Translation of SSM into R

Marie Gosme - 2021-05-11

Objectives

- no dependency on Excel (R is open-source and cross-platform), except for input files that come from the original version (to allow easy re-use of existing inputs and comparisons between versions)... but possibility to enter inputs in another format
- possibility to run from a website
- possibility to run in batch mode, and to run several simulations (i.e. several locations-climates-crop managements) at the same time (this will become useful when we want to spatialize the model, either at a large geographic scale, or to have different zones around the tree in agroforestry).

Desired improvements of the code over the original SSM.xls code

- use the same code for all crops (in excel version, part of the code is repeated 3 times (wheat, maize and legumes), with subtle differences, which makes the code (i) difficult to read and understand, (ii) difficult to maintain (one has to make the same changes in

several places in the code)=> use switches given in the crop parameter file to activate or not crop-specific parts, the rest of the code stays the same
- remove all hard-coded parameters (including species names and stage names) => put them in the appropriate parameter files (crops or GeneralPhysicalParameters)
- use more readable names of variables to facilitate code understanding =>use full names instead of acronyms.
- add modularity (to help code maintenance and future developments) => divide the code into different procedures (put them in different files?) and allow initializing all variables from parameter file, so that any module can be commented out. NOT DONE YET

Architecture of the model

The model is "encapsulated" so that function definitions and model variables are not laying around in the user's workspace. The setup function creates an object (here, named MyModel, but it could be anything else), that contains an instance of the model and that can be manipulated by handler functions (defined in file handlers.R), e.g. functions to set the simulation options, to run the model for n time steps, to plot variables etc... Within the object, there are several "global" variables (named in uppercase), procedures (i.e. R functions that have no arguments and return no value, but access and modify global variables), and functions (which are called by procedures to compute different intermediary variables that are used by procedures). All intermediary variables and state variables, as well as input parameters (e.g. climate variables and crop parameters (which vary according to the crop currently present)), are saved at each time step (the current time step is a data.frame named ALLDAYDATA, which is updated by each procedure, and at the end of the timestep, this data.frame is appended to a list (ALLSIMULATEDDATA) with one element per timestep.

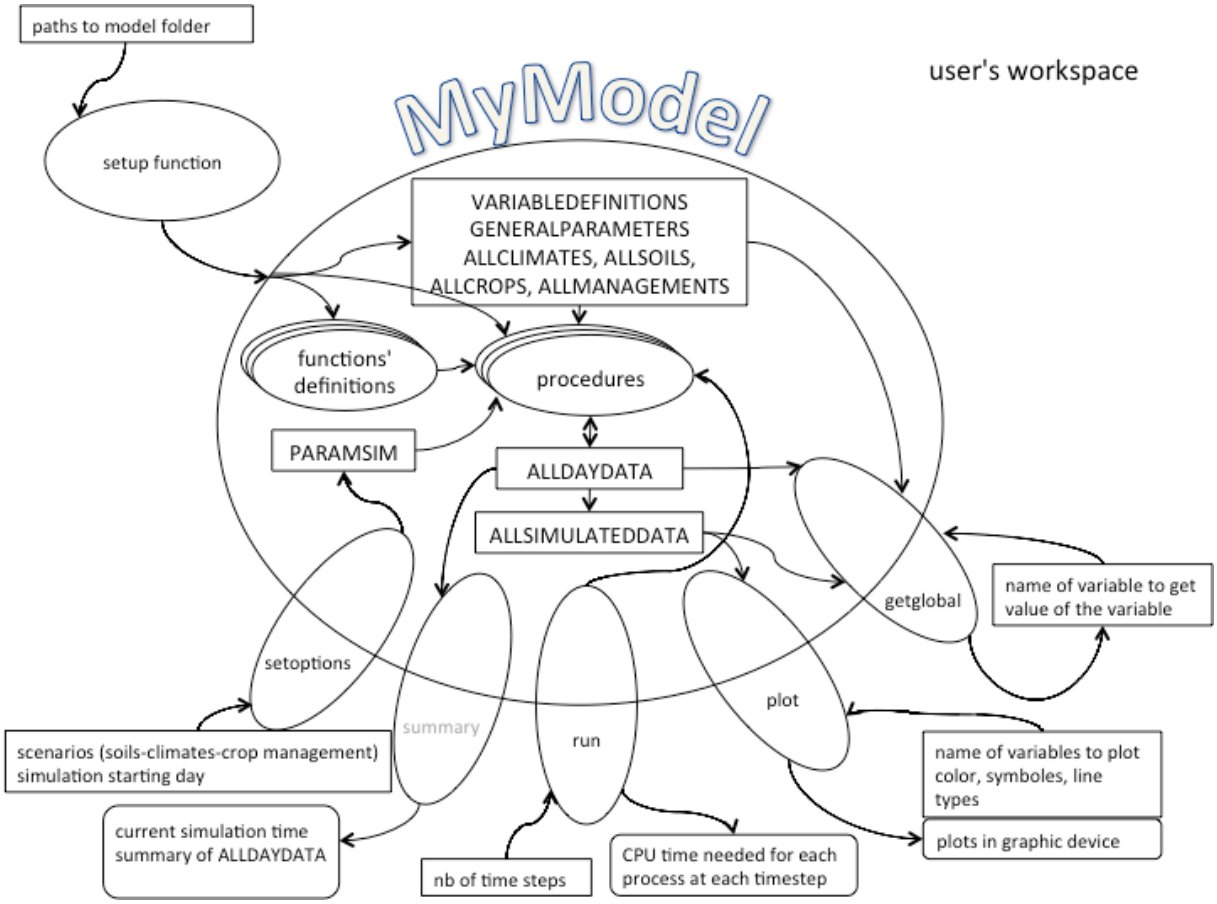


Figure 1: the setup function takes as argument the path to the model folder and returns an R object with its own environment, separated from the user's workspace. This object (here called "MyModel") can be manipulated by handler functions, for example MyModel$run(100) will run the model for 100 timesteps. Ovals: functions; squares: vectors, dataframes or lists; rounded squares: outputs to the console or graphic device

Naming conventions

UPPERCASE: "global" (i.e. within the model) variables, accessible to all procedures
lowercase: local (i.e. within each function or procedure) variable
#variable names
#iXXXX: Input variable (e.g. iPr : Precipitation)
#pXXXX: Parameter (e.g. pPhyllochron)
#cXXXX: Computed variable (e.g. cDeltaLAI): variable that is computed from scratch at each time stp
#sXXXX: State variable (e.g. sLAI): variable that keeps its value from one time step to the beginning of the next
#function names
#fXXXX: Function (e.g. fPhotosynthesis)
#rXXXX: pRocedure (e.g. rUpdateLAI)
#eXXXX: read input scenarios from External files (e.g. eReadInInputs)
#mXXXX: Model interface, i.e. handler functions (e.g. mRun, mPlotDynamics)

Model algorithm

- At the beginning of the simulation, a first element (corresponding to day 0) is created in ALLSIMULATEDDATA, with state, computed and parameter variables initialized at their default initial value found in ALLVARIABLES (read from sheet "savedeachday" of the excel file allvariables.xlsx, which contains the definition,

unit, original SSM name and initial value of all variables, usually NA for computed variables and 0 for state variables, but other values can be used, e.g. for debugging purposes).

- At the beginning of each timestep, a new ALLDAYDATA data.frame is created, with initial values of computed variables (name starting with "c") as specified in ALLVARIABLES and state variables (name starting with s) initialized at their value from the previous time step. Then the procedure (name starting with r) corresponding to each module is called sequentially. Procedure rUpdateManagement sets the crop parameter values at the value corresponding to the different crops present in the different rows of ALLDAYDATA... then subsequent procedures can use these parameters
- In each procedure, local computed variables (name starting with c) are created using different functions (name starting with f), using as arguments either state or computed variables (from ALLDAYDATA). At the end of the procedure, ALLDAYDATA is updated with all state variables (which will be used to initialise their values for the next time step) as well as parameters and computed variables (to allow access to intermediary variables when analysis the model outputs.
- Each function corresponds to the computation of one intermediary variable in the SSM code, we tried to keep the code as similar to the original SSM code as possible... notable deviations from the initial code are given in section "Details on the deviations from the original SSM model, by module", page 13

Structure of the global variables

PARAMSIM

```
List of 8
 $ simustart    : Date[1:1], format: "1997-11-01"
 $ cases        :'data.frame':     2 obs. of  4 variables:
  ..$ climatename: chr [1:2] "Ain Hamra - Meknes" "Ain Hamra - Meknes"
  ..$ soilname   : chr [1:2] "325_-35" "325_-35"
  ..$ lat        : num [1:2] 35 45
  ..$ long       : num [1:2] -5 -5
 $ directory    : chr "/Users/user/Documents/b_maison/congeMat/D4DECLIC/runSSM"
 $ climateformat: chr "standardSSM"
 $ cropformat   : chr "standardSSM"
 $ soilformat   : chr "standardSSM"
 $ managformat  : chr "standardSSM"
 $ Neffect      : logi TRUE
```

simustart is the day when the simulation should start (used to find the climate data for this day) ; cases is a data.frame with n rows (=n cases= n combinations of climate, soil, geographic coordinates (and in the future crop management) ; directory is the path to the folder where your input folder, containing your input (soil, climate, crop parameter, crop management) files are placed (and where the output folder will be created to store graphs or export files) ; climateformat, cropformat, soilformat and managformat are the formats in which these input files are (for now only "standardSSM" is available, it was planned to include other formats to allow automatic link with external climate and soil databases, but not done yet) ; Neffect is a boolean indicating which version of the model to use: with or without N effect on LAI decrease (for now, only the version with N effect is coded).

ALLDAYDATA

'data.frame': 2 obs. of 111 variables:
$ iDate : Date, format: "1997-11-04" "1997-11-04"
$ sAccumulatedLeafDryMatter : num 0 0
$ sAccumulatedStemDryMatter : num 0 0
$ sAccumulatedVegDryMatter : num 0 0
$ sLAI : logi NA NA
$ sMainstemNodeNumber : num 1.72 1.72
$ sPlantdensity : num 280 280
$ sCrop : chr "WHEAT" "WHEAT"
$ sCultivar : chr "durum wheat" "toto"
$ sBiologicalDay : num 2.88 2.88
$ sGrowthStage : chr "germination" "germination"
$ sThermalUnite : num 79.2 79.2
$ sRootFrontDepth : num 0 0
etc... (all variables listed in sheet "savedeachday" of file allvariables.xlsx)
rows are cases (=combinations of climate-soil-management)

ALLSIMULATEDDATA

list of n data.frames (ALLDAYDATA at the end of the time step, one for each timestep (plus day 0)).

VARIABLEDEFINITIONS

this data.frame is read from sheet "savedeachday" in file "allvariables.xlsx" from the model folder.
'data.frame': 111 obs. of 9 variables:
$ name : chr "sAccumulatedLeafDryMatter" "sAccumulatedStemDryMatter" "sAccumulatedVegDryMatter" "sLAI" ...
$ typeinthemodel : chr "stateVariable" "stateVariable" "stateVariable" "stateVariable" ...
$ typeR : chr "numeric" "numeric" "numeric" "numeric" ...
$ module : chr "DMDistribution" "DMDistribution" "DMDistribution" "LAI" ...
$ unit : chr "g m-2 " "g m-2" "g m-2" "m2 m-2" ...
$ definition : chr "Accumulated leaf dry matter " "Accumulated Steam dry matter " "Accumulated vegetative (leaf + stem) dry matter " "Leaf area index " ...
$ translationSSM : chr "WLF" "WSF" "WVEG" "LAI" ...
$ french : chr "Stock de matière sèche dans les feuilles" "Stock de matière sèche dans les parties végétatives" "Stock de matière sèche dans les tiges" "Index de surface foliaire" ...
$ defaultInitialvalue: num 0 0 0 0 0 0 0 0 0 0 ...

ALLCROPS

data.frame with rownames crop.cultivar, and with parameters in columns
It is created by reading sheet allcrops of file crops.xlsx in the input folder of the simulation folder (NOT the model folder, which contains example input files, which are not read). Warning: compared with the sheet Crop in GW_SSM_iCrop_TOTAL_modiflog.xlsm, I had to add parameters TBVER, TP1VER,

TP2VER, TCVER, VDSAT from the excel file where crops are in different sheets, and the parameters for stage durations (and stage names) and filters for case-specific processes. Parameters in the R version have more readable names than in the excel version, and the translation is done thanks to the file allvariables.xlsx, which contains the R names and the SSM names. In the following export, some parameters have their R name and type, and some are still as they were read from the crop parameter file, the later are the ones that are not yet in allvariables.xlsx because they aren't used in the code yet (code not finished). And the filters are not correct yet.

'data.frame': 6 obs. of 85 variables:
$ CROP : chr "WHEAT" "WHEAT" "WHEAT" "Chickpea" ...
$ Cultivar : chr "Ble_Dur_1" "Ble_Tendre_1" "Ble_Tendre_2" "Ghab2" ...
$ pPhyllochron : num 118 110 110 46 0 0
$ pcoefPlantLeafNumberNode : num 1 1 1 1 0 0
$ pExpPlantLeafNumberNode : num 2.5 2.34 2.34 2 0 0
$ a_plapow_d : chr "1" "1" "1" "1" ...
$ b_plapow_d : chr "0" "0" "0" "0" ...
$ pSpecificLeafArea : num 0.02 0.02 0.02 0.0161 0 0
$ pFreezeThresholdTemp : num -5 -5 -5 -5 0 0
$ pFreezeFracLeafDestruction : num 0.01 0.01 0.01 0 0 0
$ pHeatThresholdTemp : num 30 30 30 0 0 0
$ pHeatFracLeafDestruction : num 0.1 0.1 0.1 0 0 0
$ pTbasRUE : num 0 0 0 2 0 0
$ pTopt1RUE : num 15 15 15 14 0 0
$ pTopt2RUE : num 22 22 22 30 0 0
$ plethalRUE : num 35 35 35 38 0 0
$ pKPAR : num 0.65 0.65 0.65 0.65 0 0
$ pRadEffiencyOptimal : num 2.2 2.2 2.2 1.8 0 0
$ CO2RUE : chr "0.8" "0.8" "0.8" "0.8" ...
$ pParCoefLeafAeraLow : num 0.6 0.6 0.6 0.53 0 0
$ pParCoefLeafAeraHigh : num 0.3 0.3 0.3 0.3 0 0
$ pTotalCropMassFromCoefLeafAreaLowHigh : num 160 160 160 180 0 0
$ pParCoefLeafAeraTerminationLeafGrow : num 0.1 0.1 0.1 0.13 0 0
$ pFractionCropMassTranslocateble : num 0.22 0.22 0.22 0.22 0 0
$ pGrainConversionCoefficient : num 1 1 1 1 0 0
$ pPotentialSlopeHarvestIndex : num 0.008 0.01 0.01 0.01 0 0
$ pPotentialSlopeHarvestIndexCriticalPoint1: num 0 0 0 0 0 0
$ pPotentialSlopeHarvestIndexCriticalPoint2: num 600 600 600 0 0 0
$ pPotentialSlopeHarvestIndexCriticalPoint3: num 1200 1200 1200 9999 0 ...
$ pPotentialSlopeHarvestIndexCriticalPoint4: num 3200 3200 3200 9999 0 ...
$ iDEPORT : chr "200" "200" "200" "200" ...
$ pMaxDepthWaterExtraction : num 1000 1000 1000 1000 0 0
$ pPotentialRootGrowth : num 30 30 30 17 0 0
$ pTranspirationEfficiencyLinkedToCO2 : num 5.8 5.8 5.8 5 0 0
$ TEC700 : chr "0" "0" "0" "0" ...
$ WSSG : chr "0.3" "0.3" "0.3" "0.3" ...
$ WSSL : chr "0.5" "0.40000000000000008" "0.40000000000000008" "0.4" ...
$ WSSD : chr "0.4" "0.40000000000000008" "0.40000000000000008" "0" ...

```
 $ WSSN                          : chr  "0" "0" "0" "0.5" ...
 $ FLDKL                         : chr  "20" "20" "20" "20" ...
 $ pTbaseVernalization                : num  -1 -1 -1 NA -1 -1
 $ pTopt1Vernalization                : num  0 0 0 NA 0 0
 $ pTopt2Vernalization                : num  8 8 8 NA 8 8
 $ pTlethalVernalization              : num  12 12 12 NA 12 12
 $ pVDSAT                         : num  50 50 50 NA 50 50
 $ pSpecLeafNGreenLeaf                : num  1.8 1.5 1.5 2.3 0 0
 $ pSpecLeafNSenescenceLeaf           : num  0.4 0.4 0.4 0.68 0 0
 $ SNCG                          : chr  "1.4999999999999999E-2" "1.4999999999999999E-2"
"1.4999999999999999E-2" "1.37E-2" ...
 $ SNCS                          : chr  "5.1000000000000004E-3" "5.0000000000000001E-3"
"5.0000000000000001E-3" "3.8999999999999998E-3" ...
 $ GNCmin                         : chr  "2.0899999999999998E-2" "2.1299999999999999E-
2" "2.1299999999999999E-2" "3.3700000000000001E-2" ...
 $ GNCmax                         : chr  "2.0899999999999998E-2" "2.1299999999999999E-
2" "2.1299999999999999E-2" "3.3700000000000001E-2" ...
 $ MXNUP                         : chr  "0.25" "0.25" "0.25" "0.45" ...
 $ pTbasedev                      : num  0 0 0 2 0 0
 $ pTopt1dev                      : num  27.5 27.5 27.5 21 0 0
 $ pTopt2dev                      : num  27.5 27.5 27.5 30 0 0
 $ pTlethaldev                    : num  40 40 40 40 0 0
 $ pVernalizationSensitivity           : num  0.002 0.008 0.008 NA 0 0
 $ pCriticalPhotoPerdiod              : num  14 21 21 0 0 0
 $ pPhotoPeriodSensitivity            : num  0.17 0 0 0.05 0 0
 $ bdSOWEMR                        : chr  "6" "4" "4" "8.5" ...
 $ bdEMRTIL                       : chr  "5" "9.4925465837500003" "11.6" "36" ...
 $ bdTILSEL                       : chr  "8" "7.7740683243499999" "9.5" "5.7" ...
 $ bdSELBOT                       : chr  "6" "2.5822646555555551" "3.1555555555" "4.3" ...
 $ bdBOTEAR                       : chr  "6" "0.64556843888871" "0.788888888875" "22" ...
 $ bdEARANT                       : chr  "15" "3.155555555555555" "3.1555555555" "7" ...
 $ bdANTPM                        : chr  "43" "40.4" "37.6" "0" ...
 $ bdPMHM                         : chr  "8" "8" "8" "0" ...
 $ bdBRP                         : chr  "0" "0" "0" "8.5" ...
 $ bdTRP                         : chr  "0" "0" "0" "76.5" ...
 $ bdBSG                         : chr  "0" "0" "0" "54.5" ...
 $ bdTSG                         : chr  "0" "0" "0" "76.5" ...
 $ bdTLM                          : chr  "0" "0" "0" "52.5" ...
 $ bdTLP                         : chr  "0" "0" "0" "60.1" ...
 $ bdBLS                         : chr  "0" "0" "0" "54.5" ...
 $ bdBNF                          : chr  "0" "0" "0" "12" ...
 $ thresholds                      :List of 6
  ..$ : Named num  6 5 8 6 6 ...
  .. ..- attr(*, "names")= chr  "germination" "emergence" "tillering" "stemElongation" ...
  ..$ : Named num  4 9.493 7.774 2.582 0.646 ...
  .. ..- attr(*, "names")= chr  "germination" "emergence" "tillering" "stemElongation" ...
  ..$ : Named num  4 11.6 9.5 3.156 0.789 ...
  .. ..- attr(*, "names")= chr  "germination" "emergence" "tillering" "stemElongation" ...
  ..$ : Named num  8.5 36 5.7 4.3 22 ...
```

.. ..- attr(*, "names")= chr "germination" "emergence" "tillering" "stemElongation" ...
..$: Named num 0 0 0 0 0 ...
.. ..- attr(*, "names")= chr "germination" "emergence" "tillering" "stemElongation" ...
..$: Named num 0 0 0 0 0 ...
.. ..- attr(*, "names")= chr "germination" "emergence" "tillering" "stemElongation" ...
$ vernalisation.filter : chr "is.after('emergence', 0) & is.before('stemElongation', 0)" "is.after('emergence', 0) & is.before('stemElongation', 0)" "is.after('emergence', 0) & is.before('stemElongation', 0)" "FALSE" ...
$ photoperiod.filter : chr "FALSE" "FALSE" "FALSE" "FALSE" ...
$ waterstress.filter : chr "is.after('emergence', 0) & is.before('senescence', 10)" "is.after('emergence', 0) & is.before('senescence', 10)" "is.after('emergence', 0) & is.before('senescence', 10)" "is.after('emergence', 0) & is.before('senescence', 10)" ...
$ LAI_Mainstem.filter : chr "is.after('germination', 0) & is.before('Booting', 0)" "is.after('germination', 0) & is.before('Booting', 0)" "is.after('germination', 0) & is.before('Booting', 0)" "is.after('germination', 0) & is.before('Booting', 0)" ...
$ LAI_Secondary.filter : chr "is.after('Booting', 0) & is.before('earing', 5)" "is.after('Booting', 0) & is.before('earing', 5)" "is.after('Booting', 0) & is.before('earing', 5)" "is.after('Booting', 0) & is.before('earing', 5)" ...
$ DMDistribution_SeedGrowing.filter : chr "is.after('anthesis', 5) & is.before('anthesis', 41.5)" "is.after('anthesis', 5) & is.before('anthesis', 38.9)" "is.after('anthesis', 5) & is.before('anthesis', 36.1)" "is.after('anthesis', 5) & is.before('anthesis', -1.5)" ...
$ DMProduction.filter : chr "is.after('germination') & is.before('anthesis', 41.5)" "is.after('germination') & is.before('anthesis', 38.9)" "is.after('germination') & is.before('anthesis', 36.1)" "is.after('germination') & is.before('anthesis', -1.5)" ...
$ rRootDepth.filter : chr "is.after('emergence',0) & is.before('anthesis',5)" "is.after('emergence',0) & is.before('anthesis',5)" "is.after('emergence',0) & is.before('anthesis',5)" "is.after('emergence',0) & is.before('anthesis',5)" ...
$ name : chr "WHEAT.Ble_Dur_1" "WHEAT.Ble_Tendre_1" "WHEAT.Ble_Tendre_2" "Chickpea.Ghab2" ...

ALLCLIMATES

data.frame with one row for each date in each climate. It is created by reading all sheets of file climates.xlsx in the input folder of the simulation folder (NOT the model folder, which contains example input files, which are not read).

'data.frame': 15336 obs. of 8 variables:
$ YEAR : num 1995 1995 1995 1995 1995 ...
$ DOY : num 1 2 3 4 5 6 7 8 9 10 ...
$ iRSDS : num 11.3 9.1 11.7 13.1 7.3 12 11.4 14 13.9 13.5 ...
$ iTASMax : num 21.2 12.8 15 12.8 13.5 15.4 12.8 12.8 14.2 13.5 ...
$ iTASMin : num 7.8 8.2 4.2 5.5 4 7.8 6.4 0.1 -2.3 2.7 ...
$ iPr : num 0 2 0 0 0 0 0 0 0 0 ...
$ date : Date, format: "1995-01-01" "1995-01-02" "1995-01-03" ...
$ climatename: chr "Ain Hamra - Meknes" "Ain Hamra - Meknes" "Ain Hamra - Meknes" "Ain Hamra - Meknes" ...

ALLSOILS

not done yet.

the watermodule is being coded, for now it uses a list named PARAMSSOILS, but it might change in the future

```
List of 7
 $ pNLayer        : num [1:3] 2 2 3
 $ pDrainLayer    : num [1:3] 2 2 3
 $ pSoilAlbedo    : num [1:3] 0.12 0.13 0.15
 $ U              : logi NA
 $ pSoilCurveNumber: num [1:3] 60 70 80
 $ pVPDcoef       : num [1:3] 0.65 0.65 0.65
 $ paramlayers    :List of 3
  ..$ :'data.frame':       2 obs. of  6 variables:
  .. ..$ layer         : int [1:2] 1 2
  .. ..$ pLayerThickness: num [1:2] 300 700
  .. ..$ pSaturation    : num [1:2] 0.36 0.4
  .. ..$ pFieldCapacity : num [1:2] 0.24 0.25
  .. ..$ pWiltingPoint  : num [1:2] 0.1 0.12
  .. ..$ pSoilDryness   : num [1:2] 0.03 0.04
  ..$ :'data.frame':       2 obs. of  6 variables:
  .. ..$ layer         : int [1:2] 1 2
  .. ..$ pLayerThickness: num [1:2] 200 800
  .. ..$ pSaturation    : num [1:2] 0.36 0.4
  .. ..$ pFieldCapacity : num [1:2] 0.24 0.25
  .. ..$ pWiltingPoint  : num [1:2] 0.1 0.14
  .. ..$ pSoilDryness   : num [1:2] 0.04 0.04
  ..$ :'data.frame':       3 obs. of  6 variables:
  .. ..$ layer         : int [1:3] 1 2 3
  .. ..$ pLayerThickness: num [1:3] 300 200 400
  .. ..$ pSaturation    : num [1:3] 0.36 0.4 0.38
  .. ..$ pFieldCapacity : num [1:3] 0.24 0.25 0.22
  .. ..$ pWiltingPoint  : num [1:3] 0.1 0.12 0.09
  .. ..$ pSoilDryness   : num [1:3] 0.03 0.04 0.035
```

ALLMANAGEMENTS

It is created by reading soils.xlsx in the input folder of the simulation folder (NOT the model folder, which contains example input files, which are not read).

```
List of possible crop managements plans (named "row XX" based on the row number of
<--- MangRowNo in the excel file).
List of 3
 $ row 13:List of 11
  ..$ dfCode         :'data.frame':         1 obs. of  2 variables:
  .. ..$ Code       : chr "ROTATION_BLE"
  .. ..$ Description:: num 0
  ..$ dfSowing       :'data.frame':       1 obs. of  11 variables:
  .. ..$ FixFind: num 4
  .. ..$ Fyear  : num 2007
  .. ..$ yrno   : num 1
  .. ..$ SimDoy : num 20
  .. ..$ Fpdoy  : num 288
  .. ..$ Lpdoy  : num 20
  .. ..$ StopDAP: num 280
```

.. ..$ SowTmp : chr "-"
.. ..$ SowWat : num 0.3
.. ..$ Pden : num 280
.. ..$ STBLW : num 0.01
..$ nitrogenScenario: num 2
..$ nitrogenNumber : num 1
..$ nitrogenDatetype: num 1
..$ nitrogendf :'data.frame': 1 obs. of 4 variables:
.. ..$ NapplNumber: num 1
.. ..$ DAPorCBD : num 1
.. ..$ amount : num 5
.. ..$ FracVol : num 7
..$ waterLevel : num 0
..$ waterScenario : num 2
..$ waterNumber : num 0
..$ waterDatetype : num 0
..$ waterdf :'data.frame': 0 obs. of 2 variables:
.. ..$ DAPorCBDorDOY: logi(0)
.. ..$ amount : logi(0)
$ row 36:List of 11
idem
$ row 59:List of 11
idem

GENERALPARAMETERS

this data.frame is read from sheet " generalPhysicalParameters" in file "allvariables.xlsx" from the model folder.
'data.frame': 5 obs. of 9 variables:
$ name : chr "pMinimalSoilEvaporation" "pSoilWettingWaterQuantity" "pCanopyExtinctionCoefficient" "pCropAlbedo" ...
$ typeinthemodel : chr "parameter" "parameter" "parameter" "parameter" ...
$ typeR : chr "numeric" "numeric" "numeric" "numeric" ...
$ module : chr "rWaterBudget" "rWaterBudget" "rWaterBudget" "rWaterBudget" ...
$ unit : chr "mm" "mm" "-" "-" ...
$ definition : chr "Minimal Soil Evaporation" "amount of rainfall and/or irrigation required to wet the top soil layer to return soil evaporation from Stage II to Stage I" "canopy extinction coefficient" "Crop albedo" ...
$ translationSSM : chr "EOSMIN" "WETWAT" "KET" "CALB" ...
$ french : chr "evaporation minimale du sol" NA NA NA ...
$ defaultInitialvalue: num 1.5 10 0.5 0.23 0.48

Structure of the files in the model folder

- allvariables.xlsx : excel file with the name, module where it is used, definition, unit, original SSM name and initial value of all variables
- exampleRunSSM.R : R script giving an example of how to run the model (and the definition of the setup function : this is the script that the user will run in the console

- externalFilesReadingSSM.R : function definitions for reading in external files (does not actually read anything, the reading is done when the handler function mRun() is run for the first time)
- functionsSSM.R : definition of all functions and procedures that are at the core of the SSM model (to be done: split this file by module to facilitate maintenance)
- handlersSSM.R : definition of handler functions (mRun, mPlotDynamics...)
- headersSSM.R : loading of required packages
- HousekeepingFunctionsSSM.R : definitions of functions that are not specific of SSM but are usefull for the model algorithm.
- initialisationEnvironmenVariablesSSM.R : creation of all the encapsulated "global" variables.
- input : folder with example input files
    - climates.xlsx : excel file with one sheet per possible climate (same content as in SSM.xlsx)
    - crops.xlsx : excel file with one sheet per species, one column per cultivar (same content as in SSM.xlsx except extra parameters at the bottom: thresholds for the different stages (obtained with a formula using the bdXXX parameters, warning: for the formula to give R-readable values, the decimal separator in excel must be the point, not coma as is the default in French language setting) and filters (text in R language that will be parsed by the model) for the different modules... see "filters", page 12.
    - managementPlans.xlsx not used yet (but read anyway): excel file with one sheet per crop Management plan (rule for sowing and sowing density, nitrogen and water applications): (same content as in SSM.xlsx).
    - soils.xlsx excel file with one sheet per possible soil (same content as in SSM.xlsx)
- LICENSE: GNU licence, I don't know why it's here
- README.md : file used by github to display on https://github.com/mariegosme/SSM.R
- SSMWaterModule.R : temporary file with the water module partially coded... will eventually be merged into functionsSSM.R (or kept as separate file if modules are separated into different files).

Function setup() sources, locally within itself, the files headersSSM.R, initialisationEnvironmenVariablesSSM.R, externalFilesReadingSSM.R, HousekeepingFunctionsSSM.R, functionsSSM.R, handlersSSM.R in this order.

How to run the model

1) run the function definition of function setup

2) build the model
mymodel<-setup("/Users/user/Documents/b_maison/congeMat/D4DECLIC/SSM/")

3) prepare cases (these will be the rows of ALLSIMULATEDDATA):
mycases<-data.frame(climatename="Ain Hamra - Meknes", soilname="325_-35", lat=c(35, 45), long=-5)
#climatename: one of the sheet names of file climates (if climate in standard SSM format)
#soilname: one of the sheet names of file soils (if in standard SSM format)

```
#to do: define crop rotation and management (once management procedure is completed)
rownames(mycases)<-c("Meknes35degres", "Meknes45degres") #these will be the cases names used in the plots, outputs etc...
4) define the options of the simulation
paramsim<-list(
 simustart=as.Date("1997-11-01"), #date of start of the simulation
 cases=mycases, #cases (e.g. combinations of soil-climate-management-location)
 directory="/Users/user/Documents/b_maison/congeMat/D4DECLIC/runSSM",#folder where your inputs (with climates and soils files) are, and where the output folders will be
 climateformat="standardSSM",
 cropformat="standardSSM",
 soilformat="standardSSM",
 managformat="standardSSM",
 Neffect=FALSE
)

5)use this to set the simulation options
mymodel$setoptions(paramsim)

6)run the model for 4 timesteps
mymodel$run(4)

7)plot the dynamics of some variables
dynamiques<-mymodel$plot(c("iTASMin", "iTASMax", "iRSDS"),
        col=c(iTASMin="blue", iTASMax="red", iRSDS="black"), whatcol="variables",
        lty=c(iTASMin=1, iTASMax=1, iRSDS=2), whatllty="variables",
       pch=c(Meknes35degres=1, Meknes45degres=8), whatpch="cases")
```

Details on the deviations from the original SSM model, by module

all modules

We use "**filters**" to know to which cases to apply time-dependent functions or procedures: in the original SSM code, the conditions were hard-coded for each crop (e.g. for wheat:

```
bdBRV = bdEMR            'Beg. response to vernalization
bdTRV = bdSEL            'End of response to vernalization
If CBD >= bdBRV And CBD <= bdTRV .... do vernalisation calculations
```

In SSM.R, these conditions are part of the crop parameterisation:
vernalisation.filter = "is.after('emergence', 0) & is.before('stemElongation')" which means that vernalisation is applied only to wheat crops that are already emerged but haven't started the stem elongation stage. The syntax of is.before and is.after is the following: is.before(stage, bd=-Inf), which means that if bd is not specified, the stage itself is not included (is.before('tillering') means that it happens strictly before the beginning of tillering stage, while is.before('tillering', 5) means that it happens until the 5th biological day within the tillering stage. For is.after, the syntax is is.after(stage, bd=Inf), which means that if bd is not specified, the stage itself is not included (is.after('tillering') means that it happens strictly after the end of tillering stage, while is.

after ('tillering', 5) means that it happens starting on the 5th biological day within the tillering stage. Note that the stage names are not hard-coded: they come from the names given to the thresholds for each stage provided in the crop parameter file.

climate

icicici: unit problem: we expect mm, but we have degrees... this equation has no meaning!
#it comes from SNOMLT = TMAX + RAIN * 0.4

crop management

The model runs several cases at the same time, and runs several crops in the rotation. So the management procedure is quite different from the original VBA code.
Warning, the column names were different in the maize management than in the other crops (e.g. Fdoy instead of Fpdoy, PDEN instead of Pden, no columns for stopDAP, SowTmp etc..). Column names should be :
FixFind, Fpdoy, Lpdoy, StopDAP, SowTmp, SowWat, Pden, STBLW (Fyear, yrno and SimDoy are not used, but they can be present).
But the order is not important.

first find which cases are sowing (resp harvesting,) and update sLastSowing (resp sLastHarvest), then do the necessary things (set/remove crop parameters, initialize state variables/reset state variables to their allvariables.xls default value).
Remains to be done: irrigation and nitrogen fertilization

Phenology

In the original excel version, phenology is determined by comparing the cumulated "biological days" from sowing with stage-change biological days computed at the beginning based on the biological days necessary to complete each stage (found in the crop parameter file, or, for maize silking, from cumulated thermal units between the two previous stages). It seemed more logical to cumulate biological days within each stage (resetting the cumulated sum to almost 0 (actually, the "extra" biological days beyond the threshold obtained on the day of the stage change are used to initialise the sum for the next day), and compare this number directly with the stage duration of each stage.
Another complicated aspect was the determination of when each process took place (e.g. vernalization, secondary growth of leaves etc...): these were hard-coded in the code for each species, based on phenological stages (e.g. for wheat bdTLM=bdBOT) or on parameters (e.g. for legume, bdTLM is a parameter found in the crop parameter file). The use of filters (see "filters", page 12) is supposed to simplify this: R-readable code in the crop parameter file tells under which condition each conditional process takes place. This also allows to change the names and/or number of growth stages from the parameter file without touching the code (except for maize, which has a number of specificities that make it necessary to use hard-coded names, cf below).

For example, the following phenological scales (as I understood them from the code) can be simply expressed as the following crop parameters (note that the filters are obtained with excel formulas from the original stage durations so it's easy to create these filters for new cultivars from their original parameters):

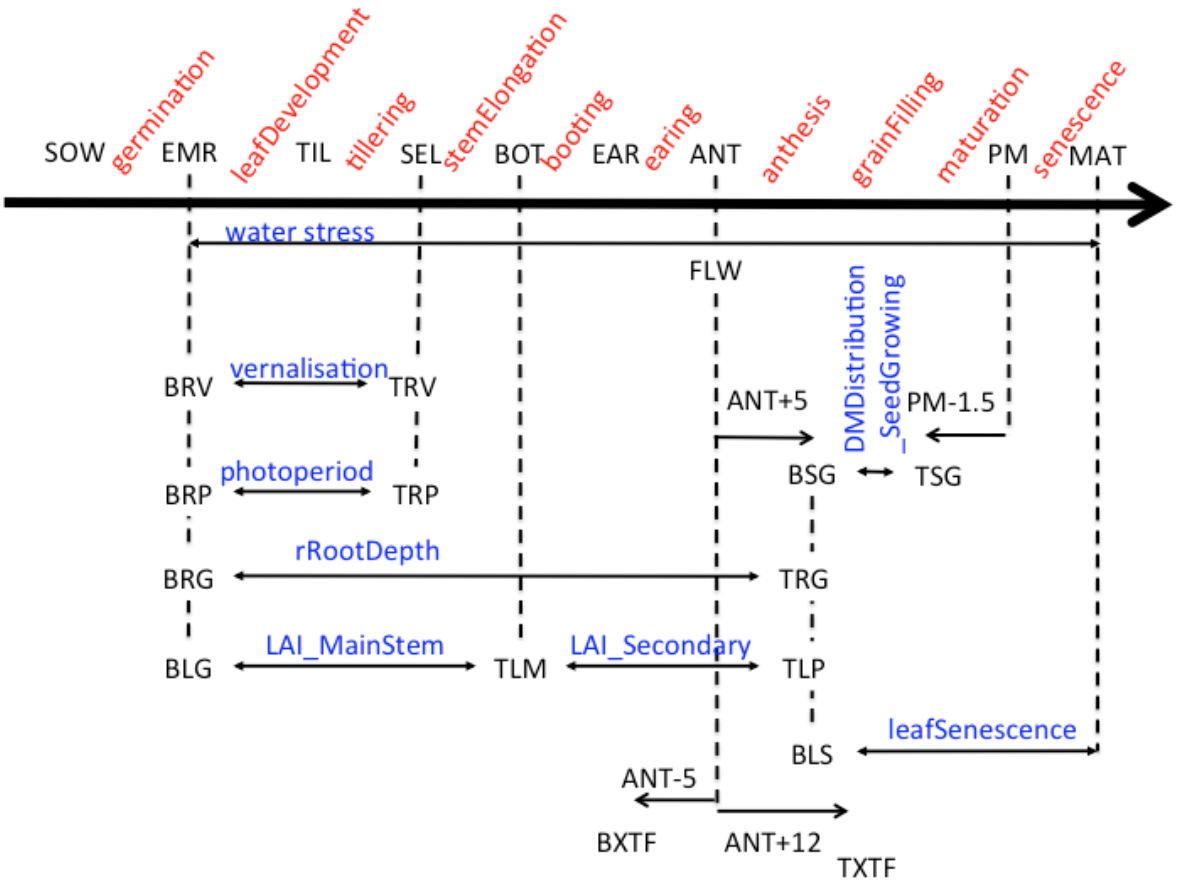


Figure 2: phenological scale of wheat. Names in black are the original version's names (except "bd" was removed), names in red are proposed stage names for SSM.R, names in blue are the names of the filters.

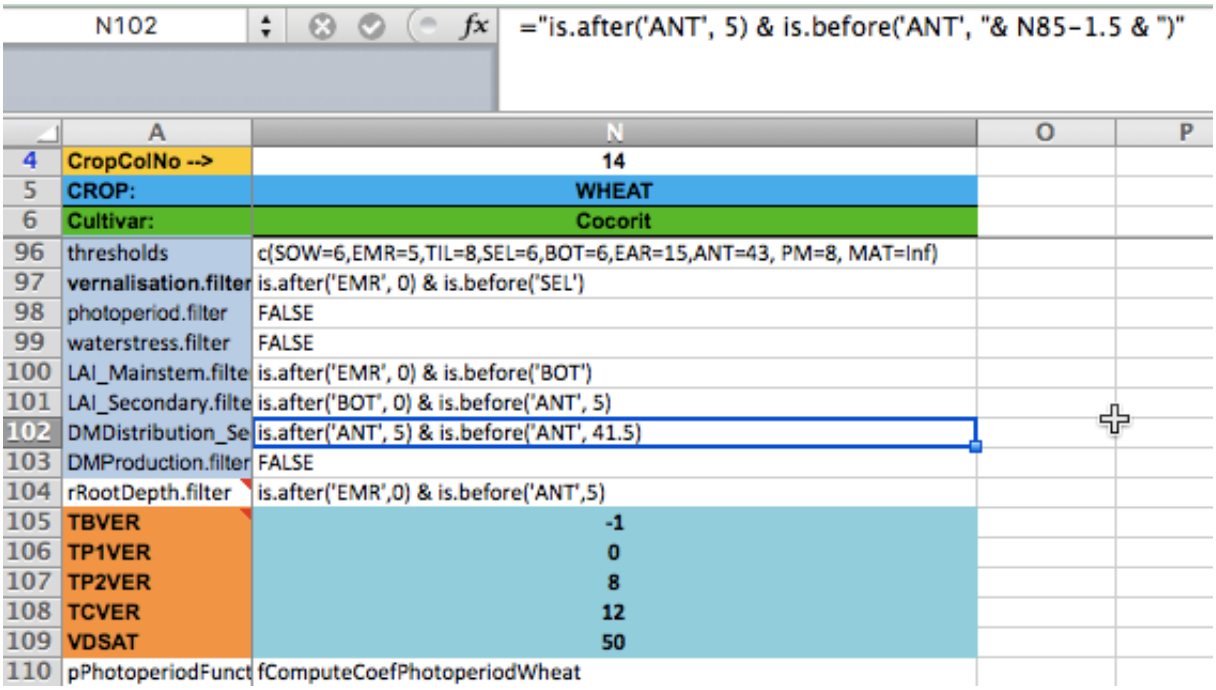
D96

```
="c(germination="&D79&",leafDevelopment="&D80&",tillering="&D81&",stemElongation="&D82&",booting="&D83&
",earing="&D84&",anthesis=5,  grainFilling="&D85-5-1.5&", maturation=1.5, senescence="&D86&", harvested=Inf)"
```

	A	D
4	CropColNo -->	4
5	CROP:	WHEAT
6	Cultivar:	Ble_Dur_1
96	thresholds	c(germination=6,leafDevelopment=5,tillering=8,stemElongation=6,booting=6,earing=15,anthesis=5, grainFilling=36.5, maturation=1.5, senescence=8, harvested=Inf)
97	vernalisation.filter	is.after('leafDevelopment', 0) & is.before('stemElongation')
98	photoperiod.filter	is.after('leafDevelopment', 0) & is.before('stemElongation')
99	waterstress.filter	FALSE
100	LAI_Mainstem.filte	is.after('leafDevelopment', 0) & is.before('booting')
101	LAI_Secondary.filte	is.after('booting', 0) & is.before('grainFilling')
102	DMDistribution_Se	is.after('grainFilling', 0) & is.before('maturation')
103	DMProduction.filter	FALSE
104	rRootDepth.filter	is.after('emergence',0) & is.before('grainFilling',0)
105	TBVER	-1
106	TP1VER	0
107	TP2VER	8
108	TCVER	12
109	VDSAT	50
110	pPhotoperiodFunct	fComputeCoefPhotoperiodWheat

Figure 3: possible specification of wheat phenology and conditioning of different crop processes to phenology

This scale uses stage names corresponding to Zadok's scale for cereals. If the user wants to keep the same phenological stage names as the original version:



N102 ="is.after('ANT', 5) & is.before('ANT', "& N85-1.5 & ")"

	A	N	O	P
4	CropColNo -->	14		
5	CROP:	WHEAT		
6	Cultivar:	Cocorit		
96	thresholds	c(SOW=6,EMR=5,TIL=8,SEL=6,BOT=6,EAR=15,ANT=43, PM=8, MAT=Inf)		
97	vernalisation.filter	is.after('EMR', 0) & is.before('SEL')		
98	photoperiod.filter	FALSE		
99	waterstress.filter	FALSE		
100	LAI_Mainstem.filte	is.after('EMR', 0) & is.before('BOT')		
101	LAI_Secondary.filte	is.after('BOT', 0) & is.before('ANT', 5)		
102	DMDistribution_Se	is.after('ANT', 5) & is.before('ANT', 41.5)		
103	DMProduction.filter	FALSE		
104	rRootDepth.filter	is.after('EMR',0) & is.before('ANT',5)		
105	TBVER	-1		
106	TP1VER	0		
107	TP2VER	8		
108	TCVER	12		
109	VDSAT	50		
110	pPhotoperiodFunct	fComputeCoefPhotoperiodWheat		

Figure 4: possible specification of wheat phenology and conditioning of different crop processes to phenology, following the original version's names

NB since these figures were drawn, I noticed that the filters were not necessary the same for the effect on DTU and on CBD (e.g. for maize between BRP and TRP), so now the filters for phenology are duplicated, one for the calculation of thermal units and one for the calculation of biological days.

For maize, the following scale can be translated in the following parameters:

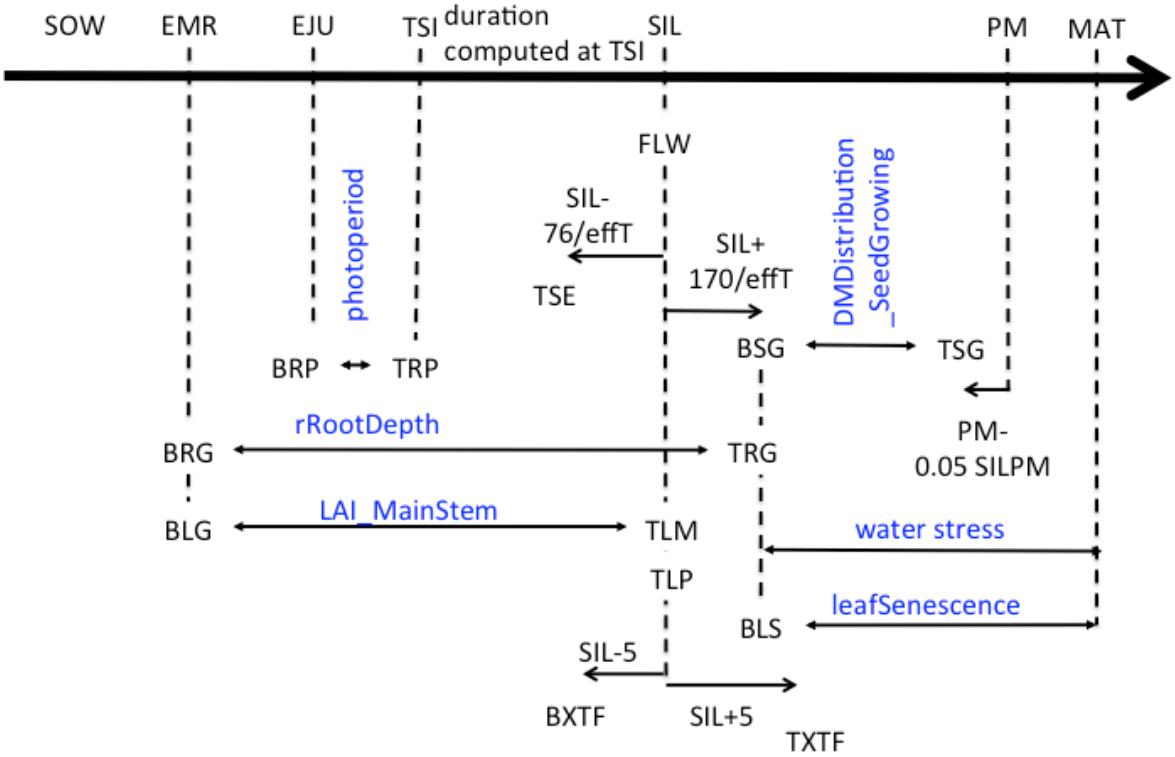
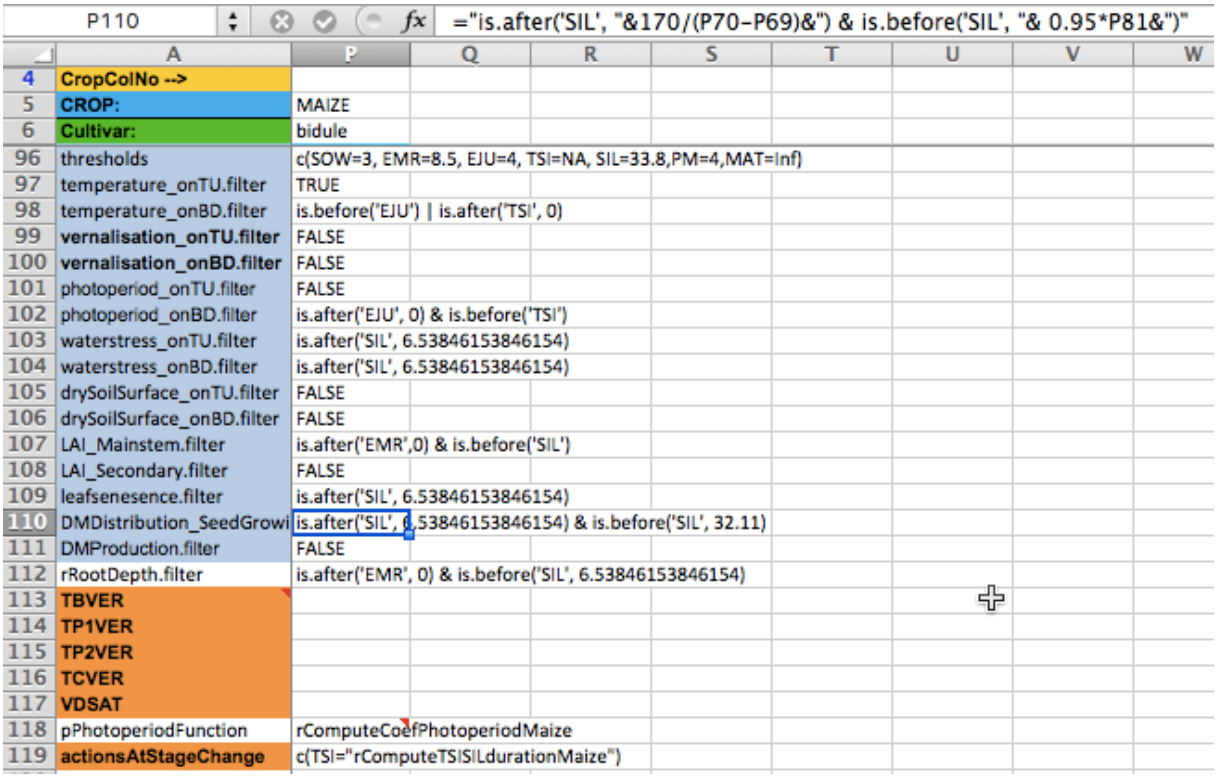


Figure 5: phenological scale of maize. Names in black are the original version's names (except "bd" was removed), names in blue are the names of the filters. effT is TP1D-TBD



P110 | ="is.after('SIL', "&170/(P70-P69)&") & is.before('SIL', "& 0.95*P81&")"

	A	P	Q	R	S	T	U	V	W
4	CropColNo -->								
5	CROP:	MAIZE							
6	Cultivar:	bidule							
96	thresholds	c(SOW=3, EMR=8.5, EJU=4, TSI=NA, SIL=33.8,PM=4,MAT=Inf)							
97	temperature_onTU.filter	TRUE							
98	temperature_onBD.filter	is.before('EJU') | is.after('TSI', 0)							
99	vernalisation_onTU.filter	FALSE							
100	vernalisation_onBD.filter	FALSE							
101	photoperiod_onTU.filter	FALSE							
102	photoperiod_onBD.filter	is.after('EJU', 0) & is.before('TSI')							
103	waterstress_onTU.filter	is.after('SIL', 6.53846153846154)							
104	waterstress_onBD.filter	is.after('SIL', 6.53846153846154)							
105	drySoilSurface_onTU.filter	FALSE							
106	drySoilSurface_onBD.filter	FALSE							
107	LAI_Mainstem.filter	is.after('EMR',0) & is.before('SIL')							
108	LAI_Secondary.filter	FALSE							
109	leafsenesence.filter	is.after('SIL', 6.53846153846154)							
110	DMDistribution_SeedGrowi	is.after('SIL', 6.53846153846154) & is.before('SIL', 32.11)							
111	DMProduction.filter	FALSE							
112	rRootDepth.filter	is.after('EMR', 0) & is.before('SIL', 6.53846153846154)							
113	TBVER								
114	TP1VER								
115	TP2VER								
116	TCVER								
117	VDSAT								
118	pPhotoperiodFunction	rComputeCoefPhotoperiodMaize							
119	actionsAtStageChange	c(TSI="rComputeTSISILdurationMaize")							

Figure 6: Sspecification of maize phenology and conditioning of different crop processes to phenology.

NB: the duration TSI-SIL is computed at runtime based on thermal units between emergence and TSI, therefore is it set as NA in the parameter file, and there is a procedure that is called at stage transition between EJU and TSI to compute the duration TSI-SIL (using the hard-coded name EMR for emergence). Note that this prevents using the stage TSE (which is not used in the model anyway) because it is defined as a duration before SIL : it is not possible to define its position in the excel file independently from the position of SIL. Therefore if TSE becomes necessary, it will be necessary to define it as a proper stage, transform the function rComputeTSISILdurationMaize into rComputeTSITSEdurationMaize (simply by substracting the duration of TSE-SIL to the computed duration) and add in the excel file the duration of TSE-SIL in the usual way.

NB photoperiod calculation for Maize uses the duration of stage EJU, on top of the regular photoperiod parameters (critical photoperiod and photoperiod sensitivity), so the name of this stage is hard-coded in the function that computes the photoperiod for maize. So it is not possible to change the name of this stage.

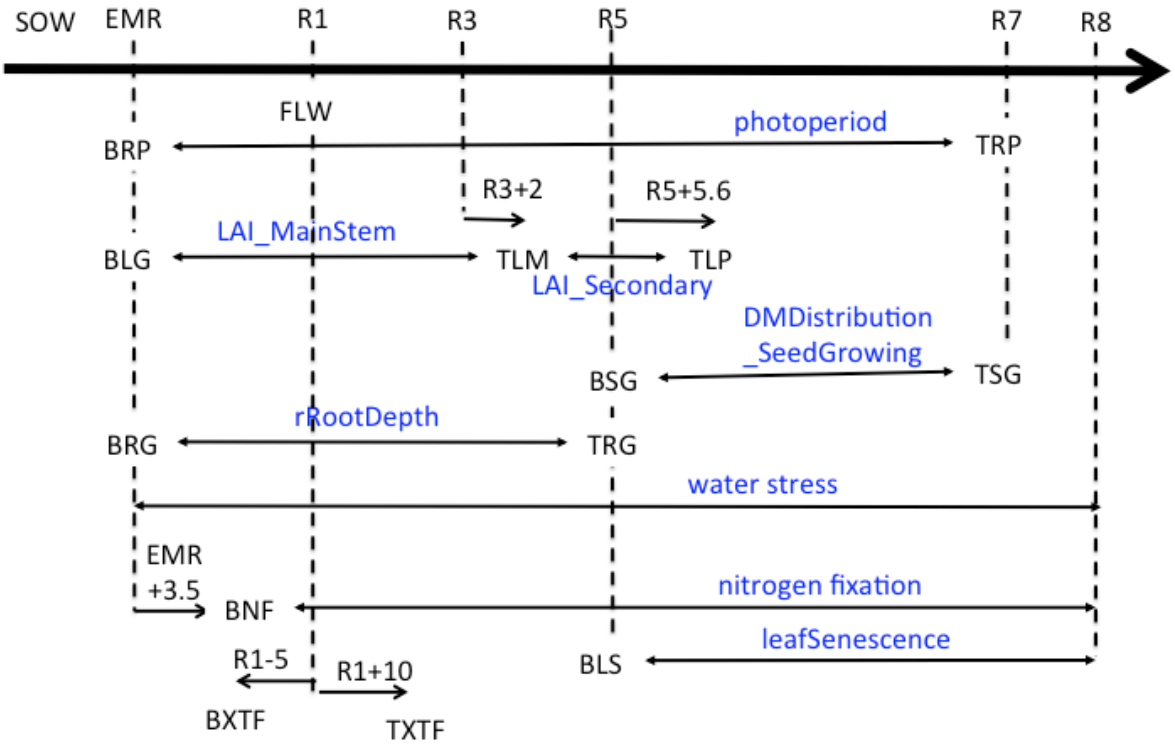


Figure 7: phenological scale of legumes. Names in black are the original version's names (except "bd" was removed), names in blue are the names of the filters

CropColNo -->	10
CROP:	Chickpea
Cultivar:	Ghab2
thresholds	c(SOW=8.5 ,EMR=36, R1=5.7, R3=4.3, R5=22, R7=7, R8=Inf)
temperature_onTU.filter	TRUE
temperature_onBD.filter	TRUE
vernalisation_onTU.filter	FALSE
vernalisation_onBD.filter	FALSE
photoperiod_onTU.filter	FALSE
photoperiod_onBD.filter	is.after('EMR', 0) & is.before('R7')
waterstress_onTU.filter	is.after('EMR', 0)
waterstress_onBD.filter	is.after('EMR', 0)
drySoilSurface_onTU.filter	FALSE
drySoilSurface_onBD.filter	is.before('EMR')
LAI_Mainstem.filter	is.after('EMR', 0) & is.before('R3', 2)
LAI_Secondary.filter	is.after('R3', 2) & is.before('R5', 5.6)
DMDistribution_SeedGrowi	is.after('R5', 0) & is.before('R7')
DMProduction.filter	FALSE
rRootDepth.filter	is.after('EMR',0) & is.before('R5')
TBVER	
TP1VER	
TP2VER	
TCVER	
VDSAT	
pPhotoperiodFunction	fComputeCoefPhotoperiodLegume

Figure 8: possible specification of legume phenology and conditioning of different crop processes to phenology.

NB: for legumes, the beginning and end of response to photoperiod, of seed growth, of LAI production and of leaf senescence were given as crop parameters in the original version, rather than relative to existing stages, most of the time it was easy to see from the numbers to which stage it corresponds, but for TLM, TLP and BNF, it doesn't match any stage change biological day, so I calculated the shift from the beginning of the stage.

LAI

In the version without N effect on senescence, the rate of LAI decrease is computed at runtime so that the LAI goes from its value at the beginning of leaf senescence to 0 at maturity stage, decreasing linearly when plotted on the biological day timescale. To do this, we used a filter as usual, to define through the parameter file the beginning of leaf senescence, but we had to add, within the LAI procedure, a way to compute the remaining biological days from the current time to the end of the crop cycle...which is the beginning of the last stage in the vector of stages (therefore, the vector of stages in the parameter file has to end with a "fake" stage, with infinite duration). This may not be optimal because (1) the end of leaf senescence is hard-coded to be the beginning of last stage, which may not be necessary true for future crops and (2) it might not seem logical to all users to have a "fake" stage with infinite duration at the end (for now, it is necessary because crop management is not coded yet so harvest never takes place... in the future, when harvest is coded, it might not be necessary anymore). We also added a state variable, sDecreaseLAIperBD, to keep the rate of decrease in memory from the time it is computed (at beginning of leaf senescence) to the end of the crop cycle ; in the original version, this variable was implicit from the equation DLAI = bd / (bdMAT - bdBLS) * BLSLAI, but we couldn't use it because in the R version the biological days are cumulated within each stage, so if there is more than one stage between the beginning of leaf senescence and maturity, the equation won't work.
LAI at the beginning of leaf senescence (BLSLAI) is also used in the computation of evapotranspiration. So we created a variable, sLAIforEvapotranspiration, to keep track of this variable. BUT there is a problem: in the code, BLSLAI is overwritten at different places: at beginning of leaf senescence (BLS), and at termination of secondary LAI growth (TLP).... and depending on the crop, BLS is before (legumes), after (maize) or the same (wheat) as TLP. I haven't had time to study in detail in which case BLSLAI is really at beginning of leaf senescence and in which case it isn't. For now, I use the value at beginning of leaf senescence.
LAI decrease with N effect hasn't been coded yet. Sot in the simulation options, Neffect has to be FALSE.

Dry matter production

No difference with the original version

DMDistribution

No difference with the original VBA code, which does not correspond to the code presented in the book: for example it does not take into account tuBSG and tuTSG
icicici cEffectOfHeatOnDHI (FrHtDHI) is never changed in the code, it is always 1
icicici I don't understand this equation : it seems to me DryMatterProduction is added twice:

cDailySeedWeightIncrease[phaseSeedGrowth]<-pmax(0, ((sDailyRateHIincrease*cEffectOfHeatOnDHI))*(sAccumulatedAboveGroundDryMatter + DryMatterProduction)))+ DryMatterProduction*sHarvestIndex)[phaseSeedGrowth])

root Depth

no difference with original version

Water budget

Note: I haven't been as careful in this module as in other modules to explicitly copy the needed variables at the beginning of the module, separating variables that will be updated and saved at the end from the variable that are just read... it doesn't change anything in the algorithm, but it makes it more difficult to check that no "illegal" overwriting is done, and to use default values instead of computed values for debugging purposes.

Only sWater.n (with n from 1 to 10) is saved, as well as the water stress coefficients (because they are computed in a module at the beginning of the timestep and are used after by other modules), all other water-related variables (ATSW(L), FTSW, FTWSRZ, WSOL...) are recomputed when needed. The function (actually it's an hybrid function/procedure because it returns a value, but it accesses global variables) is called fFindWater. Refer to the Roxygen comments to see how it works. The function fFindRLYER works the same way.

The only problem with this is for cEfficientRootLength (AROOT), which is computed twice, once in rUpdateStresses, with root length from the day before, and once in rWaterBudget, with the current day's root length (because rWaterBudget comes after rRootDepth). I haven't figured out yet what variables (the current day or the day before) are used in the code because it's quite difficult to follow (the variables are updated within layer loops, and dependent variables are often computed in the same loop, sometimes before, sometimes after the update). So this is still an open issue.

icicici dont forget to code irrigation in management module

icicicici BLSLAI is overwritten at different places in the original code: at beginning of leaf senescence (BLS), and at termination of secondary LAI growth (TLP).... and depending on the crop, BLS is before (legumes), after (maize) or the same (wheat) as TLP. I haven't had time to study in detail in which case BLSLAI is really at beginning of leaf senescence and in which case it isn't

#warning: in SSM.R, VPDF (and latitude) has been moved from location-specific parameters to soil parameters, to avoid having a file just for locations

check units: #icicicic VPD in kPa, TEC in Pa

Soil nitrogen

not coded yet

Plant nitrogen

not coded yet