Package ‘AquaCropR’

December 14, 2018

Title AquaCrop for R

Version 0.0.0.9000

Maintainer Anyela V Camargo <anyelavcamargo@gmail.com>

Description The AquaCropR package is the R implementation of the AquaCrop crop growth model developed by FAO (http://www.fao.org/aquacrop) to address food security and assess the effect of the environment and management on crop production. AquaCrop simulates the yield response of herbaceous crops to water and is particularly well suited to conditions in which water is a key limiting factor in crop production. The AquaCrop model simulates final crop yield in four steps which consist on the simulation of development of green crop canopy cover, crop transpiration, above-ground biomass, and final crop yield. Temperature and water stresses directly affect one or more of the above processes.
Nutrient deficiencies and salinity effects are simulated indirectly by moderating canopy cover development over the season, and by reducing crop transpiration and the normalized water productivity. The effect of CO2 concentration on biomass is simulated by altering the normalised water productivity.
AquaCrop requires a relatively small number of explicit parameter values such as weather and soil propertiers, and crop management practices.
Crop associated variables are normally known and are available for the majority of crops.

Imports XML, xml2, pracma, kulife, Rdpack, dplyr

Depends R (>= 3.5.1)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

RdMacros Rdpack

R topics documented:

`About` *About how to run the simulation*

Description

About how to run the simulation

Usage

```
About()
```

Format

These are the steps to run your simulation:

First Create all the input files as specified in the ReadFileLocations section.

Second Use the ReadFileLocations() function to load your files into the model (e.g. F <- ReadFileLocations('FileLocation.xml'))

Third Use the Initialise() function to initialise your variables. Refer to the Initialise section to get familiar with the function (e.g. I <- Initialise(F))

Fourth Use the PerformTimeStep() function to perform the simulation. Refer to the PerformTimeStep section to get familiar with the function (e.g. O <- PerformTimeStep(I)). PerformTimeStep will output all the variables create by the model in the form of a list. Refer to OutputParameters for more information about these variables

Examples

```
F <- ReadFileLocations('FileSetup.xml')
I <- Initialise(F)
O <- PerformTimeStep(I)
names(O)



Example of FileSetup.xml file:

<?xml version="1.0"?>
<FileSetup>
<Input>input/</Input>
<WeatherFilename>Weather.csv</WeatherFilename>
<CO2Filename>MaunaLoaCO2.csv</CO2Filename>
<ClockFilename>Clock.xml</ClockFilename>
<CropRotationFilename>CropRotation.xml</CropRotationFilename>
<FieldManagementFilename>FieldManagement.xml</FieldManagementFilename>
<InitialWCFilename>InitialWaterContent.xml</InitialWCFilename>
<GroundwaterFilename>WaterTable.xml</GroundwaterFilename>
<SoilFilename>Soil.xml</SoilFilename>
<CropRotationCalendarFilename>CropRotationCalendar.xml</CropRotationCalendarFilename>
</FileSetup>
```

`AdjustCCx` *Adjust CCx value for changes in CGC due to water stress during the growing season*

Description

Adjust CCx value for changes in CGC due to water stress during the growing season

Usage

```
AdjustCCx(CCprev, CCo, CCx, CGC, CDC, dt, tSum, Crop)
```

Arguments

`CCprev`	Prev Cannopy cover
`CCo`	initial canopy cover at the time of 90% crop emergence
`CCx`	Maximum canopy cover
`CGC`	Canopy growth coefficient
`CDC`	Canopy decline coefficient
`dt`	Delta time
`tSum`	tSum
`Crop`	Parameters for a given crop

Value

with `CCxAdj` for a n time-step.

Examples

```
AdjustCCx(CCprev, CCo, CCx, CGC, CDC, dt, tSum, Crop)
```

`AerationStress` *Calculate aeration stress coefficient*

Description

Calculate aeration stress coefficient

Usage

```
AerationStress(Crop, InitCond, thRZ)
```

Arguments

`Crop`	Parameters for a given crop
`InitCond`	Crop setting initial structure
`thRZ`	aeration stress (root zone)

Value

list with `NewCond` and `Ksa` aeration stress coefficient for a n time-step.

Examples

```
AerationStress(Crop, InitCond, thRZ)
```

`as_date` *Number of days from sdat to origin*

Description

Number of days from sdat to origin

Usage

```
as_date(sdat)
```

Arguments

`sdat`	date in format "%Y-%m-%d"

as_datenum — *Number of days from sdat to origin*

Description

Number of days from sdat to origin

Usage

```
as_datenum(sdat, o = "0000-01-01", f = "%Y-%m-%d")
```

Arguments

sdat	date in format "%Y-%m-%d" as default
o	origin
f	format

as_datenum_string — *Number of days from sdat to origin*

Description

Number of days from sdat to origin

Usage

```
as_datenum_string(x)
```

Arguments

x	date where x should be in the format: x['Year'], x['Month'], x['Day']

as_date_list — *It's datavec in Matlab. Converts data in numeric format to date in vector format i.e. yyyy, m, d*

Description

It's datavec in Matlab. Converts data in numeric format to date in vector format i.e. yyyy, m, d

Usage

```
as_date_list(sdat, origin_value = "0000-01-01")
```

BiomassAccumulation — *Calculate biomass accumulation (g m-2)*

Description

Calculate biomass accumulation (g m-2)

Usage

```
BiomassAccumulation(Crop, InitCond, Tr, TrPot, Et0, Tmax, Tmin, GDD,
  GrowingSeason)
```

Arguments

`Crop`	Parameters for a given crop
`InitCond`	Crop setting initial structure
`Tr`	actual transpiration
`TrPot`	potential transpiration
`Et0`	Evapotranspiration
`Tmax`	max temp for n time-step
`Tmin`	min temp for n time-step
`GDD`	Growing degree days
`GrowingSeason`	crop developmental stage

Value

`NewCond` for a n time-step.

Examples

```
BiomassAccumulation(Crop, InitCond, Tr, TrPot, Et0, Tmax, Tmin, GDD, GrowingSeason)
```

calcAVP — *calculate Water vapour pressure (Kpa) according to FAO*

Description

calculate Water vapour pressure (Kpa) according to FAO

Usage

```
calcAVP(x)
```

Arguments

`x`	list with three parameters x[[1]] = RH, x[[2]] = TMAX, x[[3]] = TMIN
`AVP`	actual vapour pressure

Examples

```
RH = 57.8
TMAX = 22.96
TMIN = 9.23
calcAVP(RH, TMAX, TMIN)
```

CalculateHIGC *Calculate harvest index growth coefficient.*

Description

Calculate harvest index growth coefficient.

Usage

```
CalculateHIGC(Crop)
```

Arguments

Crop	list

Value

list with `HIGC` and `tHI`.

Examples

```
CalculateHIGC(Crop)
```

CalculateHILinear *Calculate time to switch to linear harvest index build-up, and associated linear rate of build-up. Only for fruit/grain crops.*

Description

Calculate time to switch to linear harvest index build-up, and associated linear rate of build-up. Only for fruit/grain crops.

Usage

```
CalculateHILinear(Crop)
```

Arguments

Crop	list

Value

list with `tSwitch` and `dHILin`.

Examples

```
CalculateHILinear(Crop)
```

calculate_sowingdate *calculate sowingdate*

Description

calculate sowingdate

Usage

```
calculate_sowingdate(weather_data, start_date, end_date, thr = 5, day,
  month, year, P, year_list, Crop, ccl = 100)
```

Arguments

weather_data	weather data
start_date	start window 'dd/mm'
day	label in dataset
month	label in dataset
year	label in dataset
P	label irrigation
year_list	year to be analysed
Crop	crop name
ccl	crop calendar length

CanopyCover *Simulate canopy growth/decline*

Description

Simulate canopy growth/decline

Usage

```
CanopyCover(Crop, Soil, InitCond, GDD, Et0, GrowingSeason)
```

Arguments

Crop	Parameters for a given crop
Soil	properties of soil
InitCond	Crop setting initial structure
GDD	Growing degree days
Et0	Evapotranspiration
GrowingSeason	crop developmental stage

Value

NewCond for a n time-step.

Examples

```
CanopyCover(Crop, Soil, InitCond, GDD, Et0, GrowingSeason)
```

CapillaryRise — *Calculate capillary rise from a shallow groundwater table*

Description

Calculate capillary rise from a shallow groundwater table

Usage

```
CapillaryRise(Soil, Groundwater, InitCond, FluxOut)
```

Arguments

Soil	properties of soil
Groundwater	ground water table
InitCond	Crop setting initial structure
FluxOut	Flux

Value

list with NewCond and CrTot for a n time-step.

Examples

```
CapillaryRise(Soil, Groundwater, InitCond, FluxOut)
```

CCDevelopment — *Calculate canopy cover development by end of the current simulation day*

Description

Calculate canopy cover development by end of the current simulation day

Usage

```
CCDevelopment(CCo, CCx, CGC, CDC, dt, Mode)
```

Arguments

`CCo`	CCDevelopment
`CCx`	Maximum canopy cover
`CGC`	Canopy growth coefficient
`CDC`	Canopy decline coefficient
`dt`	Canopy approaching maximum size
`Mode`	crop calendar mode

Value

`CC` canopy for a n time-step.

Examples

```
CCDevelopment(CCo, CCx, CGC, CDC, dt, Mode)
```

`CCRequiredTime` *Function to find time required to reach CC at end of previous day, given current CGC or CDC*

Description

Function to find time required to reach CC at end of previous day, given current CGC or CDC

Usage

```
CCRequiredTime(CCprev, CCo, CCx, CGC, CDC, dt, tSum, Mode)
```

Arguments

`CCprev`	Prev Cannopy cover
`CCo`	Initial canopy cover at the time of 90% crop emergence
`CCx`	Maximum canopy cover
`CGC`	Canopy growth coefficient
`CDC`	Canopy decline coefficient
`dt`	delta time
`tSum`	tSum
`Mode`	Stage

Value

`tReq` eequired time for a n time-step.

Examples

```
CCRequiredTime(CCprev, CCo, CCx, CGC, CDC, dt, tSum, Mode)
```

`ChangeDateFormat` *change date format*

Description

change date format

Usage

```
ChangeDateFormat(x)
```

Arguments

x	date, where year = x[[1]], DOY = x[[2]]

Value

d

Examples

```
year = '2011'
DOY = 1
ChangeDateFormat(year, DOY)
```

`CheckGroundwaterTable` *Check for presence of a groundwater table, and, if present, to adjust compartment water contents and field capacities where necessary*

Description

Check for presence of a groundwater table, and, if present, to adjust compartment water contents and field capacities where necessary

Usage

```
CheckGroundwaterTable(Soil, Groundwater, InitCond, ClockStruct)
```

Arguments

Soil	structure of Soil
Groundwater	ground water table
InitCond	Crop setting initial structure
ClockStruct	Model time settings

Value

NewCond model values for n time-step

Examples

```
CheckGroundwaterTable(Soil, Groundwater, InitCond)
```

`CheckModelTermination` *Function to check and declare model termination*

Description

Function to check and declare model termination

Usage

```
CheckModelTermination(ClockStruct, InitialiseStruct)
```

Arguments

`ClockStruct`	the Clock list
`InitialiseStruct`	the crop initial conditions in list format

Value

`ClockStruct`.

Examples

```
CheckModelTermination(ClockStruct, InitialiseStruct)
```

`check_file_exist` *check file exist and load*

Description

check file exist and load

Usage

```
check_file_exist(filename)
```

Arguments

`filename,`	file name

check_xml_exist *check file exist and load*

Description

check file exist and load

Usage

```
check_xml_exist(filename)
```

Arguments

`filename,`	file name

check_xml_table_exist *check file exist and load*

Description

check file exist and load

Usage

```
check_xml_table_exist(filename)
```

Arguments

`filename,`	file name

ComputeCropCalendar *Compute additional parameters needed to define crop*

Description

Compute additional parameters needed to define crop

Usage

```
ComputeCropCalendar(Crop, CropName, CropChoices, Weather, ClockStruct)
```

Arguments

`Crop`	list
`CropName`	list with crops names
`CropChoices`	crops to be analysed
`Weather`	dataset with weather data
`ClockStruct`	crop calendar

Value

Crop.

Examples

```
ComputeCropCalendar(Crop, CropName, CropChoices, Weather, ClockStruct)
```

ComputeVariables — *Compute additional variables needed to run AquaCrop*

Description

Compute additional variables needed to run AquaCrop

Usage

```
ComputeVariables(ParamStruct, Weather, ClockStruct, GwStruct, CropChoices,
  FileLocation)
```

Arguments

ParamStruct	Crop Structure
Weather	Weather data
ClockStruct	Crop calendar
GwStruct	Ground water table
CropChoices	Crops to be analysed
FileLocation	list with file locations

Value

ParamStruct.

Examples

```
ComputeVariables(ParamStruct, Weather, ClockStruct, GwStruct, CropChoices, FileLocation)
```

---

conver2num — *Convert to numeric*

Description

Convert to numeric

Usage

```
conver2num(sdata)
```

Arguments

sdata	dataset

convertDOY — *convert to DOY*

Description

convert to DOY

Usage

```
convertDOY(fdate)
```

Arguments

`fdate`	date

Value

DOY

Examples

```
convertDOY('01-01-2000')
```

convert_list_2numeric — *convert list values to numeric*

Description

convert list values to numeric

Usage

```
convert_list_2numeric(slist, lnames = NULL)
```

Arguments

`slist`	list
`lnames`	field names to transform FIXME: do it for subsets

convert_table_list_2numeric
convert list values to numeric

Description

convert list values to numeric

Usage

```
convert_table_list_2numeric(slist, lnames = NULL)
```

Arguments

`slist`	list
`lnames`	field names to transform FIXME: do it for subsets

Convert_to_List
convert data.frame to list

Description

convert data.frame to list

Usage

```
Convert_to_List(sdat)
```

Arguments

`sdat`	data.frame

CropParameters
Crop Parameters to be provided in CropFile.

Description

Crop Parameters to be provided in CropFile.

Usage

```
CropParameters()
```

References

There are no references for Rd macro \insertAllCites on this help page.

Drainage *Redistribute stored soil water*

Description

Redistribute stored soil water

Usage

```
Drainage(Soil, InitCond)
```

Arguments

Soil	structure characteristics
InitCond	Crop setting initial structure

Value

NewCond, DeepPerc and FluxOut for n time-step

Examples

```
Drainage(Soil, InitCond)
```

EvapLayerWaterContent *Get water contents in the evaporation layer*

Description

Get water contents in the evaporation layer

Usage

```
EvapLayerWaterContent(InitCond, Soil, Wevap)
```

Arguments

InitCond	Crop setting initial structure
Soil	properties of soil
wevap	watwe evaporation

Value

Wevap for a n time-step.

Examples

```
EvapLayerWaterContent(InitCond, Soil, Wevap)
```

`export_as_xml` *export dataframe as xml*

Description

export dataframe as xml

Usage

```
export_as_xml(sdata, filename)
```

Arguments

`sdata`	dataset
`filename`	name of file

`ExtractWeatherData` *Extract weather data for current time step.*

Description

Extract weather data for current time step.

Usage

```
ExtractWeatherData(InitialiseStruct)
```

Arguments

`InitialiseStruct`	Crop setting initial structure
`ClockStruct`	crop calendar

Value

list with `Weather` for n time-step

Examples

```
ExtractWeatherData(ClockStruct, InitialiseStruct)
```

`flowerfun` *Flower function*

Description

Flower function

Usage

```
flowerfun(xx)
```

Arguments

`xx`	parameter

`Germination` *Check if crop has germinated*

Description

Check if crop has germinated

Usage

```
Germination(InitCond, Soil, Crop, GDD, GrowingSeason)
```

Arguments

`InitCond`	Crop setting initial structure
`Soil`	properties of soil
`Crop`	Parameters for a given crop
`GDD`	Growing degree days
`GrowingSeason`	crop developmental stage

Value

NewCond for a n time-step.

Examples

```
Germination(InitCond, Soil, Crop, GDD, GrowingSeason)
```

`get_atmospheric_pressure` *Calculate atmospheric pressure (P)*

Description

Calculate atmospheric pressure (P)

Usage

```
get_atmospheric_pressure(z)
```

Arguments

`z`	elevation above sea level (m)

Value

P atmospheric pressure (kPa)

Examples

```
get_atmospheric_pressure(1800)
```

`get_day` *get parameter day from date string*

Description

get parameter day from date string

Usage

```
get_day(x)
```

Arguments

`x`	date, where year = x[[1]], DOY = x[[2]]

Value

day of the month

`get_ea_dp` *Calculate actual vapor pressure (ea) derived from dewpoint temperature*

Description

Calculate actual vapor pressure (ea) derived from dewpoint temperature

Usage

```
get_ea_dp(Tdew)
```

Arguments

Tdew	Dewpoint temperature (oC)

Value

ea

Examples

```
get_ea_dp(14.65)
```

`get_ea_rh` *Calculate actual vapor pressure (ea) derived from mean relative humidity*

Description

Calculate actual vapor pressure (ea) derived from mean relative humidity

Usage

```
get_ea_rh(rh, eoTmax, eoTmin)
```

Arguments

rh	relative humidity (%)
eoTmax	saturation vapour Tmax
eoTmin	saturation vapour Tmin

Value

ea

Examples

```
get_ea_rh(57.44)
```

`get_Eto` *Calculate ETo using Penman_Monteith*

Description

Calculate ETo using Penman_Monteith

Usage

```
get_Eto(x)
```

Arguments

`x`	weather parameters Tmax = x[[1]], Tmax <- x[[1]], Tmin <- x[[2]], RA <- x[[3]], Wind <- x[[4]], Tdew <- x[[5]], altitude <- x[[6]] = max temperature

Examples

```
get_Eto(x)
```

`get_month` *get parameter month from date string*

Description

get parameter month from date string

Usage

```
get_month(x)
```

Arguments

`x`	date, where year = x[[1]], DOY = x[[2]]

Value

month of the year

get_net_radiation *Calculate net radiation mm/day*

Description

Calculate net radiation mm/day

Usage

```
get_net_radiation(Tmax, Tmin, RA, altitude, easqrt)
```

Arguments

Tmax	max temperature
Tmin	min temperature
RA	solar radiation, all Sky Insolation Incident on a Horizontal Surface (MJ/m^2/day)
altitude	(m)
easqrt	sqrt(ea)

Value

rn_mm_day mm/day

Examples

```
get_net_radiation(29.5, 18.88, 27.47, 83.64)
```

get_slope_saturation_vp *Slope of saturation vapour pressure curve*

Description

Slope of saturation vapour pressure curve

Usage

```
get_slope_saturation_vp(Tmean)
```

Arguments

Tmean	mean temperature (oC)

Value

delta Slope of saturation vapour pressure curve T(kPa oC-1)

Examples

```
get_slope_saturation_vp(1800)
```

`get_stefan_Boltzmann` *get stefan Boltzmann temp*

Description

get stefan Boltzmann temp

Usage

```
get_stefan_Boltzmann(t)
```

Arguments

`t`	temperature

Value

temperature examples get_stefan_Boltzmann(14.88)

`get_year` *get parameter year from date string*

Description

get parameter year from date string

Usage

```
get_year(x)
```

Arguments

`x`	date, where year = x[[1]], DOY = x[[2]]

Value

year

`GroundwaterInflow` *Calculate capillary rise in the presence of a shallow groundwater table*

Description

Calculate capillary rise in the presence of a shallow groundwater table

Usage

```
GroundwaterInflow(Soil, InitCond)
```

Arguments

`Soil`	properties of soil
`InitCond`	Crop setting initial structure

Value

list with `NewCond` and `GwIn` groundwater inflow for a n time-step.

Examples

```
GroundwaterInflow(Soil, InitCond)
```

`GrowingDegreeDay` *Calculate number of growing degree days on current day.*

Description

Calculate number of growing degree days on current day.

Usage

```
GrowingDegreeDay(Crop, InitCond, Tmax, Tmin)
```

Arguments

`Crop`	structure
`InitCond`	initial crop settings
`Tmax`	max temperature for n time-step
`Tmin`	min temperature for n time-step

Value

list with `NewCond` and `GDD` for n time-step

Examples

```
GrowingDegreeDay(Crop, InitCond, Tmax, Tmin)
```

GrowthStage — *Calculate number of growing degree days on current day*

Description

Calculate number of growing degree days on current day

Usage

```
GrowthStage(Crop, InitCond, GrowingSeason)
```

Arguments

`Crop`	Parameters for a given crop
`InitCond`	Crop setting initial structure
`GrowingSeason`	crop developmental stage

Value

`NewCond` for a n time-step.

Examples

```
GrowthStage(Crop, InitCond, GrowingSeason)
```

HarvestIndex — *# Function to simulate build up of harvest index*

Description

Function to simulate build up of harvest index

Usage

```
HarvestIndex(Soil, Crop, InitCond, Et0, Tmax, Tmin, GDD, GrowingSeason)
```

Arguments

`Soil`	properties of soil
`Crop`	Parameters for a given crop
`InitCond`	Crop setting initial structure
`Et0`	Evapotranspiration
`Tmax`	max temp for n time-step
`Tmin`	min temp for n time-step
`GDD`	Growing degree days
`GrowingSeason`	crop developmental stage

Value

NewCond for a n time-step.

Examples

```
HarvestIndex(Soil, Crop, InitCond, Et0, Tmax, Tmin, GDD, GrowingSeason)
```

HIadjPollination *Calculate adjustment to harvest index for failure of pollination due to water or temperature stress*

Description

Calculate adjustment to harvest index for failure of pollination due to water or temperature stress

Usage

```
HIadjPollination(InitCond, Crop, Ksw, Kst, HIt)
```

Arguments

InitCond	Crop setting initial structure
Crop	Parameters for a given crop
Ksw	Water stress
Kst	Temperature stress
HIt	Harvest index on current day

Value

with NewCond for a n time-step.

Examples

```
HIadjPollination(InitCond, Crop, Ksw, Kst, HIt)
```

HIadjPostAnthesis *Function to calculate adjustment to harvest index for post-anthesis water stress*

Description

Function to calculate adjustment to harvest index for post-anthesis water stress

Usage

```
HIadjPostAnthesis(InitCond, Crop, Ksw)
```

Arguments

`InitCond`	Crop setting initial structure
`Crop`	Parameters for a given crop
`Ksw`	

Value

`NewCond` for a n time-step.

Examples

```
HIadjPostAnthesis(InitCond, Crop, Ksw)
```

`HIadjPreAnthesis` *Function to calculate adjustment to harvest index for pre-anthesis water stress*

Description

Function to calculate adjustment to harvest index for pre-anthesis water stress

Usage

```
HIadjPreAnthesis(InitCond, Crop)
```

Arguments

`InitCond`	Crop setting initial structure
`Crop`	Parameters for a given crop

Value

`NewCond` for a n time-step.

Examples

```
HIadjPreAnthesis(InitCond, Crop)
```

`HIrefCurrentDay` *Calculate reference (no adjustment for stress effects) harvest index on current day*

Description

Calculate reference (no adjustment for stress effects) harvest index on current day

Usage

```
HIrefCurrentDay(InitCond, Crop, GrowingSeason)
```

Arguments

`InitCond`	Crop setting initial structure
`Crop`	Parameters for a given crop
`GrowingSeason`	crop developmental stage

Value

`NewCond` for a n time-step.

Examples

```
HIrefCurrentDay(InitCond, Crop, GrowingSeason)
```

`Infiltration` *Infiltrate incoming water (rainfall and irrigation)*

Description

Infiltrate incoming water (rainfall and irrigation)

Usage

```
Infiltration(Soil, InitCond, Infl, Irr, IrrMngt, FieldMngt, FluxOut,
  DeepPerc0, Runoff0)
```

Arguments

`Soil`	properties of soil
`InitCond`	Crop setting initial structure
`Infl`	Infiltration
`Irr`	Irrigation
`IrrMngt`	Irrigation management
`FieldMngt`	Field management
`FluxOut`	Flux
`DeepPerc0`	Deep percolation
`Runoff`	water Runoff

Value

list with `NewCond`, `DeepPerc`, `RunoffTot`, `Infl` and `FluxOut` for a n time-step.

Examples

```
Infiltration(Soil, InitCond, Infl, Irr, IrrMngt, FieldMngt, FluxOut, DeepPerc0, Runoff0)
```

`Initialise` *Function to initialise AquaCropR*

Description

Function to initialise AquaCropR

Usage

```
Initialise(FileLocation)
```

Arguments

`filename`	An xml file with locations

Value

list with `FileLocation`.

Examples

```
ReadFileLocations('dummy.xml')
```

`Irrigation` *Get irrigation depth for current day*

Description

Get irrigation depth for current day

Usage

```
Irrigation(InitCond, IrrMngt, Crop, Soil, ClockStruct, GrowingSeason, P,
  Runoff)
```

Arguments

`InitCond`	Crop setting initial structure
`IrrMngt`	Irrigation management
`Crop`	parameters for a given crop
`Soil`	properties of soil
`ClockStruct`	crop calendar
`GrowingSeason`	crop developmental stage
`P`	precipitation
`Runoff`	water Runoff

Value

list with `NewCond` and `Irr` for a n time-step.

Examples

```
Irrigation(InitCond, IrrMngt, Crop, Soil, ClockStruct, GrowingSeason, P, Runoff)
```

`IrrigationManagementParameters`
Irrigation Management Parameters

Description

Irrigation Management Parameters

Usage

```
IrrigationManagementParameters()
```

Format

The irrigation management file defines the input variables controlling irrigation practices in AquaCrop file in xml format should be provided with the following fields:

IrrSchFilename File name of an irrigation schedule. This file will only be used if triggering irrigation based on an input time series. When writing the file, the following information should be provided:

- **day**
- **month**
- **year**
- **irrigationDepth** mm

IrrMethod Method of irrigation, where:

- **0** rainfed
- **1** irrigation based on soil moisture status
- **2** irrigation on a fixed interval
- **3** pre-specified irrigation time-series
- **4** net irrigation

IrrInterval Time interval between irrigation events (days), if triggering based on a fixed interval

SMT1 Percentage of total available water at which irrigation is initiated in the first of the four main crop growth stages, if triggering based on soil moisture status.

SMT2 Percentage of total available water at which irrigation is initiated in the second of the four main crop growth stages, if triggering based on soil moisture status.

SMT3 Percentage of total available water at which irrigation is initiated in the third of the four main crop growth stages, if triggering based on soil moisture status.

SMT4 Percentage of total available water at which irrigation is initiated in the fourth of the four main crop growth stages, if triggering based on soil moisture status.

MaxIrr Maximum irrigation depth (mm day-1).

AppEff Irrigation application efficiency (%).

NetIrrSMT Percentage of total available water to maintain when in net irrigation mode.

WetSurf Soil surface area wetted by irrigation (%).

When multiple crop types are considered in a single simulation, a unique irrigation management file can be created, and is assigned to each crop type in the crop rotation file.

`OutputParameters` *OutputParameters to be provided in at the end of the simulation*

Description

OutputParameters to be provided in at the end of the simulation

Usage

```
OutputParameters()
```

Format

A dataset with the following variables will be provided as output of the AquaCropR simulation:

Water contents output file The water contents list reports the simulated water content (m3 m-3) in each soil compartment at the end of each simulation day. A variable, 'Season', is also reported that takes a value of 1 on days during a growing season, and a value of 0 on days outside a growing season. Note that if the soil water balance is not simulated in the off-season, water contents on these days will be denoted by zero values

Water fluxes output file The water fluxes list provides various simulated water fluxes and states on each simulation day, including:

- **wRZ** Water in the crop root zone (mm)
- **zGW** Water table depth (m). A value of -999 indicates no groundwater table was considered
- **wSurf** Ponded water (mm)
- **Irr** TotIrr (mm)
- **Infl** Infiltration (mm)
- **DP** Deep percolation below the base of the soil profile (mm)
- **CR** Capillary rise in to the soil profile (mm)
- **GWin** Horizontal groundwater inflow to the soil profile (mm)
- **Es** Soil evaporation (mm)
- **EsX** Potential soil evaporation (mm)
- **Tr** Crop transpiration (mm)
- **TrX** Potential crop transpiration (mm)
- **Et0** Reference Evapotranspiration (mm)

Crop growth output file The crop growth output list reports various simulated aspects of crop development on each simulation day, including:

- **GDD** Number of growing degree days on the current day
- **TotGDD** Cumulative growing degree days in the current season
- **RootDepth** Crop effective rooting depth (m)
- **CC** Fractional canopy cover

RefCC Fractional canopy cover under no water-stress conditions

Bio Accumulated aboveground biomass (g m-2)

RefBio Accumulated aboveground biomass under no water stress conditions (g m-2)

HI Fractional reference harvest index

HIadj Fractional harvest index adjusted for water stress effects

Yield Crop yield (tonne ha-1)

PlantingDate Calendar planting date (dd/mm/yyyy)

As previously noted, the variable 'Season' denotes whether a growing season was active on a given day and values of zero are assigned to all fluxes/states outside of the growing season if the off-season soil water balance is not simulated.

`PerformSimulation` *Perform simulation*

Description

Perform simulation

Usage

```
PerformSimulation(InitialiseStruct)
```

Arguments

`InitialiseStruct`
Crop setting initial structure

Value

list with `Outputs` results.

Examples

```
PerfomSimulation(InitialiseStruct)
```

`PreIrrigation` *Calculate pre-irrigation when in net irrigation mode*

Description

Calculate pre-irrigation when in net irrigation mode

Usage

```
PreIrrigation(Soil, Crop, IrrMngt, InitCond)
```

Arguments

`Soil`	structure characteristics
`Crop`	crop settings
`IrrMngt`	irigation management settings
`InitCond`	Crop setting initial structure

Value

list with `NewCond` and `PreIrr` for n time-step

Examples

```
PreIrrigation((Soil, Crop, IrrMngt, InitCond)
```

`RainfallPartition` *Partition rainfall into surface runoff and infiltration*

Description

Partition rainfall into surface runoff and infiltration

Usage

```
RainfallPartition(P, Soil, FieldMngt, InitCond)
```

Arguments

`P`	precipitation
`Soil`	structure characteristics
`InitCond`	Crop setting initial structure
`IrrMngt`	Irrigation management

Value

list with `NewCond`, `Runoff` and `Infl` for n time-step

Examples

```
RainfallPartition(P, Soil, FieldMngt, InitCond)
```

`ReadClockParameters` *Read input files and initialise model clock parameters*

Description

Read input files and initialise model clock parameters

Usage

```
ReadClockParameters(FileLocation)
```

Arguments

`FileLocation`	list with file locations

Value

`ClockStruct` crop calendar.

Examples

```
ReadClockParameters(FileLocation)
```

`ReadFieldManagement` *Read input files and initialise model clock parameters*

Description

Read input files and initialise model clock parameters

Usage

```
ReadFieldManagement(FileLocation)
```

Arguments

`FileLocation`	list with file locations

Value

`FieldManagement` list.

Examples

```
ReadFieldManagement(FileLocation)
```

`ReadFileLocations` *Read files input and output file locations*

Description

Read files input and output file locations

Usage

```
ReadFileLocations(filename)
```

Arguments

`filename`	An xml file with locations

Format

An xml file should be provided with the following fields

Input Files location

ClockFilename Name of clock file (xml format) which is used to set the duration of the simulation. When writing the file, the following information should be provided:

- **SimulationStartTime** Time when the simulation starts (yyyy-mm-dd)
- **SimulationEndTime** Time when the simulation ends (yyyy-mm-dd)
- **OffSeason** Specifies whether (as 'Y' or 'N') Soil water balance is simulated outside the growing season

WeatherFilename Name of weather input file (csv format) which defines time-series of daily weather inputs. When writing the file, the following information should be provided in comma separated format:

- **day** day
- **month** month
- **year** year
- **mintp** Day Minimum Temperature (oC)
- **mxtp** Day Maximum Temperature (oC)
- **p** Daily precipitation (mm)
- **evp** Daily reference evapotranspiration (mm)

CO2Filename Name of Annual CO2 file (csv format) which contains time-series of atmospheric carbon dioxide (CO2) concentrations. When writing the file, the following information should be provided in comma separated format:

- **year** year
- **co2** CO2 concentation (ppm)

SoilFilename Name of Soil file (xml format) which defines the input variables needed to parameterise the soil components in the model. When writing the file, the following file's names, as well as the variables listed in the SoilParameters section, should be provided.

- **SoilProfileFilename** Name of Soil profile name (xml format) which defines the discretisation of the soil profile in to compartments and layers. When writing the file, the following information should be provided.

CompartmentNo Soil compartment number

Thickness Compartment thickness (m)

LayerNo Associated soil layer number

SoilTextureFilename Name of Soil Texture file (xml format) which calculates soil hydraulic properties from textural properties. If the user specifies in the soil input file to calculate soil hydraulic properties from textural properties, the soil texture input file must be provided. The soil texture input file defines the textural properties of each soil layer, which are then assigned to individual soil compartments according to the discretisation of the soil profile. When writing the file, the following information should be provided:

LayerNo Soil layer number

Thickness Thickness of the layer (m)

Sand Sand content (%)

Clay Clay content (%)

OrgMat t (% by weight)

DensityFactor default value of 1

Note. The number of data rows should match the number of soil layers specified in the soil input file. Using these input values, AquaCrop calculates the hydraulic properties of each soil layer (water contents at saturation, field capacity, and permanent wilting, along with the saturated hydraulic conductivity) based on the pedotransfer function model developed by .

SoilHydrologyFilename Name of FieldManagement file (xml format). If the user specifies in the soil input file that soil hydraulic properties are pre-defined, a soil hydrology input file must be provided. When writing the file, the following information should be provided:

LayerNo Soil layer number

LayerThickness Thickness of the layer (m)

thS Water content at saturation (m3 m-3)

thF Water content at field capacity (m3 m-3)

thWP Water content at permanent wilting point (m3 m-3)

Ksat Saturated hydraulic conductivity (mm day-1)

Note: The number of data rows should equal exactly the number of soil layers specified in thesoil input file.

CropRotationFilename Name of Crop Rotation file (xml format) which defines the crop types and any specified rotation to be simulated. When writing the file, the following information should be provided:

NumberofCropOptions Number of crop types modelled

SpecifiedPlantingCalendar If a crop rotation calendar is specified, denoted by a 'Y' or 'N' character

CropInfo Information about each crop should be provided through these variables:

croptype Crop name

CropFilename Crop input filename in xml format. The crop file defines the input variables needed to parameterise the crop component of AquaCrop. A unique version of the crop input file should be created for each individual crop type modelled during the simulation period, as defined by the variables **NumberofCropOptions** and **CropRotationCalendarFilename**. When writing the file, the variables listed in CropParameters should be provided

IrrigationFilename Irrigation management input filename in xml format. When writing the file, the variables listed in the IrrigationManagementParameters section should be provided.

A rotation calendar must be specified if more than one crop type is considered. When a rotation calendar is specified, the planting and latest harvest dates specified in each crop input file will be overwritten by the values given in the rotation calendar

FieldManagementFilename Name of FieldManagement file (xml format) which defines the input variables controlling field management practices in the model. When writing the file, the following information should be provided

Mulches If the soil surface is covered by mulches, where '0' is 'No' and '1' is 'Yes'.

MulchPctGS Soil surface area (%) covered by any mulches during the growing season.

MulchPctOS Soil surface area (%) covered by any mulches during the off season.

fMulch Factor defining the proportional reduction of soil evaporation due to presence of mulches.

Bunds If soil bunds are present on the field, where '0' is 'No' and '1' is 'Yes'.

zBund Height of any soil bunds (m).

BundWater Initial depth of water between any soil bunds (mm).

GroundwaterFilename Name of Groundwater file (xml format) which defines any shallow water table conditions that may influence soil moisture levels

Watertable_present If water table is present, denoted by a 'Y' or 'N' character

Watertable_method If a water table is present, whether it is 'Constant' or 'Variable'.

observations Observations of groundwater, where each observation contains two variables:

date (dd/mm/yyyy)

depth Water table depth (m)

InitialWCFilename Name of Initial Water Content file (xml format) which defines the initial moisture conditions throughout the soil profile at the start of the simulation, and also at the beginning of each growing season if the model does not simulate the soil water balance in the off-season. When writing the file, the following information should be provided:

type_of_value Format in which soil moisture input data is provided. Options available are to specify values based on:

Prop soil hydraulic properties

Num numerical values

Pct percentages of total available water

method Method used to calculate compartment water contents. If method is depth-based ('Depth'), observations will be linearly interpolated to the centre of each compartment. Alternatively, a layer-based method ('Layer') will apply uniform values to all compartments within a soil layer

number_of_input_points Number of input soil moisture data points

input_data_points Soil moisture data observations, where each row contains two variables:

depth_layer Point depth (m) or layer number

value Soil moisture value (defined format). If the format of soil moisture values is:

Prop values must be specified as either 'SAT' (Saturation), 'FC' (Field capacity), or 'WP' (Wilting point)

Num values must have units of m3 m-3

Pct values must have units of %

CropRotationCalendarFilename Name of Crop Rotation file (xml format). If the user specifies multiple crop types in the CropRotationFile or wishes to consider variable planting dates over a multi-season simulation, a crop rotation input file must be provided that defines the time series of growing seasons that will be simulated by AquaCrop. When writing the file, the following information should be provided:

PlantDate Planting date (dd/mm/yyyy)

HarvestDate Latest possible harvest date (dd/mm/yyyy)

Crop Crop type

Data only need to be specified for days when irrigation occurs. AquaCrop will automatically apply zero irrigation values to all other simulation days.

Value

list with `FileLocation`

References

Adummy A (2018). "Not avalable." Failed to insert reference with keys = Saxton2006 from package = 'AquaCropR'. Possible cause — missing REFERENCES.bib in package 'AquaCropR' or 'AquaCropR' not installed.

Examples

```
ReadFileLocations('dummy.xml')
```

`ReadGroundwaterTable` *Read input file and initialise groundwater table parameters*

Description

Read input file and initialise groundwater table parameters

Usage

```
ReadGroundwaterTable(FileLocation, ClockStruct)
```

Arguments

`FileLocation`	list with file locations
`ClockStruct`	crop calendar

Value

`GwStruct` water table.

Examples

```
ReadGroundwaterTable(FileLocation, ClockStruct)
FIXME there is abug here
```

ReadIrrigationManagement

Compute additional variables needed to run AOS

Description

Compute additional variables needed to run AOS

Usage

```
ReadIrrigationManagement(ParamStruct, FileLocation, ClockStruct)
```

Arguments

ParamStruct	Crop parameters structure
FileLocation	list with file locations
ClockStruct	crop calendar

Value

ParamStruct.

Examples

```
ReadIrrigationManagement(ParamStruct, FileLocation, ClockStruct)
```

ReadModelInitialConditions

Set up initial model conditions

Description

Set up initial model conditions

Usage

```
ReadModelInitialConditions(ParamStruct, GwStruct, FieldMngtStruct,
  CropChoices, FileLocation, ClockStruct)
```

Arguments

ParamStruct	Crop details
GwStruct	Water table details
FieldMngtStruct	field management structure
CropChoices	crops whose performance is to be modelled
FileLocation	file locations
ClockStruct	crop calendar structure

Value

`InitCondStruct` with intial conditions.

Examples

```
ReadModelInitialConditions(ParamStruct, GwStruct, FieldMngtStruct, CropChoices, FileLocation,
ClockStruct)
```

`ReadModelParameters` *Read input files and initialise soil and crop parameters*

Description

Read input files and initialise soil and crop parameters

Usage

```
ReadModelParameters(FileLocation, ClockStruct)
```

Arguments

`FileLocation`	list with file locations
`ClockStruct`	crop calendar

Value

list with uodated `ClockStruct` `ParamStruct` and `CropChoices`.

Examples

```
ReadModelParameters(FileLocation, ClockStruct)
```

`ReadWeatherInputs` *Read and process input weather time-series*

Description

Read and process input weather time-series

Usage

```
ReadWeatherInputs(FileLocation, ClockStruct)
```

Arguments

`FileLocation`	file locations
`ClockStruct`	list time

Value

list with `WeatherDB`.

Examples

```
ReadWeatherInputs(FileLocation, ClockStruct)
```

`ResetInitialConditions` *Reset initial model conditions for start of growing*

Description

Reset initial model conditions for start of growing

Usage

```
ResetInitialConditions(InitialiseStruct, ClockStruct)
```

Arguments

`InitialiseStruct`	Crop setting initial structure
`ClockStruct`	crop calendar

Value

list with `InitialiseStruct` and `ClockStruct` for a n time-step

Examples

```
ResetInitialConditions(InitialiseStruct, ClockStruct)
```

`RootDevelopment` *Calculate root zone expansion*

Description

Calculate root zone expansion

Usage

```
RootDevelopment(Crop, Soil, GroundWater, InitCond, GDD, GrowingSeason)
```

Arguments

`Crop`	Parameters for a given crop
`Soil`	properties of soil
`InitCond`	Crop setting initial structure
`GDD`	Growing degree days
`GrowingSeason`	crop developmental stage
`Groundwater`	ground water table

Value

`NewCond` for a n time-step.

Examples

```
RootDevelopment(Crop, Soil, GroundWater, InitCond, GDD, GrowingSeason)
```

`RootZoneWater` *Calculate actual and total available water in the root*

Description

Calculate actual and total available water in the root

Usage

```
RootZoneWater(Soil, Crop, InitCond)
```

Arguments

`Soil`	properties of soil
`Crop`	parameters for a given crop
`InitCond`	Crop setting initial structure

Value

list with `Wr`, `Dr`, `TAW` and `thRZ` for a n time-step.

Examples

```
RootZoneWater(Soil, Crop, InitCond)
```

`SoilEvaporation` *Calculate daily soil evaporation in AOS*

Description

Calculate daily soil evaporation in AOS

Usage

```
SoilEvaporation(ClockStruct, Soil, Crop, IrrMngt, FieldMngt, InitCond, Et0,
  Infl, Rain, Irr, GrowingSeason)
```

Arguments

`ClockStruct`	crop calendar
`Soil`	properties of soil
`Crop`	Parameters for a given crop
`IrrMngt`	Irrigation management
`FieldMngt`	Field management
`InitCond`	Crop setting initial structure
`Et0`	Evapotranspiration
`Infl`	Infiltration
`Irr`	Irrigation
`GrowingSeason`	crop developmental stage
`P`	precipitation

Value

list with `NewCond`, `EsAct` actual evaporation and `EsPot` potential soil evaporation for a n time-step.

Examples

```
SoilEvaporation(ClockStruct, Soil, Crop, IrrMngt, FieldMngt,InitCond,Et0,Infl,Rain,Irr,GrowingSeason)
```

`SoilHydraulicProperties` *Calculate soil hydraulic properties, given textural inputs. Calculations use pedotransfer function equations described in Saxton and Rawls (2006)*

Description

Calculate soil hydraulic properties, given textural inputs. Calculations use pedotransfer function equations described in Saxton and Rawls (2006)

Usage

```
SoilHydraulicProperties(Soil)
```

Arguments

Soil   properties of soil #' @return list with thdry, thwp, hfc, ths and ksa for a n time-step.

Examples

```
SoilHydraulicProperties(Soil)
```

SoilParameters   *Soil Parameters to be provided in SoilFile*

Description

Soil Parameters to be provided in SoilFile

Usage

```
SoilParameters()
```

Format

An .csv file should be provided with the following fields

CalcSHP Determines if soil hydraulic properties are calculated from textural characteristics of soil layers Units:0 = No; 1 = Yes Default value: -

zSoil Total thickness of the soil profile Units:Metres Default value: -

nComp Number of soil compartments Units:- Default value: -

nLayer Number of soil layers Units:- Default value: -

EvapZsurf Thickness of the evaporating soil surface layer in direct contact with the atmosphere Units:Metres Default value: 0.04

EvapZmin Minimum thickness of full soil surface evaporation layer Units:Metres Default value: 0.15

EvapZmax Maximum thickness of full soil surface evaporation layer Units:Metres Default value: 0.3

Kex Maximum soil evaporation coefficient Units:- Default value: 1.1

fevap Shape factor describing the reduction in evaporation with decreasing water content in the soil surface layer Units:- Default value: 4

fWrelExp Relative water content of the soil surface layer at which the evaporation layer depth expands Units:- Default value: 0.4

fwcc Coefficient expressing the reduction in soil evaporation due to the sheltering effect of withered canopy cover Units:- Default value: 0.5

AdjREW Determines if the calculated value of readily evaporable water (REW) is overwritten by a user-defined value Units:0 = No, 1 = Yes Default value:

REW User-defined REW depth (only used if adjusting from the calculated value) Units:Millimetres Default value: -

AdjCN Determines if the curve number (CN) is adjusted each day based on surface moisture conditions Units:0 = No, 1 = Yes Default value: -

CN Curve number Units:- Default value: -

zCN Thickness of the soil surface layer used to calculate moisture content to adjust the curve number Units:Metres Default value: 0.3

zGerm Thickness of the soil surface layer used to calculate moisture content to determine if germination can occur Units:Metres Default value: 0.3

zRes Depth of any restrictive soil layer that inhibits root deepening Units:Metres Default value: Set to negative value if no restriction is present

fshape_cr Shape factor describing the strength of the effect of any shallow groundwater table on soil water content Units:- Default value: 16

`Solution` *Run AquaCrop for a n time step*

Description

Run AquaCrop for a n time step

Usage

```
Solution(Weather, InitialiseStruct)
```

Arguments

`Weather` weather parameters or n time-step

`InitialiseStruct` Crop setting initial structure

Value

list with `NewCond`, `Outputs` and `ClockStruct` for n time-step

Examples

```
Solution(Weather, ClockStruct, InitialiseStruct)
```

`TemperatureStress` *Calculate temperature stress coefficients*

Description

Calculate temperature stress coefficients

Usage

```
TemperatureStress(Crop, Tmax, Tmin, GDD)
```

Arguments

`Crop`	Parameters for a given crop
`Tmax`	max temp for n time-step
`Tmin`	min temp for n time-step
`GDD`	Growing degree days

Value

`Kst` temperature stress a n time-step.

Examples

```
TemperatureStress(Crop, Tmax, Tmin, GDD)
```

`Transpiration` *Calculate crop transpiration on current day*

Description

Calculate crop transpiration on current day

Usage

```
Transpiration(Soil, Crop, IrrMngt, InitCond, Et0, CO2, GrowingSeason)
```

Arguments

`Soil`	properties of soil
`Crop`	Parameters for a given crop
`IrrMngt`	Irrigation management
`InitCond`	Crop setting initial structure
`Et0`	Evapotranspiration
`CO2`	Infiltration
`GrowingSeason`	crop developmental stage

Value

list with `NewCond`, `TrAct` actual transpiration, `TrPot_NS` potential transpiration rate no water stress, `TrPot0` potential transpiration rate, `NewCond` and `IrrNet` Initialise net irrigation, for a n time-step.

Examples

```
Transpiration(Soil, Crop, IrrMngt, InitCond, Et0, CO2, GrowingSeason)
```

`UpdateCCxCDC` *Update CCx and CDC parameter valyes for rewatering in late season of an early declining canopy*

Description

Update CCx and CDC parameter valyes for rewatering in late season of an early declining canopy

Usage

```
UpdateCCxCDC(CCprev, CDC, CCx, dt)
```

Arguments

`CCprev`	Prev Canopy Cover
`CDC`	Canopy decline coefficient
`CCx`	Maximum canopy cover
`dt`	dt parameter

Value

list with `CCXadj` and `CDCadj` for a n time-step.

Examples

```
UpdateCCxCDC(CCprev,CDC,CCx,dt)
```

`UpdateTime` *Update current time in model*

Description

Update current time in model

Usage

```
UpdateTime(ClockStruct, InitialiseStruct)
```

Arguments

`ClockStruct`	crop calendar
`InitialiseStruct`	Crop setting initial structure

Value

list with `InitialiseStruct` and `ClockStruct` for a n time-step

Examples

```
UpdateTime(ClockStruct, InitialiseStruct)
```

`WaterStress` *Calculate relative root zone water depletion for each stress type*

Description

Calculate relative root zone water depletion for each stress type

Usage

```
WaterStress(Crop, InitCond, Dr, TAW, Et0, beta)
```

Arguments

`Crop`	Parameters for a given crop
`InitCond`	Crop setting initial structure
`Dr`	Root zone water content at air dry
`TAW`	Total available water
`Et0`	Evapotranspiration
`beta`	beta

Value

water stress list with `Exp`, `Sto`, `Sen`, `Pol` and `StoLin` for a n time-step.

Examples

```
WaterStress(Crop,InitCond,Dr,TAW,Et0, beta)
```

Index