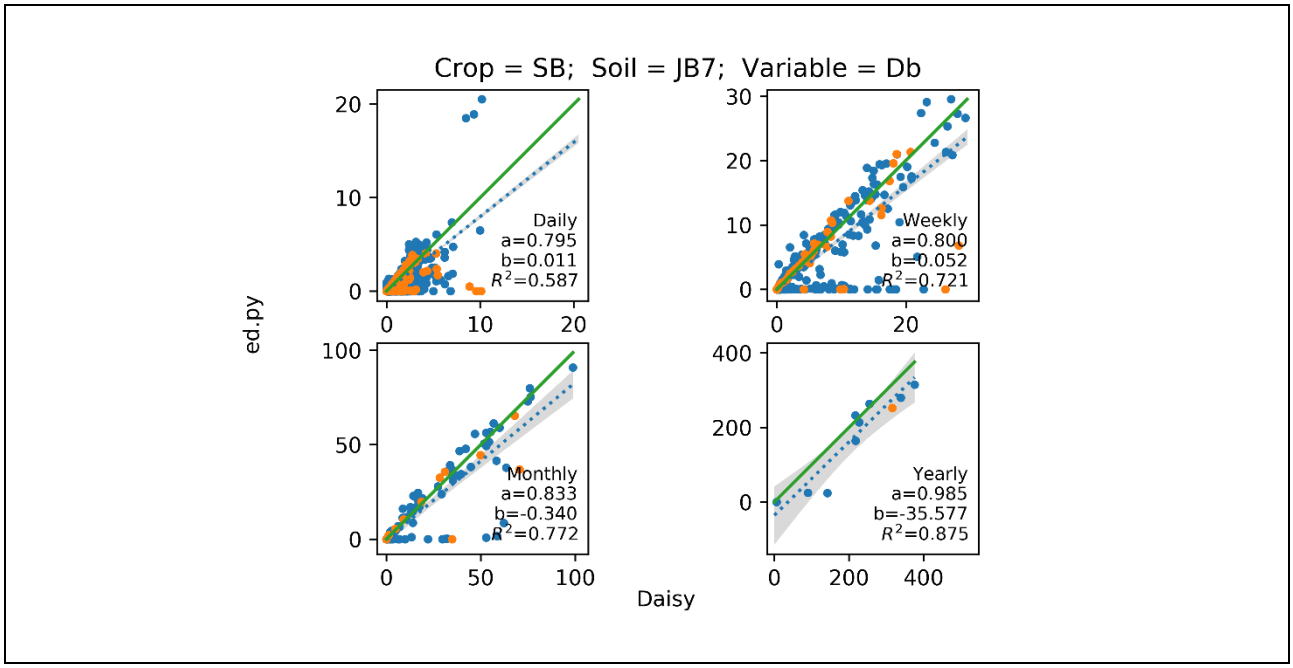

Figure B.13 Comparison of drainage, D_b, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for spring barley (SB) growing on clayey soil (JB7) from 1990 to 1999. The caption of Figure B.1 explains the meaning of points, lines, and text.

Figure B.12 compares actual evapotranspiration, E_a. This shows poorer correlation between the two sets of model results than for SB growing on JB1 soil (Figure B.2). E_p. Excluding years 1990 and 1993, the average annual E_a simulated by Edcrop is 457 mm/y, while it is 430 mm/y for Daisy – a difference of roughly 6%. The difference in yearly E_a tends to decrease with increasing E_a.

Figure B.13 compares drainage from the subzone, D_b. It is noticed that for daily, weekly, and monthly D_b, there are several days, weeks, or months where Daisy simulates large drainage but Edcrop simulates no or very low drainage. This results in Edcrop simulating an average annual drainage of 187 mm/y, while Daisy simulates 216 mm/y – a difference of 29 mm/y, corresponding to 13% of Daisy simulated annual drainage.

Figure B.14 shows simulated leaf area and root depth for 1991 and 1993, respectively. During 1991, Daisy and Edcrop simulation of plant development is quite similar, and during 1993 the two simulations are much more comparable than for an SB crop growing on JB1 soil (Figure B.4). This is because, also in dry years, the amount of plant available water is higher in the JB7 (clayey) soil than in the JB1 (sandy), causing less stunt of plant growth.

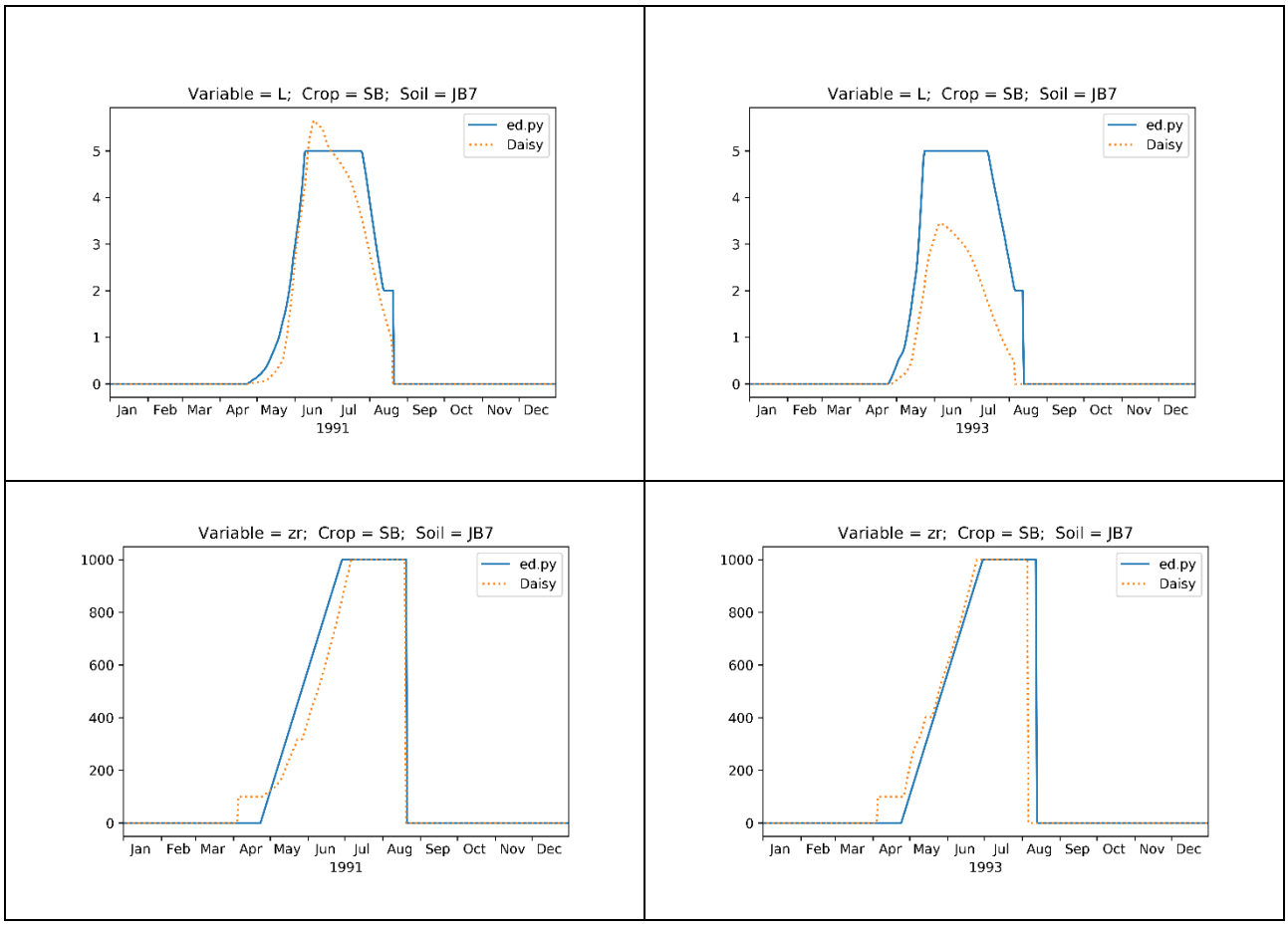

Figure B.14 Total leaf area index, L, and root depth, z_r, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for spring barley (SB) growing on clayey soil (JB7).

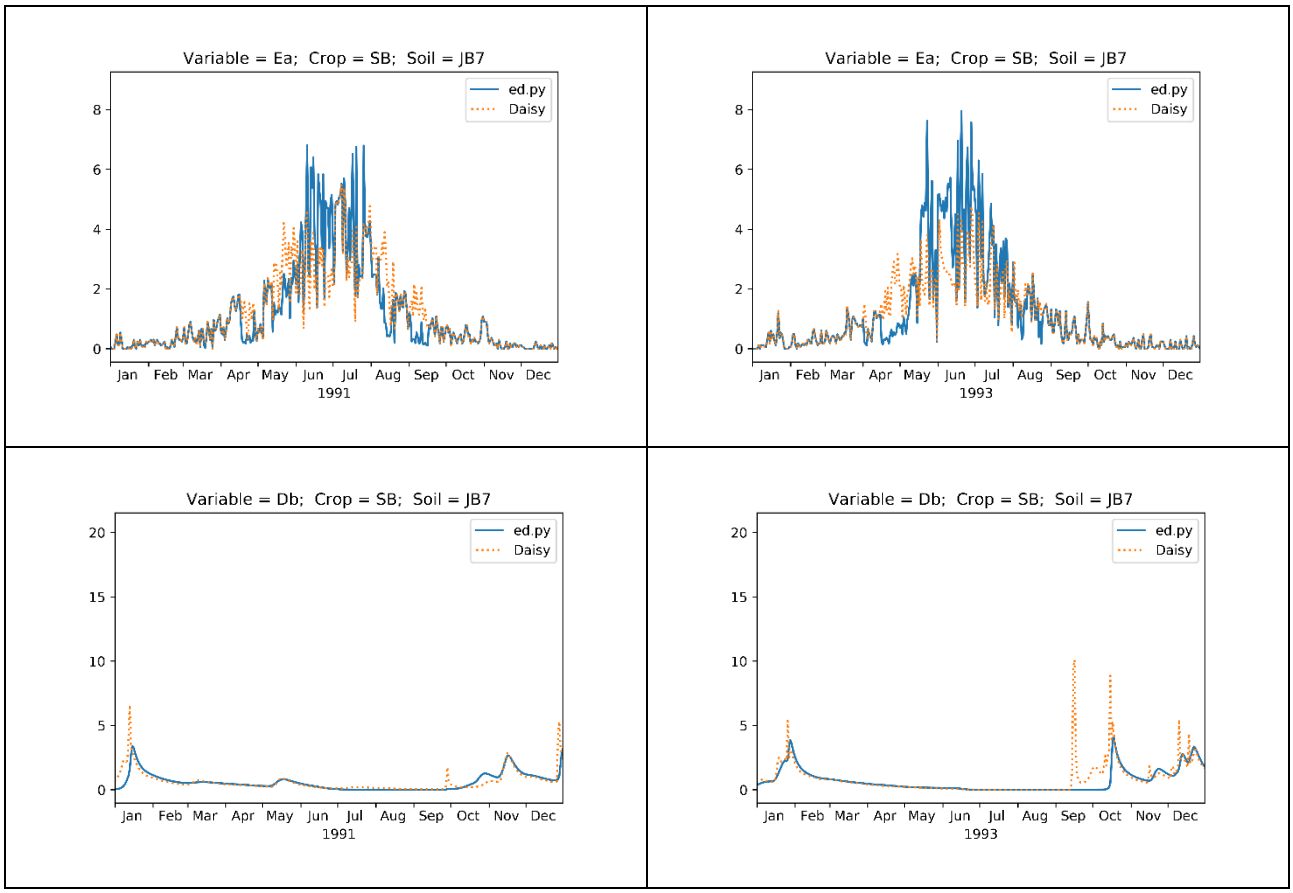

Figure B.15 Actual evapotranspiration, E_a, and drainage from subzone, D_b, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for spring barley (SB) growing on clayey soil (JB7).

Figure B.15 shows simulated actual evapotranspiration and simulated drainage from the subzone for 1991 and 1993. It is noticed that there are periods when there is large difference between E_a simulated by the two codes. Also, notice that Edcrop simulates smoother temporal variation of drainage than Daisy, and that there are periods when Daisy simulates drainage but Edcrop does not. The latter was also mentioned in the comments to Figure B.8.

For the simulation period 1990 to 1999, the lowest yearly E_a simulated by Daisy is 371 mm, which is for 1992. For this year, Edcrop simulates 435 mm. Figure B.16 shows the daily precipitation for this year together with variables simulated by Daisy and Edcrop. It is noticed that almost no precipitation falls from May 14th to July 12th.

During the entire year, the daily potential evapotranspiration simulated by Daisy and Edcrop are nearly identical (Figure B.16) because their simulated plant growth and crop coefficient are very similar (not shown). The simulated rooting is fully developed by the end of June.

With respect to daily actual evapotranspiration during 1992, Figure B.16 shows that the simulation of the two codes become very different from the time of the peak value in beginning of June until it begins to rain again in mid-July. For this period, the Daisy simulation is remarkably lower than the Edcrop simulation:

Daisy-E_a drops quickly from the peak value, stays low, and then quickly rises again when it begins to rain; strangely, during this period there is very little sign in Daisy simulated E_a from the fluctuations in potential evapotranspiration. The Edcrop simulation during the same period, on the other hand, shows a general decrease with fluctuations that correlate with the fluctuations in potential evapotranspiration. The simulated water content within the entire soil profile is similar between the two codes until a week into June, after which Edcrop soil water falls below that of Daisy (Figure B.16). Even though soil water is not the same as plant-available water, because the roots are still growing, this indicates that plant-available water simulated by the two codes are likely to be comparable between the two codes. It therefore remains an open question exactly why there is this difference between Daisy and Edcrop simulation of actual evapotranspiration for this simulation period.

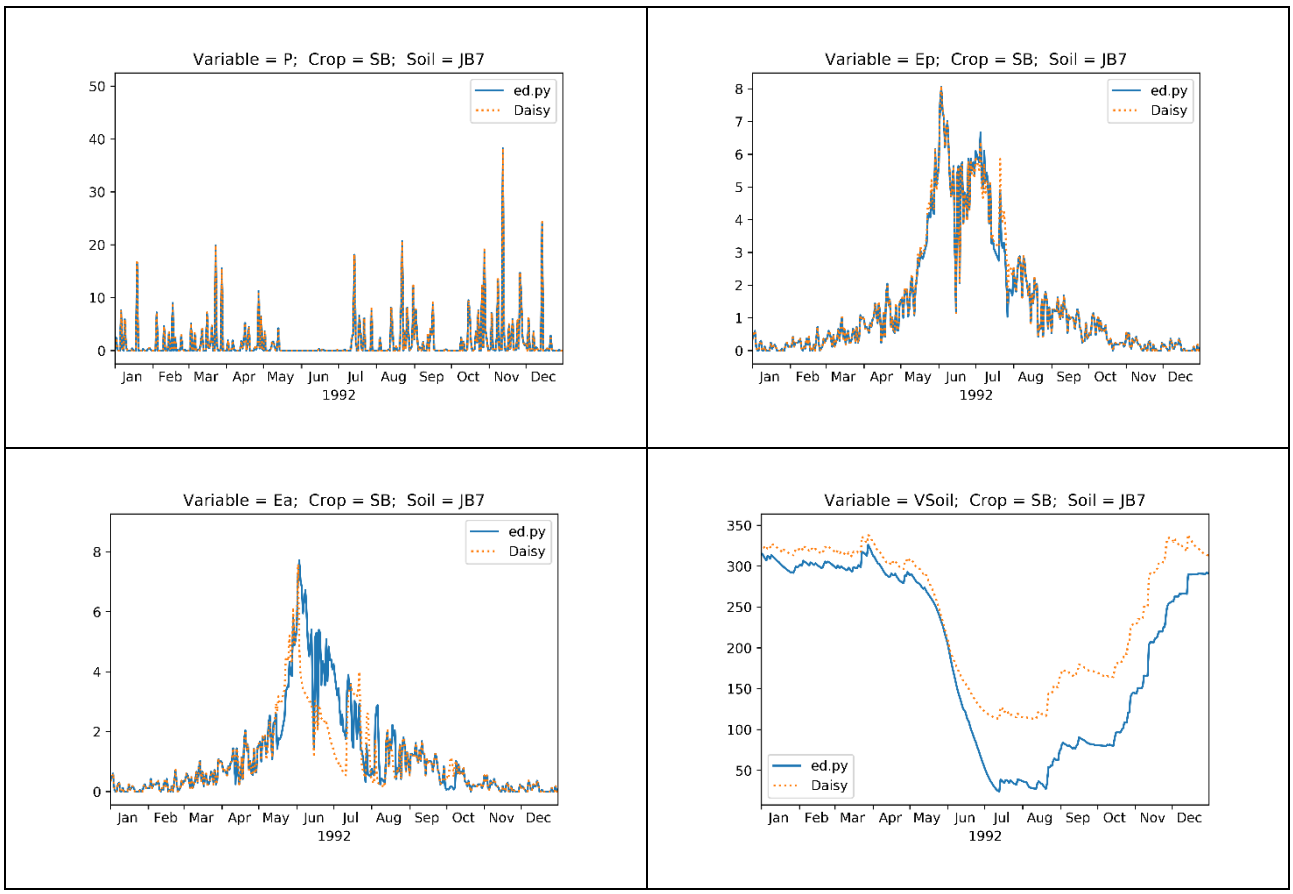

Figure B.16 Precipitation input, P, and potential evapotranspiration, E_p, actual evapotranspiration, E_a, and soil water, V_{soil}, simulated by Daisy and Edcrop (named ed.py in plot), respectively, for spring barley (SB) growing on clayey soil (JB7).

B.4. Conclusions

This appendix presents a comparison of results obtained by Edcrop and Daisy, respectively (Hansen et al., 1990). Daisy is an advanced soil-plant-atmosphere system model that simulates crop production, water dynamics, and nitrogen dynamics under various agricultural management practices and strategies. Edcrop is a simpler model simulating field-scale evapotranspiration and drainage from crop, wetland, or forest. The introduction of the appendix mentions the most remarkable differences between Daisy and Edcrop.

The comparison uses ten years of Danish climate data as well as Danish soil and crop-growth parameter values. Both models are used with Mualem – van Genuchten functions and parameters. The used Edcrop parameter values are default values unless explicitly specified. The base example in section B.1, which is most carefully described, is for spring barley (SB) growing on sandy soil (JB1). The second example, in section B.2, is for winter wheat (WW) growing on sandy soil (JB1). The third example, in section B.3, is for spring barley (SB) growing on clayey soil (JB7).

The daily climate data are from Taastrup, Denmark, and cover the years 1990-1999. The year 1993 had a very dry spring with low precipitation in March, April, and May. The low spring precipitation causes low amount of plant available water during April, May, June, and July of 1993, if the soil is sandy (JB1). This will stunt development of leaf area and rooting of a spring crop like SB, here sown in the beginning of April. Stunting is simulated by Daisy but not by Edcrop, because in Edcrop only the temperature sum drives plant development. In Daisy, plant available water is also a driver of growth. This difference makes Edcrop simulate higher potential evapotranspiration than Daisy during May, June and July of 1993. If irrigation had been applied in the simulations, there would probably not be such difference between Daisy and Edcrop results.

A winter crop like WW has a much better developed rooting when reaching the dry spring of 1993, so for WW growing on JB1 soil there is no remarkable difference between evapotranspiration simulated by Daisy and Edcrop, respectively, during this period. It is somewhat similar for SB growing on JB7 soil, because the amount of plant available water in the clayey soil is sufficient to allow full development of the rooting in 1993, which both Daisy and Edcrop simulate.

Anyhow, for SB grown on JB1 soil, excluding year 1993, there appears to be good correspondence between Daisy and Edcrop in their simulation of potential evapotranspiration, actual evapotranspiration, and drainage, respectively.

For WW grown on JB1 soil, there is also good correspondence between Daisy and Edcrop in simulation of potential evapotranspiration, actual evapotranspiration, and drainage. There is some difference between the WW growth model of Daisy and Edcrop during winter, and with respect to time of harvest, but this does not have a remarkable impact on the water balance results.

For SB grown on JB7 soil, there is fair correspondence between Daisy and Edcrop in simulation of potential evapotranspiration, although the former annual average simulation is 3% higher than the latter. There is poorer correspondence between the codes' simulation of actual evapotranspiration: the Edcrop average simulation is 6% higher than the Daisy simulation; the yearly difference tends to decrease with increasing value of simulated yearly E_a. Correspondingly, Edcrop simulates 13% less yearly drainage than Daisy. Also: there are periods when there is large difference between daily E_a simulated by the two codes; Edcrop simulates smoother temporal variation of drainage than Daisy; and there are periods when Daisy simulates drainage but Edcrop does not. Particular focus is put on a dry summer period in June and July of 1992, where Daisy simulates remarkably lower actual evapotranspiration than the Edcrop. The Daisy simulation

shows a sudden drop at the beginning of the period, a sudden rise at the end of the period, and little response to daily variation in potential evapotranspiration in-between. Edcrop simulation responds to the daily variation in potential evapotranspiration also during this period.