Minimalist Mixture Model - M^3

Technical documentation of the model and its source code

Berghuijs, H.N.C.[1,2]; Vico G.[1]

[1]Department of Crop Production Ecology, Swedish University of Agricultural Sciences (SLU), Ulls väg 16, 750 07, Uppsala, Sweden

[2]Plant Production Systems Group, Wageningen University and Research (WUR), Droevendaalsesteeg 1, 6708 PB, Wageningen, The Netherlands

Contacts: Herman Berghuijs (herman.berghuijs@wur.nl); Giulia Vico (giulia.vico@slu.se)

For the model rationale, calibration and validation see

Berghuijs HNC, Wang Z, Stomph TJ, Weih M, Van der Werf W, Vico G (2020), Identification of crop traits enhancing wheat-faba bean intercrop productivity: a crop growth model and sensitivity analyses, *Plant and Soil*, 455, 203–226, https://doi.org/10.1007/s11104-020-04668-0

Contents

1. Introduction

The M^3 model (Minimalist Mixture Model) is a crop growth model simulating the growth of strip intercrops and of pure cultures of two species. The model was developed with the intent of limiting as much as possible the model parameters and hence the requirements of data for model calibration. The model and its rationale are described in detail by Berghuijs *et al.* (2020) and summarized in this document. This document provides the technical documentation on the source code of the crop growth model M^3 (Minimalist Mixture Model).

This crop growth model is capable of simulating pure cultures as well as intercrops consisting of two crop species, and the effects of nitrogen-limited conditions. The model is based on six state variables for each crop species (aboveground dry matter, crop height, crop nitrogen amount, leaf area index, storage organ weight, temperature sum from sowing) and an additional non-crop specific state variable (soil mineral nitrogen amount).

The model is coded in C# programming language (Hejlsberg *et al.*, 2008) within the 4.6.1 .NET framework. The source code consists of several projects, which in turn consist of various classes. The classes contain methods (called class methods) and variables of various types (called class variables).

A batch of simulations starts by reading a simulation input file that contains the locations of the input files of each simulation in the batch. For each simulation files, M^3 reads a number of input files (relative to crop, farmer, simulation, soil, and weather station input files). The stored model output of the simulation is written to an output file, listing the values of each state variable for each day of the simulation and a number of auxiliary variables that are calculated from these states..

The aim of this document is to provide a documentation of the source code and contains i) Instructions on how to use the model (Section 2), ii) the overall description of the model and of the model equations (Section 3), iii) the inventory of the different projects and their classes, making up the source code and the input and output files of the model (Section 4); iv) a description of how the different parts of the model interact during a run (Section 5); v) a technical description of each model component (Section 6); vi) a description of the model input files (Section 7); and vii) a description of model output files (Section 8).

2. User instructions

M^3 can be run as follows:

- If M^3 is used for the first time, unpack M3_source_code.zip at a chosen location.

- Navigate to ...\M3_source_code\M3\RunSimulationsFromInputFile\bin\Debug

- Run the file "RunSimulationsFromInputFile.exe"

This will run all the simulations that are specified in the file "...\M3_source_code\M3\RunSimulationsFromInputFile\bin\Debug\simulationInput.csv". The output files from the simulations will be saved in the folder ""...\M3_source_code\M3\RunSimulationsFromInputFile\bin\Debug\Output files".

M^3 source.zip contains, besides the model, also a number of input files for sample simulations. More information on the structure of these input files can be found in Section 8.

3. Model description

3.1 Overall model description

M^3 simulates the change of the state of an intercrop during the course of a simulation, starting at sowing date and ending when the last crop in the intercrop is harvested. The state of the intercrop is described by the state variables, referring to species 1 and species 2 in the intercrop:

a1) Aboveground dry matter (kg DM m^{-2} ground) species 1

a2) Aboveground dry matter (kg DM m^{-2} ground) species 2

b1) Crop nitrogen amount (kg N m^{-2} ground) species 1

b2) Crop nitrogen amount (kg N m^{-2} ground) species 2

c1) Crop height (m) species 1

c2) Crop height (m) species 2

d1) Leaf area index (m^2 leaf m^{-2} ground) species 1

d2) Leaf area index (m^2 leaf m^{-2} ground) species 2

e1) Storage organ dry matter (kg DM m^{-2} ground) species 1

e2) Storage organ dry matter (kg DM m^{-2} ground) species 2

f1) Temperature sum from the sowing date (°C d) species 1

f2) Temperature sum from the sowing date (°C d) species 2

Additionally, we assume that two species of the intercrop share the same pool of soil mineral nitrogen. Therefore, there is a single additional state variable representing

g) Soil mineral nitrogen amount (kg N m^{-2})

M^3 starts a simulation with the initialization of the state variables. All the state variables, except the soil mineral nitrogen amount, are set to 0. The soil mineral nitrogen amount is set to a constant value, defined in the soil input file.

From the sowing date until the harvest date of the last crop, M^3 simulates the daily gain of soil mineral nitrogen due to fertilization events and net mineralization and the loss by uptake of soil mineral nitrogen by the crops. Each crop species emerges once the temperature sum from sowing is larger than a threshold value. From the date that the first crop emerges until the date that the last crop species is harvested, M^3 calculates the daily fraction of the global radiation that is intercepted by each crop species, in turn a function of the light extinction coefficients, crop heights and strip widths, and leaf area indices of both species. Next, it calculates the daily intercepted photosynthetically active radiation for each species. Then, it calculates for each species the potential daily biomass production, considering the species radiation use efficiency. The actual daily biomass production of a species equals its potential daily biomass production, if it has access to enough soil mineral nitrogen and nitrogen obtained by N_2 fixation to maintain its nitrogen content above a critical value – a function also of its aboveground dry biomass. If the crop cannot maintain this nitrogen concentration, the actual biomass production is reduced with respect to the potential biomass production. The actual biomass production is completely halted if a species' crop nitrogen content is equal or smaller than the minimum nitrogen concentration of that species. After M^3 has determined the daily biomass production of each species, it calculates how this new biomass is partitioned over the storage organs, leaves and remaining vegetative organs. This

biomass partitioning depends on the developmental stage of the crop, in turn determined by the state variable temperature sum from emergence and the parameters temperature sum from emergence to anthesis and temperature sum from anthesis to maturity. It also simulates how much leaf dry matter (and therefore leaf area index) and crop nitrogen are lost due to leaf senescence. Leaf senescence only takes place after anthesis, at a rate dependent on temperature.

3.2 Model equations

3.2.1 State variables

The dynamics of each state variable is described by finite difference equations of the form:

$$x(t + \Delta t) = x(t) + \Delta t \cdot G_x(t) \quad (3.1)$$

where $x(t)$ is a generic state variable, Δt is the time step (d) and $G_x(t)$ is the net growth rate of this variable. The time step is always set to 1 day in M^3. The M^3 model consists of 13 state variables (2 sets of crop species specific state variables + 1 non-crop species specific state variable). M^3 simulates at each time t what the values of these variables are. The crop species specific state variables of each species *i* are:

a) Aboveground dry matter $B_i(t)$ (kg DM m^{-2} ground)

b) Crop nitrogen amount $N_i(t)$ (kg N m^{-2} ground)

c) Crop height H_i (m)

d) $L_i(t)$ (m^2 leaf m^{-2} ground)

e) Storage organ dry matter $Y_i(t)$ (kg DM m^{-2} ground)

f) Temperature sum from the sowing date $S_i(t)$ (°C d)

Further, there is one non-crop species specific state variable:

g) Amount of nitrogen in the soil ($N_s(t)$: in kg N m^{-2} ground)

3.2.2 Rates of change of the state variables

The net growth rate G_x of a state variable x can be described as the rate $P_x(t)$ of the various processes that increase x and the rate $M_x(t)$ of the various processes that decrease x:

$$G_x(t) = P_x(t) - M_x(t) \quad (3.2)$$

Substitution of equation 3.2 in equation 3.1 gives:

$$x(t + \Delta t) = x(t) + \Delta t \cdot \big(P_x(t) - M_x(t)\big) \quad (3.3)$$

Hence, the finite difference equations for each of the crop specific state variables in M^3 are:

$$S_i(t+\Delta t) = S(t) + \Delta t \cdot P_{S,i}(t) \quad (3.4)$$
$$L_i(t+\Delta t) = L(t) + \Delta t \cdot \left(P_{L,i}(t) - M_{L,i}(t)\right) \quad (3.5)$$
$$B_i(t+\Delta t) = B(t) + \Delta t \cdot \left(P_{B,i}(t) - M_{B,i}(t)\right) \quad (3.6)$$
$$Y_i(t+\Delta t) = Y(t) + \Delta t \cdot P_Y(t) \quad (3.7)$$
$$H_i(t+\Delta t) = H_i(t) + \Delta t \cdot P_{H,i}(t) \quad (3.8)$$
$$N_{\mathrm{c},i}(t+\Delta t) = N_{\mathrm{c},i}(t) + \Delta t \cdot \left(P_{N_c,i}(t) - M_{N_c,i}(t)\right) \quad (3.9)$$

The remaining state variable for the amount of nitrogen in the soil is crop-specific (i.e. the crops share the same nitrogen pool). The finite difference equation for the amount of nitrogen in the soil is

$$N_s(t+\Delta t) = N_s(t) + \Delta t \cdot \left(P_{N_s}(t) - M_{N_s}(t)\right) \quad (3.10)$$

In the remainder of section 3.2.2, we will specify the rates of change further.

3.2.2.1 Rate of change for the temperature sum $S_i(t)$

The net increase of the temperature sum equals the gross rate of those processes that increase the temperature sum, as there are no processes that decrease this temperature sum. This gross growth rate of the temperature sum is calculated as:

$$P_{S,i}(t) = T_{\mathrm{eff},i}(t) \quad (3.11)$$

where $T_{\mathrm{eff},i}(t)$ is the effective temperature (°C) at time t for species i. It is calculated as:

$$T_{\mathrm{eff,i}}(t) = \max\left(0, \frac{T_{\min}(t) + T_{\max}(t)}{2} - T_{\mathrm{b},i}\right) \quad (3.12)$$

where $T_{\min}(t)$ and $T_{\max}(t)$ are, respectively, the minimum and maximum temperatures (°C) between time t and time $t+\Delta t$ (i.e., over each day, considering that the time step is 1 d). $T_{\mathrm{b,i}}$ is the base temperature of species i, i.e., the effective temperature below which the temperature sum of species i does not increase. For each time step, the developmental stage δ_i is calculated from the temperature sum as:

$$\delta_i(t) = \begin{cases} 0 & | \quad S_{\mathrm{i}}(t) \le S_{\mathrm{e},i} \\ \dfrac{S_i(t) - S_{\mathrm{e,i}}}{S_{\mathrm{a,i}}} & | \quad S_{\mathrm{e},i} < S_i(t) \le S_{\mathrm{e},i} + S_{\mathrm{a},i} \\ 1 + \dfrac{S_i(t) - S_{\mathrm{a,i}} - S_{\mathrm{e},i}}{S_{\mathrm{m,i}}} & | \quad S_{\mathrm{e},i} + S_{\mathrm{a},i} < S_i(t) \le S_{\mathrm{e},i} + S_{\mathrm{a},i} + S_{\mathrm{m},i} \\ 2 & | \quad S_i(t) \ge S_{\mathrm{e},i} + S_{\mathrm{m},i} + S_{\mathrm{a},i} \end{cases} \quad (3.13)$$

where $S_{\mathrm{e,i}}$ is the temperature sum between sowing and emerging, $S_{\mathrm{a,i}}$ is the temperature sum from emergence to anthesis, and $S_{\mathrm{m,i}}$ is the temperature sum from anthesis to maturity.

3.2.2.2 Rates of change for the leaf area index $L_i(t)$

The net increase in leaf area index is calculated as the difference between the rates of processes that increase the leaf area index $P_{L,i}(t)$ and processes that decrease the leaf area index $M_{L,i}(t)$.

The leaf area index is increased by the interception of photosynthetic active radiation, the subsequent conversion of intercepted radiation into dry matter, and the allocation of this newly produced biomass to the leaves:

$$P_{L,i}(t) = \sigma_{\mathrm{N},i}(t) \cdot S_{\mathrm{la,i}} \cdot f_{\mathrm{L,i}}(t) \cdot P_{\mathrm{pot},i}(t) \tag{3.14}$$

where $P_{\mathrm{pot,i}}(t)$ is the daily potential production rate of dry matter, $f_{\mathrm{L},i}(t)$ is the fraction of newly produced biomass that is allocated to leaves (kg dry matter leaf kg^{-1} dry matter), $S_{\mathrm{la},i}$ is the specific leaf area (m^2 leaf kg^{-1} leaf dry matter), and $\sigma_{\mathrm{N,i}}(t)$ is a reduction factor between 0 that 1 that accounts for nitrogen shortage. The variables $P_{\mathrm{pot},i}(t)$ and $f_{\mathrm{L},i}(t)$ will be further specified in this section. The variable $\sigma_{\mathrm{N,i}}(t)$ will be further specified in section 3.28. The potential production rate is calculated as:

$$P_{\mathrm{pot},i}(t) = \eta_i \cdot f_p \cdot f_{\mathrm{int},i}(t) \cdot I_0(t) \tag{3.15}$$

where η_i is the light use efficiency (kg dry matter J^{-1} PAR), f_{p} is the fraction of photosynthetically active radiation (PAR) in the global radiation, and $I_0(t)$ is the daily amount of global radiation (J radiation m^{-2} d^{-1}), $f_{\mathrm{int,i}}(t)$ is the fraction of PAR that is intercepted by the canopy of the crop i: its calculation is detailed in the next subsection.

The fraction of newly produced biomass that is allocated to the leaves is calculated as:

$$f_{\mathrm{L,i}}(t) = \begin{cases} f_{\mathrm{L,0,i}} & | \quad \delta_{\mathrm{i}}(t) \le d_{1,\mathrm{i}} \\ f_{\mathrm{L,0,i}} \cdot \dfrac{d_{2,\mathrm{i}} - \delta_{\mathrm{i}}(t)}{d_{2,\mathrm{i}} - d_{1,\mathrm{i}}} & | \quad d_{1,\mathrm{i}} < \delta_{\mathrm{i}}(t) \le d_{2,\mathrm{i}} \\ 0 & | \quad \delta_{\mathrm{i}}(t) > d_{2,\mathrm{i}} \end{cases} \tag{3.22}$$

where $f_{\mathrm{L,0}}$ is the fraction of newly produced biomass partitioning to the leaves if the developmental stage of species i is smaller than $d_{1,i}$. $d_{1,i}$ is the developmental stage above which the fraction of newly produced biomass that is partitioned to the leaves starts to decrease. $d_{2,i}$ is the developmental stage above which no more newly produced biomass is allocated to the leaves.

The rate of the processes that decrease the leaf area index $M_{L,i}(t)$ is calculated as:

$$M_{L,i}(t) = \mu_{L,i}(t) \cdot L_i(t) \tag{3.23}$$

where $\mu_{L,i}$ is the relative mortality of leaf dry matter (d^{-1}):

$$\mu_{\mathrm{i}}(t) = \begin{cases} 0 & | \quad \delta_{\mathrm{i}}(t) \le 1 \vee \dfrac{T_{\min}(t) + T_{\max}(t)}{2} \le T_{\mathrm{sen,i}} \\ \min\left(1, s_{\mathrm{sen,i}} \cdot \left(\dfrac{T_{\min}(t) + T_{\max}(t)}{2} - T_{\mathrm{sen,i}}\right)\right) & | \quad \delta_{\mathrm{i}}(t) > 1 \wedge \dfrac{T_{\min}(t) + T_{\max}(t)}{2} > T_{\mathrm{sen,i}} \end{cases} \tag{3.24}$$

where $T_{\mathrm{sen,i}}$ is the temperature above which leaf senescence takes place (°C). $s_{\mathrm{sen,i}}$ is the relative, temperature-dependent rate of leaf senescence ((°C d)$^{-1}$).

3.2.2.3 Light intercepted fractions $f_{int,i}$ and their components

The light interception fractions are calculated almost completely as in the model developed by Gou *et al.* (2017) for monocultures or strip intercrops. The only difference between their radiation interception model and the radiation interception model in M^3 is that the radiation interception model in M^3 does not consider light interception by storage organs (Berghuijs *et al.*, 2020).

The light interception model in M^3 can be used just for up to two species. This is the only part of the model that is not immediately extensible to intercrops including more than two species. At each instant the model determines which crop is the shortest one (subscript *s* hereafter) and which is the tallest one (subscript *t*), based on their heights, $H_{\mathrm{t}}(t)$ and $H_{\mathrm{s}}(t)$ respectively. The tall canopy has strip width $\ell_{\mathrm{t}}(t)$ and light extinction coefficient $k_{\mathrm{t}}(t)$; the shortest canopy has strip width $\ell_{\mathrm{s}}(t)$ and light extinction coefficient $k_{\mathrm{s}}(t)$.

The light intercepted by the tallest canopy is given by the contribution of the interception by its upper part, i.e., the part of its canopy emerging above the shortest canopy (of depth $H_{t,u}(t)$), $f_{\mathrm{t,u}}(t)$; and that of the lower canopy, from the ground to the height of the shortest crop (i.e., of depth $H_{t,l}(t)$), $f_{\mathrm{t,l}}(t)$, i.e., $f_{\mathrm{int,t}}(t) = f_{\mathrm{t,l}}(t) + f_{\mathrm{t,u}}(t)$. The fraction $f_{\mathrm{t,u}}(t)$ is calculated as the weighted average of the light interception by homogeneous canopy and a compressed one (i.e., a canopy where all the rows were pushed together; subscript *c*)

$$f_{\mathrm{t,u}}(t) = \left(1 - w(t)\right)\left(1 - \mathrm{e}^{-k_{\mathrm{t}}(t)\cdot L_{\mathrm{t,u}}(t)}\right) + w(t)\left(1 - \mathrm{e}^{-k_{\mathrm{t}}(t)\cdot L_{\mathrm{t,u,c}}(\mathrm{t})}\right)\cdot\frac{\ell_{\mathrm{t}}(t)}{\ell_{\mathrm{s}}(t) + \ell_{\mathrm{t}}(t)} \tag{3.16}$$

where $L_{\mathrm{t,u}}(t) = \frac{H_{t,u}(t)}{H_t(t)}\, L_{\mathrm{t}}(t)$ is the leaf area index of the upper part of the tall canopy – a fraction of the total leaf area of the tall canopy $L_{\mathrm{t}}(t)$ – and the weighing coefficient $w(t)$ represents how similar the intercropped canopy is to a compressed canopy and is zero for a homogeneous canopy.

$f_{\mathrm{t,l}}(t)$ is calculated as

$$f_{\mathrm{t,l}}(t) = f_{\mathrm{trans},t}(t)\cdot\left(1 - e^{-k_t(t)\cdot L_{t,l,c}(t)}\right)\cdot\frac{\ell_t(t)}{\ell_s(t) + \ell_t(t)} \tag{3.17}$$

where $L_{t,l,c}(t) = \frac{H_s(t)}{H_t(t)}\frac{\ell_s(t)+\ell_t(t)}{\ell_t(t)} L_t(t)$ is the leaf area index of the compressed lower part of the tall canopy and $f_{\mathrm{trans},t}(t)$ is the fraction of incoming radiation transmitted to the top of the lower part of the tall canopy. The latter is determined considering the contribution of light transmitted completely through the upper part of the taller canopy and the light travelling sideways and being only partially transmitted through the upper part of the tall canopy, i.e.,

$$f_{\mathrm{trans},t}(t) = V_t(t)\cdot e^{-k_t(t)\cdot L_{t,u,c}(t)} + (1 - V_t(t))e^{-k_t(t)\cdot L_{t,u}(t)} \tag{3.18}$$

with $V_t(t)$ being the sky view factor (or relative radiation onto the path) of the top of the lower part of the tall canopy.

The shortest crop canopy intercepts only the light that reaches its top:

$$f_s(t) = f_{trans,s}(t) \cdot (1 - e^{-k_s(t) \cdot L_{s,c}(t)}) \cdot \frac{\ell_s(t)}{\ell_s(t) + \ell_t(t)} \quad (3.19)$$

where the leaf area index of the compressed short canopy is given by $L_{\mathrm{s,c}}(t) = \frac{\ell_s(t)+\ell_t(t)}{\ell_s(t)} L_s(t)$.

The fraction of light reaching the top of the shortest canopy, $f_{trans,s}(t)$, is given by

$$f_{\mathrm{trans,s}}(t) = V_{\mathrm{s}}(t) + \left(1 - V_{\mathrm{s}}(t)\right) \cdot \mathrm{e}^{-k_{\mathrm{t}}(t) \cdot L_{\mathrm{t,u}}(t)} \quad (3.20)$$

where $V_s(t)$ is the sky view factor of top of the short canopy. The second term in the above equation considers the amount of light reaching the top of the short canopy by travelling through the upper part of the tall canopy.

In the equations above, the view factors are calculated as

$$V_i(t) = \frac{\sqrt{\ell_i(t)^2 + H_{t,u}(t)^2} - H_{t,u}(t)}{\ell_i(t)} \quad (3.20)$$

with *i=s* or *t* depending on whether the view factor refers to the tall or short canopy. The weighing factor $w(t)$ is determined as

$$w(t) = \frac{f_{\mathrm{trans,s}}(t) - f_{\mathrm{trans,t}}(t)}{1 - \mathrm{e}^{-k_{\mathrm{t}}(t) \cdot L_{\mathrm{t,u,c}}(t)}} \quad (3.21)$$

following the rationale described by Goudriaan (1977), Pronk *et al.* (2003), and Gou *et al.* (2017). A more detailed description of the light interception model is provided in the supplementary material in Berghuijs *et al.* (2020).

While the above light interception model was developed for strip intercrops, it can be effectively applied also to homogenous intercrops, in the limit of strip widths of both species, ℓ_t and ℓ_s, going to zero, while maintaining a finite ratio $\frac{\ell_s(t)}{\ell_t(t)} = s_r(t)$. In this limit, $\frac{\ell_s(t)}{\ell_s(t)+\ell_t(t)} = \frac{s_r(t)}{1+s_r(t)}$, $\frac{\ell_t(t)}{\ell_s(t)+\ell_t(t)} = \frac{1}{1+s_r(t)}$, and $V_j(t) \to 0$.

3.2.2.4. Rates of change for the total aboveground biomass $B_i(t)$

The net increase in the total aboveground biomass is calculated as the difference between the rates of processes that increase the total aboveground dry biomass and processes that decrease this. The gross rate of the increase of the aboveground biomass is calculated as:

$$P_{B,i}(t) = \sigma_{\mathrm{N,i}}(t) \cdot P_{\mathrm{pot},i}(t) \quad (3.22)$$

$\sigma_{\mathrm{N,i}}(t)$ quantifies the reduction of crop growth of species *i* at time *t* due to nitrogen limitation, and will be specified in section 3.2.2.9. M^3 assumes that only leaf tissue dies off during the growing season.

M^3 calculates the gross decrease in the aboveground biomass as:

$$M_{B,i}(t) = \mu_{\mathrm{i}}(t) \cdot \frac{L_{\mathrm{i}}(t)}{S_{\mathrm{la,i}}} \tag{3.23}$$

3.2.2.5 Rate of change for storage organ weight $Y_i(t)$

The net increase of the storage organ weight equals its gross rate of increase, as there are no processes that decrease the storage organ weight. It is calculated as:

$$P_{Y,i}(t) = \sigma_{\mathrm{N,i}}(t) \cdot f_{\mathrm{y,i}}(t) \cdot P_{\mathrm{pot}}(t) \tag{3.24}$$

$f_{\mathrm{y,i}}(t)$ is the fraction of newly produced biomass that is assigned to the storage organs. M^3 calculates it as:

$$f_{\mathrm{y,i}}(t) = \begin{cases} 0 & | \quad \delta_{\mathrm{i}}(t) < d_{3,\mathrm{i}} \\ \left(1 - f_{\mathrm{L,i}}(t)\right) \cdot \dfrac{\delta_{\mathrm{i}}(t) - d_{3,\mathrm{i}}}{d_{4,\mathrm{i}} - d_{3,\mathrm{i}}} & | \quad d_{3,i} < \delta_{\mathrm{i}}(t) < d_{4,\mathrm{i}} \\ 1 - f_{\mathrm{L,i}}(t) & | \quad \delta_{\mathrm{i}}(t) > d_{4,\mathrm{i}} \end{cases} \tag{3.25}$$

where $d_{3,i}$ is the developmental stage above which the crop starts to allocate newly produced biomass to the storage organs. $d_{4,i}$ is the developmental stage above which the crop no longer assigns newly produced biomass to other organs than leaves and storage organs.

3.2.2.6 Rate of change for crop height $H_i(t)$

The net increase of the crop height equals its gross rate of increase, as there are no processes in M^3 that decrease crop height. This rate is calculated as:

$$P_{H,i}(t) = r_{\mathrm{h,i}} \cdot T_{\mathrm{eff}}(t) \cdot H_{\mathrm{i}}(t) \cdot \left(1 - \frac{H_{\mathrm{i}}(t)}{H_{\mathrm{m,i}}}\right) \tag{3.26}$$

where $r_{\mathrm{h,i}}$ is the relative height growth rate ((°C d)$^{-1}$). $H_{\mathrm{m},i}$ is the maximum crop height (m).

3.2.2.7. Rates of change for mineral soil nitrogen $N_s(t)$

The net increase of the amount of mineral soil nitrogen is the difference between processes that add soil mineral nitrogen to the soil and remove it. Nitrogen can be added to the soil by fertilization and by mineralization of soil organic matter. The gross rate of increase of soil mineral nitrogen is calculated as:

$$P_{N_s}(t) = f_{\mathrm{recov}} \cdot F(t) + m \tag{3.27}$$

where $F(t)$ is the daily amount of fertilizer that is added at time t and f_{recov} is the recovery fraction of added fertilizer. m is the net mineralization rate (kg N m^{-2} d^{-1}). m can be negative, which would indicate that more soil mineral nitrogen is removed by immobilization than released by mineralization. It is assumed to be constant over time. The gross rate of decrease of soil mineral nitrogen is calculated as:

$$M_{N_s}(t) = \sum_{j=1}^{n_c} U_j(t) \tag{3.28}$$

where $\sum_j^{n_c} U_j(t)$ is the sum of nitrogen uptake by all species in the strip intercrop. n_c is the number of crop species. In the current version of M^3, n_c equals 1 for monoculture and or 2 for two-crop intercrops. In section 3.2.2.8, we will further specify how the nitrogen uptake of by species is calculated (equation 3.38).

3.2.2.8 Rates of change for crop nitrogen $N_{c,i}(t)$

The net increase in the amount of crop nitrogen $N_{c,i}(t)$ is calculated as:

$$P_{N_{c,i}}(t) = U_i(t) + \Phi_i(t) \tag{3.29}$$

where $U_i(t)$ is the rate of nitrogen uptake by species i from the soil (kg N m^{-2} ground d^{-1}) and $\Phi_i(t)$ is the rate of nitrogen fixation by species i (kg N m^{-2} ground d^{-1}). Both of these parameters will be further specified in this section in equation 3.36 ($\Phi_i(t)$) and 3.38 ($U_i(t)$). The nitrogen uptake and fixation of a species i is calculated from, amongst others, the demand for nitrogen of this species, $D_i(t)$. The nitrogen demand of the crop is calculated as:

$$D_i(t) = \Delta t \cdot \max\left(0, B_i(t) \cdot \left(\nu_{\max,i}(t) - \nu_i(t)\right)\right) \tag{3.30}$$

i.e., the nitrogen demand equals the amount of nitrogen that a crop needs to take up in order to maintain its nitrogen concentration $\nu_i(t)$ equal to the maximum nitrogen concentration of that species $\nu_{\max,i}(t)$. In order to calculate this demand, M^3 calculates the actual nitrogen concentration of the crop as

$$\nu_i(t) = \frac{N_{c,i}(t)}{B_i(t)} \tag{3.31}$$

The crop maximum nitrogen concentration is determined based on an empirical relationship with aboveground dry matter (nitrogen-dilution curve), as:

$$\nu_{\max,i}(t) = \nu_{\max,0,i} \cdot \min\left(1, \left(\frac{\ell_i + \ell_j}{\ell_i} \cdot \frac{B_i(t)}{B_{\max,i}}\right)^{-\beta_{\max,i}}\right) \tag{3.32}$$

where $\nu_{\max,0,i}$ is the maximum nitrogen content for low aboveground biomasses of species i. ℓ_i is the strip width of species i and ℓ_j is the strip width of the second species of the intercrop species j. The parameter $B_{\max,i}$ is the aboveground biomass for which the maximum nitrogen content would equal 1, in case the model would not assume a constant value of nitrogen contents at low biomass (i.e. $\nu_{\max,0,i}$). $\beta_{\max,i}$ is a shape parameter.

Legumes are capable of fixing N_2 from the atmosphere and can thus fulfil a fraction $f_{fix,i}$ of their nitrogen demand by means of N_2 fixation; this fraction is set to 0 for crops not capable to fix atmospheric nitrogen. The rate of N fixation is calculated as:

$$\Phi_i(t) = f_{\mathrm{fix},i} \cdot D_i(t) \tag{3.33}$$

Crops can only fulfil the remaining fraction of their nitrogen demand by the uptake of soil mineral nitrogen of the soil. Therefore, the demand for soil mineral nitrogen of species i is calculated as:

$$D_{\mathrm{soil},i}(t) = \left(1 - f_{\mathrm{fix},i}\right) \cdot D_i(t) \tag{3.34}$$

To determine whether there is enough soil nitrogen for species i to fulfil its demand, M^3 calculates how much soil nitrogen is available to this species. Each crop species has access to a fraction $f_{\mathrm{ac},i}(t)$ of the soil mineral nitrogen, determined by the actual transpiration rate of species i, $\tau(t)$, relative to the actual transpiration of all species in the intercrop. The actual transpiration rate is calculated as:

$$\tau_{\mathrm{i}}(t) = \sigma_{\mathrm{w,i}}(t) \cdot K_{\mathrm{tr,i}}(t) \cdot E_0(t) \cdot f_{\mathrm{int},i}(t) \tag{3.35}$$

where $\sigma_{\mathrm{w,i}}(t)$ is the fraction of reduction of growth and transpiration due to water stress of species i. $K_{\mathrm{tr,i}}(t)$ is the potential transpiration coefficient of species i at time t; this is the ratio of the potential transpiration rate of crop i to a reference crop as defined by the FAO (Allen *et al.*, 1998). $E_0(t)$ is the reference evapotranspiration rate (m^3 H_2O m^{-2} ground d^{-1}) at time *t*. In a two species intercrop with species i and j, M^3 calculates $f_{\mathrm{ac,i}}(t)$ as:

$$f_{\mathrm{ac},i}(t) = \frac{\tau_i(t)}{\tau_i(t) + \tau_j(t)} \tag{3.36}$$

This version of M^3 assumes that neither of the species i and j experiences water stress (i.e., $\sigma_w(t) = 1$). It is further assumed that the two crops have equal transpiration coefficients ($K_{\mathrm{tr},i}(t) = K_{\mathrm{tr},j}(t)$). Hence, $f_{\mathrm{ac,i}}(t)$ is entirely determined by the relative light interception of species i and simplifies to

$$f_{\mathrm{ac},i}(t) = \frac{f_{\mathrm{int},i}(t)}{f_{\mathrm{int},i}(t) + f_{\mathrm{int},j}(t)} \tag{3.37}$$

The uptake of nitrogen from the soil of species i equals its demand from the soil $D_{\mathrm{soil},i}$, if species i has access to enough soil mineral nitrogen. Else, the uptake is limited to the nitrogen the crop has access to:

$$U_{\mathrm{i}}(t) = \begin{cases} D_{\mathrm{soil,i}}(t) & | \quad D_{\mathrm{soil,i}}(t) \leq f_{\mathrm{acN,i}}(t) \cdot N_{\mathrm{s}}(t) \\ f_{\mathrm{acN,i}}(t) \cdot N_{\mathrm{s}}(i) & | \quad D_{\mathrm{soil,i}}(t) > f_{\mathrm{acN,i}}(t) \cdot N_{\mathrm{s}}(t) \end{cases} \tag{3.38}$$

Finally, M^3 assumes that crops can only lose nitrogen by the death of leaf tissue, but that the nitrogen in the senescing leaves is retranslocated to other plant parts. Therefore, dying leaves are at their minimum nitrogen content and the loss of nitrogen due to leaf senescence is calculated as:

$$M_{N_{\mathrm{c},i}}(t) = \nu_{\mathrm{min,i}}(t) \cdot \mu_{\mathrm{i}}(t) \cdot \frac{L_{\mathrm{i}}(t)}{S_{\mathrm{la,i}}} \tag{3.39}$$

3.2.2.9 Calculation of crop growth reduction factor due to nitrogen stress

The reduction factor of biomass production of species i, $\sigma_{\mathrm{N},i}(t)$, is calculated as:

$$\sigma_{\mathrm{N},i}(t) = \max\left(0, \min\left(1, \frac{\nu_{\mathrm{i}}(t) - \nu_{\mathrm{min,i}}(t)}{\nu_{\mathrm{crit,i}}(t) - \nu_{\mathrm{min,i}}(t)}\right)\right) \quad (3.40)$$

3.3 Model initialization M^3-W

The initial amount of nitrogen in the soil is set equal to the amount of soil mineral nitrogen in the rootable soil layer:

$$N_{\mathrm{s}}(t_0) = N_{\mathrm{s,rt,0}} \quad (3.41)$$

The initial values of the temperature sum from emergence, the leaf area index, the aboveground biomass, the storage organ weight, the crop height and the nitrogen content are 0. At the time of emergence of species $t_{\mathrm{e},i}$, the storage organ weight remains 0, but the remaining state variables are set to:

$$H_i(t_{\mathrm{e},i}) = H_{0,i} \quad (3.42)$$
$$L_i(t_{\mathrm{e,i}}) = A_{0,i} \cdot \rho_i \quad (3.43)$$
$$B_i(t_{\mathrm{e,i}}) = \frac{A_{0,\mathrm{i}} \cdot \rho_i}{S_{\mathrm{la},i}} \quad (3.44)$$
$$N_{\mathrm{c},i}(t_{\mathrm{e},i}) = \frac{A_{0,i} \cdot \rho_i}{S_{\mathrm{la},i}} \cdot \nu_{\mathrm{max,0,i}} \quad (3.45)$$

where $H_{0,i}$ is the height at emergence of species i. $A_{0,i}$ is the leaf area at emergence per plant (m^2 plant^{-1}). ρ_{i} is the sowing density of species i. It is assumed that the nitrogen content of the crop at emergence equals its maximum nitrogen content.

4. Inventory of model components

4.1 Solution and projects

The M^3 model is implemented in the C# programming language (Hejlsberg *et al.*, 2008) within the 4.6.1 .NET framework (Microsoft cooperative). The source code consists of a solution called M3.sln, including three projects. The first two projects are .NET framework Class Libraries of the type Visual C#:

- M3_library.proj

- ModelRunner_library.proj

The third project is a .NET framework Console App of the type Visual C#, named:

- RunSimulationsFromInputFile.proj

This latter project is also the StartUp project and contains the only main function within M3.sln.

4.2 Classes

Figure 1 shows an overview of all the classes in M3.sln, sorted by the project to which they belong and inheritance relationships.

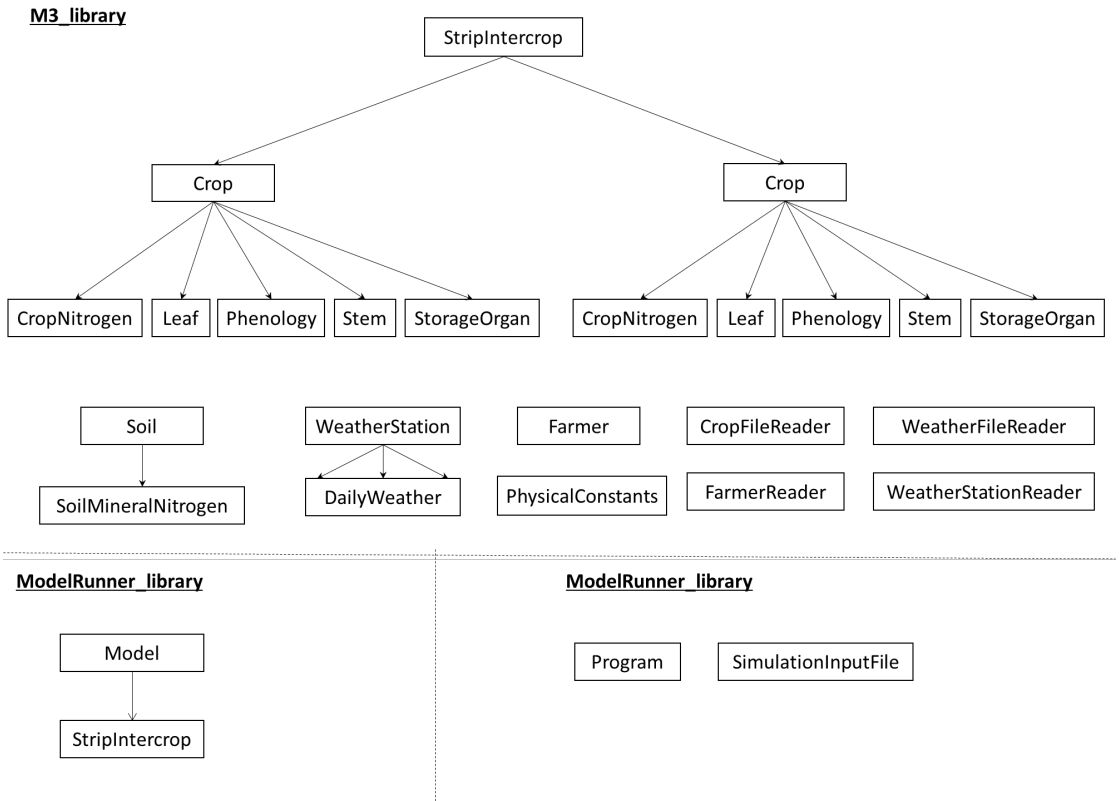

Figure 1: Schematic overview of all projects and their classes within M3.sln. The projects are bold and underlined, separated by dashed lines. The classes are represented as rectangles. An arrow with a filled point from one class to another indicates that the class that is pointing contains an instance of the class that it is pointing at (HAS-A relationship). The class Intercrop points to two Crop objects, as it contains an array of two crop objects. WeatherStation points several times to DailyWeather as this object contains an array with many DailyWeather objects. An arrow with an open point between two classes indicates that the class that is pointing is a superclass of the class it is pointing at.

4.2.1 RunSimulationsFromInputFile

The console app RunSimulationsFromInputFile contains the following classes:

- Program.cs
- SimulationInputFiles.cs

4.2.2 ModelRunner_libary

The class library ModelRunner_libary contains the following classes:

- Program.cs
- SimulationInputFiles.cs

4.2.2 M3_library

The class library M3_library contains the following classes:

- Crop.cs
- CropFileReader.cs
- CropNitrogen.cs
- DailyWeather.cs
- Farmer.cs
- FarmerReader.cs
- Phenology.cs
- PhysicalConstants.cs
- Soil.cs
- SoilFileReader.cs
- SoilMineralNitrogen.cs
- Stem.cs
- StorageOrgan.cs
- StripIntercrop.cs
- StripIntercropReader.cs
- Timer.cs
- WeatherFileReader.cs
- WeatherStation.cs
- WeatherStationReader.cs

4.3 Input files

M^3 obtains the specifications for a particular simulation from various input files:

- Crop files (see Section 7.1 for details)

- Farmer files (Section 7.2)

- SimulationInput files (Section 7.3)

- Soil files (Section 7.4)

- Weather files (Section 7.5)

- Weather station files (Section 7.6)

5. Interactions of the model component during a run

M^3 can be run by setting the project RunSimulationsFromInputFile.proj as the StartUp project and compiling and running the solution M3.sln. Once the project is compiled, the model is run returning model output files, unless an error occurs during compilation or run time.

A simulation can be subdivided into three phases:

1. Initialization phase, during which input files are read. The data from these input files are used to initialize model components;

2. Dynamic phase, during which model calculations are performed for each simulated day. It consists of three steps:

- 2.1 Calculate growth rates, during which the growth rate of each state variable is calculated from the state variables at the start of that simulated date, the model parameters, and the external weather data at that date;

- 2.2 Store state variables, their growth rates, and other model output variables;

- 2.3 Numerical integration and updating of the state variables and the time;

3. Post-processing, when output files are written using the daily values of the output variables that were stored during the dynamic phase

In the next sub sections, we will describe in more detail which methods from various classes are called during each of these phases and in which order.

5.1 Initialization phase

Figure 2 shows a schematic overview of the various input files, the classes that are involved in reading these input files and storing the input data, and how these classes interact during the initialization phase. Simulations are conducted by running the project RunSimulationsFromInputFile. Figures 3 and 4 show an overview of the methods that are called during the initialization phase.

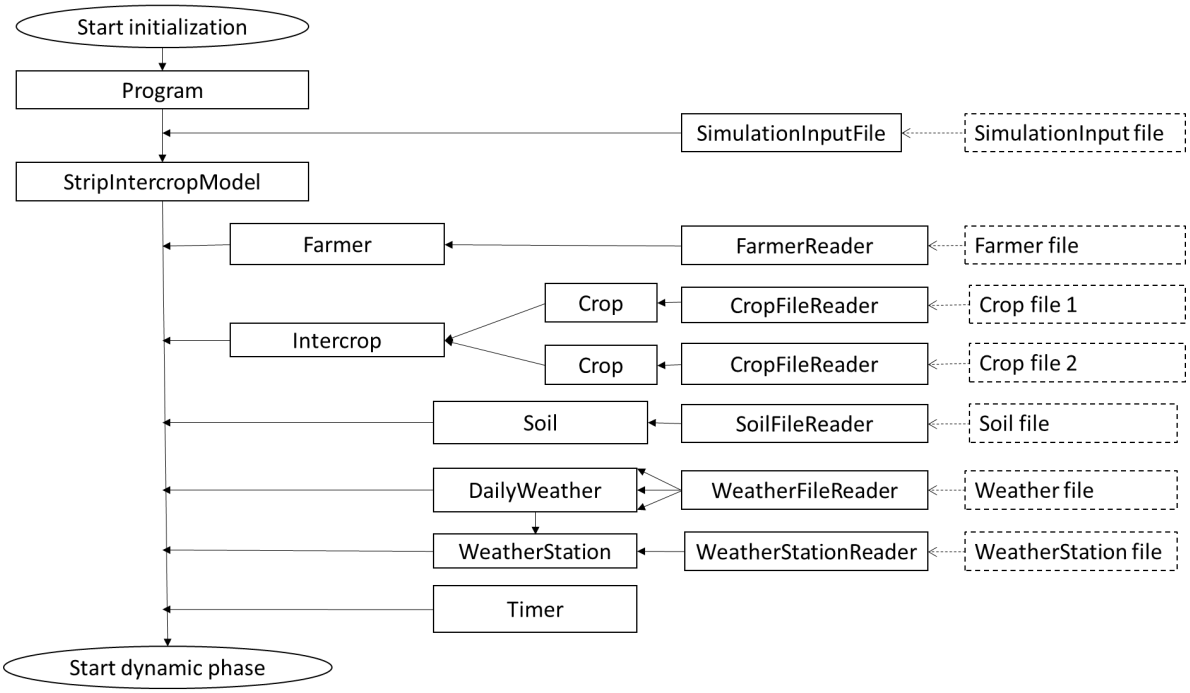

Figure 2: Schematic overview of classes and input files that are involved in the initialization phase and their interactions during this phase. Rectangles with dashed edges represent input files. Rectangles with solid edges represent classes. A dashed arrow that point from an input file to a class indicates that the input file is read by that class. A solid arrow from one class to another one indicates that methods in this class are used in the initialization of the class it is pointing at.

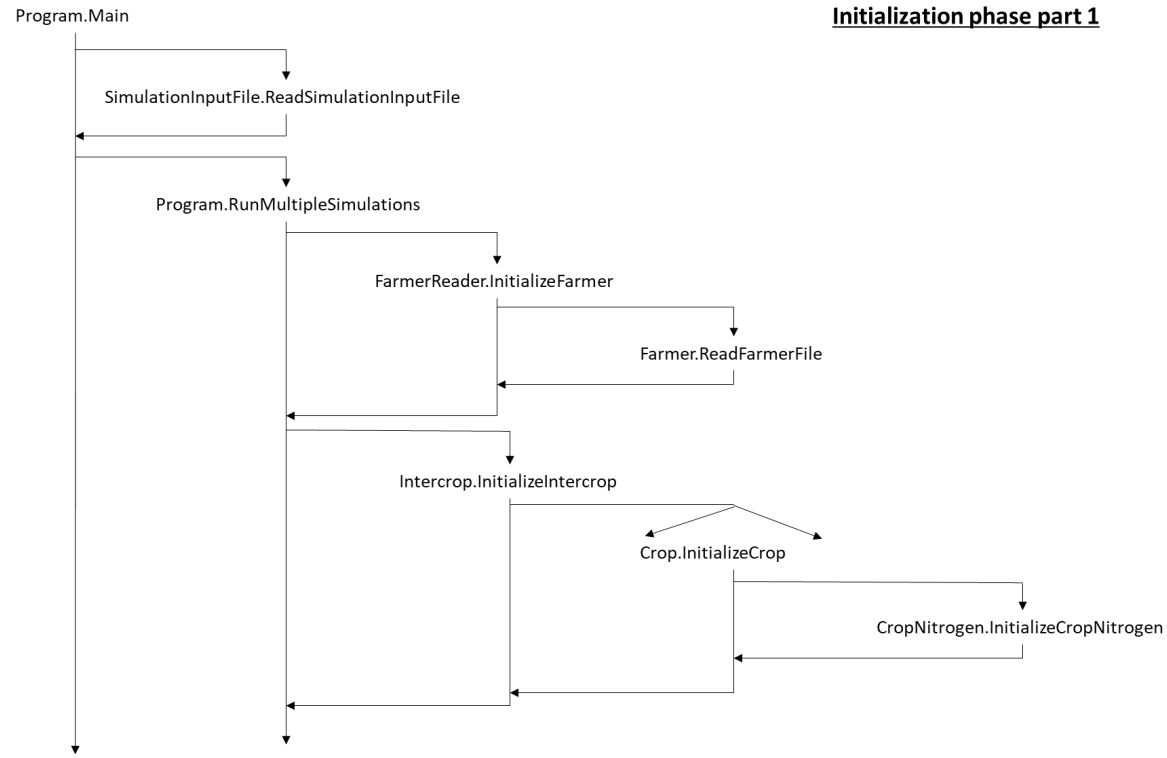

Figure 3: Sequence diagram of the first part of the initialization phase and the different methods in M3.sln. An arrow from one method to another indicates that the pointing method is calling the method that it points at. There are two arrows pointing to Crop.InitializeCrop, because M^3 initializes two crop object here (one for each crop species in the intercorp).

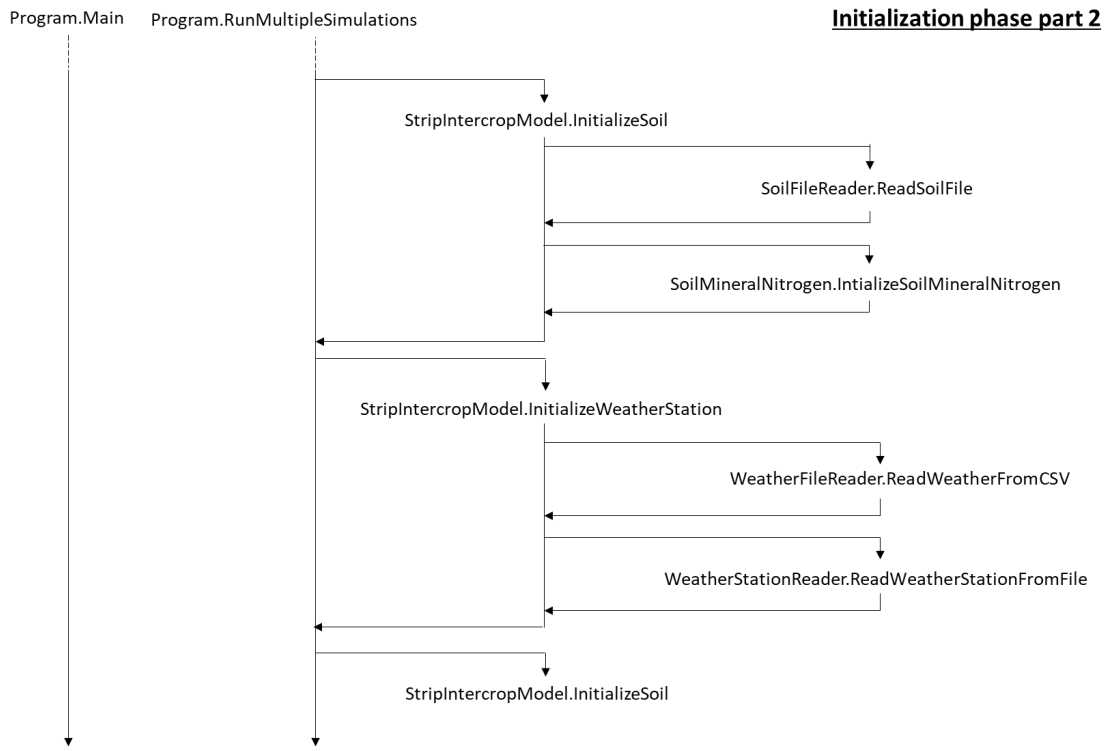

Figure 4: Sequence diagram of the second part of the initialization phase and the different methods in M3.sln. An arrow from one method to another indicates that the method from which the arrow originate is calling the method that it points at. The dashed arrows indicates that methods were previously called.

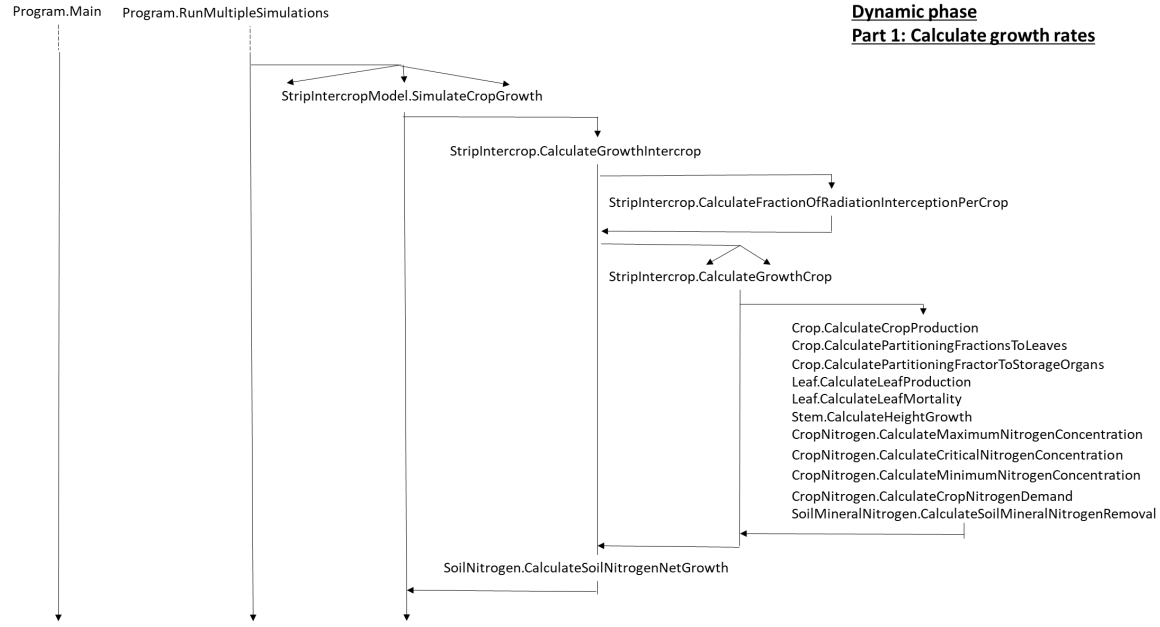

Figure 5: Sequence diagram of the first part of the dynamic phase. The texts represent methods in M3.sln. An arrow from one method to another indicates that the method from which the arrow originate is calling the method that it points at. The dashed arrows indicates that methods were previously called.

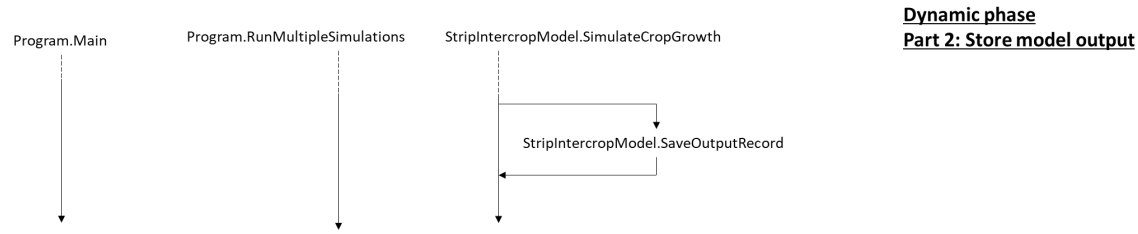

Figure 6: Sequence diagram of the second part of the dynamic phase. The texts represent methods in M3.sln. An arrow from one method to another indicates that the method from which the arrow originate is calling the method that it points at. The dashed arrows indicates that methods were previously called.

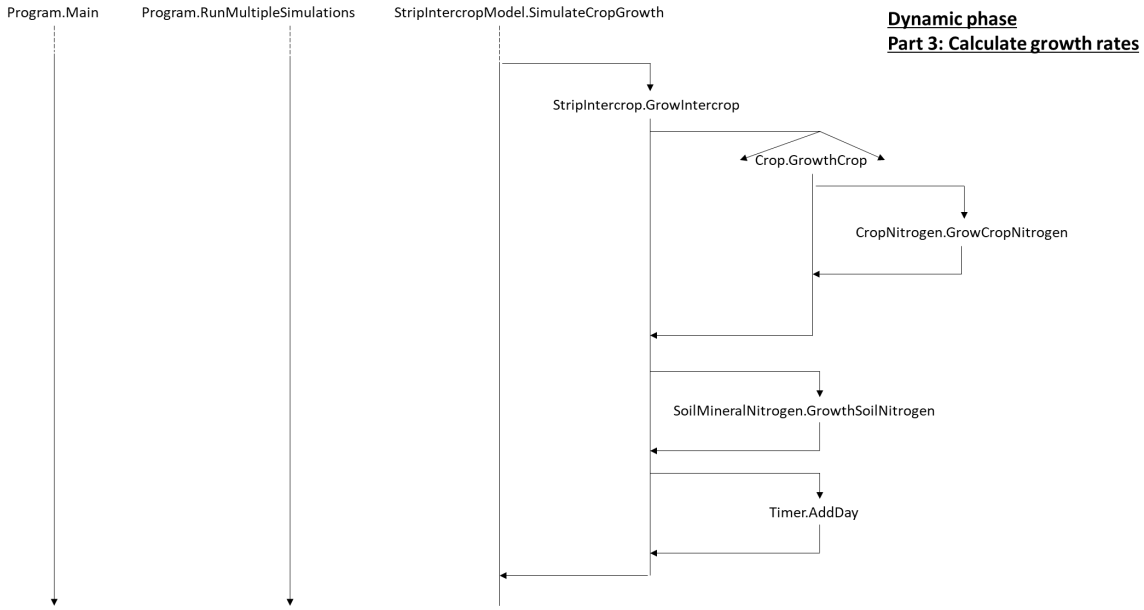

Figure 7: Sequence diagram of the third part of the dynamic phase. The texts represent methods in M3.sln. An arrow from one method to another indicates that the method from which the arrow originate is calling the method that it points at. The dashed arrows indicates that methods were previously called.

Figure 8: Sequence diagram of the post-processing phase. The texts represent methods in M3.sln. An arrow from one method to another indicates that the method from which the arrow originate is calling the method that it points at. The dashed arrows indicates that methods were previously called.

The main function is the function Program.Main. The first commands in the Program.Main function declare and initialize a SimulationInputFile object, which is subsequently used to read a simulation input file. The file path of the simulation input file is hard coded in the method Program.Main and equals:

~/simulationInput.csv

where "~" indicates the root directory, which is the directory in which the executable that was called (runSimulations.exe) is stored. The simulation input file contains the file names of the input files and the

directories where they are stored for each simulation. For each field for which the directories are not provided by simulationInput.csv, M^3 assumes for each input file type that they are stored in the following directories:

Input file type	**Default directory**
Crop input file	~/Crop files/
Farmer input file	~/Farmer files/
Soil input file	~/Soil files/
Weather file	~/Weather files/
Weather station file	~/Weather station files/

The simulation input file consists of several rows and each rows specifies the file names and directories of the input files used in a single simulation. The function Program.Main calls the method Program.RunMultipleSimulations. The aim of this function is to run each simulation that was specified in simulationInput.csv. This function loops through each specified simulation. For each simulation, it declares and initializes a StripIntercropObject. This object contains a number of initialization methods. These methods are called by calling the function StripIntercropModel.InitializeModelComponents. StripIntercrop.InitializeModelComponents calls a number of initialization functions that are called to successively initialize a Farmer object (by calling StripIntercropModel.InitializeFarmer), an StripIntercrop object (by calling StripIntercropModel.InitializeStripIntercrop), a Soil object (by calling StripIntercropModel.InitializeSoil), a WeatherStation object (by calling StripIntercropModel .InitializeWeatherStation), and a Timer object (by calling StripIntercropModel.InitializeTimer). During the initialization of a StripIntercrop object, two Crop objects are initialized as well (by calling StripIntercrop.InitializeCrops). During the initialization of a Soil object, the Soil object initializes a SoilMineralNitrogenObject (by calling Soil.SoilMineralNitrogen.InitializeSoilMineralNitrogen). The WeatherStation object reads weather files during its initialization and stores the data in an array of DailyWeather objects. This array is initialized by calling the function WeatherFileReader.ReadWeatherFromCSV. The initialization phase ends after all commands in StripIntercropModel.InitializeModelComponents have been executed.

5.2 Dynamic phase

The dynamic phase starts when the method Program.RunMultipleSimulations calls the method StripIntercropModel.SimulateCropGrowth. This method calls the method StripIntercropModel.CalculateGrowthIntercrop, which in turn first calls the method StripIntercrop.CalculateFractionOfRadiationInterceptionPerCrop (calculating the fraction of the global radiation that is intercepted by both crop species in the strip intercrop) and then calls the method StripIntercrop.GrowthCrop for each crop species. This latter method calls several methods that calculate the growth rate of the crop species specific state variables as well as the growth rate of the amount of soil mineral nitrogen. After calculating the growth rates, the function StripIntercropModel.SaveOutputRecord saves both the state of the values of the state variables and the growth rates.The growth rates are also used to calculate the values of the state variables in the next time step, by calling the method StripIntercrop.GrowthIntercrop. In turn, this method calls the methods Crop.GrowthCrop and SoilMineralNitrogen.GrowthNitrogen. Finally, the timer Timer.AddDay() increases the time variable with one time step. When the harvest date is reached, the simulation ends.

5.3 Post-processing phase

The post-processing phase starts when the simulated date equals the harvest date of the last crop species of the system that is harvested. During the dynamic phase, model output is stored for each simulated day. This output is used in the post-processing phase to write an output file. First, column

titles are written to a text file by calling the method StripIntercropModel.SaveHeader. Next, the method StripIntercropModel.WriteOutput to write the output records to that same text file.

6 Description of individual model components

The aim of this section is to explain the individual parts of the source code model, including inputs and outputs.

6.1 RunSimulationsFromInputFile.Program.cs

The file Program.cs contains the class Program. It contains the main function of the C# .NET Console application RunSimulationsFromInputFile. If this project is run, the main functino is executed. The class Program has no class variables.

6.1.1 RunSimulationsFromInputFile.Program.Main

Description

The method Program.Main(string[] args) is the main function of the project RunSimulationsFromInputFile. It reads the file simulationInput.csv that is located in the same directory as the executable that results from building the project RunSimulationsFromInputFile. It stores the information from simulationInput.csv in a SimulationInput object simulationInputFile. Program.Main then calls the function RunMultipleSimulations using the object simulationInputFile as an input. This function then runs all the simulations that have been specified in simulationInputFile.

Input

None

Output

None

6.1.2 RunSimulationsFromInputFile.Program.RunMultipleSimulations

Description

The purpose of the method Program.RunMultipleSimulations is to run a batch of simulations for which the input is specified in the input SimulationInputFile object simulationInputFile. It first reads the integer array runIDs from the object simulationInputFile and determines the number of simulations to be run as the length of the array runIDs. It then opens a for loop to run each simulation.

For each simulation specified in simulationInputFile, the method first declares a model object. Then, it uses the information stored in simulationInputFile initializes the directory of a number of class variables of the class Model that represent file paths or a weather station index. These variables are model.farmerFilePath, model.outputFilePath, model.soilFilePat model.weatherStationID, model.weatherStationFilePath, and model.weatherFilePath. Next, it calls the function model.InitializeModelComponents to further initialize the Model object model. Finally, it runs the simulation by calling model.SimulateCropGrowth.

Input

Variable name	Type	Description
simulationInputFile	SimulationInputFile	Contains input data for a batch of simulations that are to be run.

Output

None

6.2 RunSimulationsFromInputFile.SimulationInputFile.cs

	Type	Description
farmerFileDirectories	string array	Contains directories of farmer files for each simulation in the simulation input file
farmerFileNames	string array	Contains files names of farmer files for each simulation in the batch
farmerFilePaths	string array	Contains paths of farmer files for each simulation in the batch
outputFileDirectories	string array	Contains directories of output files for each simulation in the batch
outputFileNames	string array	Contains files names of output files for each simulation in the batch
outputFilePaths	string array	Contains paths of output files for each simulation in the batch
runIDs	int array	Contains the run IDs of each simulation in the batch
simulationInputFileDirectory	string	Contains the directory of the simulation input file
simulationInputFileName	string	Contains the file name of the simulation input file
simulationInputFilePath	string	Contains the file path of the simulation input file
soilFileDirectories	string array	Contains directories of soil files for each simulation in the batch
soilFileNames	string array	Contains files names of soil files for each simulation in the batch
soilFilePaths	string array	Contains paths of soil files for each simulation in the batch
weatherFileDirectories	string array	Contains directories of weather files for each simulation in the batch
weatherFileNames	string array	Contains files names of weather files for each simulation in the batch
weatherFilePaths	string array	Contains paths of weather files for each simulation in the batch
weatherStationFileDirectories	string array	Contains directories of weather station files for each simulation in the batch
weatherStationFileNames	string array	Contains files names of weather station files for each simulation in the batch
weatherStationFilePaths	string array	Contains paths of weather station files for each simulation in the batch
weatherStationIDs	int array	Contains weather station IDs for each simulation in the batch

6.3 ModelRunner.Model.cs

The file Model.cs contains the class Model. The purpose of the class Model is to declare the class variables that are required to set up a simulation. These class variables are inherited by the class StripIntercropModel. It also contains virtual class methods that are inherited by StripIntercropModel. The table below shows an overview of the class variables and their types.

Name	Type	Description
cropFilePaths	string array	Contains the file paths of all crop files
farmer	Farmer	Contains information about the field management of the whole intercrop
farmerFilePath	string	Contains the file path of the farmer file
outputFilePath	string	Contains the file path to which the model output is written
outputRecords	string m x n matrix	Contains the daily records of various model output variables
soil	Soil	Contains soil variables and a SoilNitrogen object
timer	Timer	Contains information about the time
soilFilePath	string	Contains the soil files
weatherStationID	int	Identifier of the weather station
weatherFilePath	string	Contains the file path of the weather file
weatherStation	WeatherStation	WeatherStation object
weatherStationFilePath	string	Contains the file path of the WeatherStationFilePath

6.3.1 ModelRunner.Model.InitializeFarmer()

Description

The method Model.InitializeFarmer() initializes the previously declared class variable farmer and returns it.

Input

None

Output

The method Model.InitializeFarmer() initializes the class variable farmer and returns it.

6.3.2 ModelRunner.Model.InitializeModelComponents

Description

The method Model.InitializeModelComponents() returns the Model object.

Input

None

Output

The method Model.InitializeModelComponents() returns the Model object that calls this function.

6.3.3 ModelRunner.Model.InitializeSoil()

Description

The method Model.InitializeSoil() initializes the previously declared class variable soil and returns it.

Input

None

Output

The method Model.InitializeModelComponents() initializes the class variable soil and returns it.

6.3.4 ModelRunner.Model.InitializeTimer()

Description

The method Model.InitializeTimer() initializes the previously declared class variable timer and returns it.

Input

None

Output

The method Model.InitializeTimer() initializes the class variable soil and returns it.

6.3.5 ModelRunner.Model.InitializeWeatherStation()

Description

The method Model.InitializeWeatherStation() initializes the previously declared class variable timer and returns it.

Input

None

Output

The method Model.InitializeWeatherStation() initializes the class variable soil and returns it.

6.3.6 ModelRunner.Model.SaveHeader()

Description

The method Model.SaveHeader() declares a generic header and intializes it. It returns this header.

Input

None

Output

The method Model.SaveHeader() returns a string array which contains the header.

6.3.7 ModelRunner.Model.SaveOutputRecord()

Description

The method Model.SaveOutputRecord() declares a generic output record and initializes it. It returns this output reader.

Input

None

Output

The method Model.SaveOutputRecord() returns a generic model output record.

6.3.8 ModelRunner.Model.SimulateCropGrowth()

Description

The method Model.SimulateCropGrowth() is an empty method that is supposed to be overwritten by classes that inherit from this class; in this case the class StripIntercropModel.

Input

None

Output

None

6.3.9 ModelRunner.Model.WriteOutput()

Description

The method Model.WriteOutput() is an empty method that is supposed to be overwritten by classes that inherit from this class; in this case the class StripIntercropModel.

Input

Variable name	Type	Description
header	string	Contains the header of an output record
outputRecord	string m x n matrix	Contains the output records

Output

None

6.4 ModelRunner.StripIntercropModel.cs

The file StripIntercropModel.cs contains the class StripIntercropModel. It is a subclass of the class Model, inherits its class variables and overrides various methods from Model. The purpose of StripIntercropModel is to call methods from the classes in the library M3_library in order to 1) read simulation input, 2) run the simulations, 3) to write model output to output files. The table below shows the class variables, their types.

Name	Type	Description
numberOfCrops	int	Number of crops in the intercrop
numberOfDays	int	Total number of days that is simulated.
numberOfSharedOutputVariables	int	Number of output variables that are not specific for crops that are a part of the intercrop.
numberOfOutputVariables	int	Total number of output variables
numberOfCropSpecificVariables	int	Number of output variables that are calculated for each crop in the intercrop.
intercrop	StripIntercrop	StripIntercrop object

6.4.1 ModelRunner.StripIntercropModel.InitializeFarmer

Description

The method StripIntercropModel.InitializeFarmer() initializes the Farmer class object farmer by initializing and declaring a FarmerReader object farmerReader to read information from a farmer input file and store it in farmer. It uses the information in farmer to initialize the class variable cropFilePaths.

6.4.2 ModelRunner.StripIntercropModel.InitializeModelComponents

Description

The method StripIntercrop.InitializeModelComponents() calls a series of class methods to initialize the class objects farmer, intercrop, soil, timer, and weatherStation.

Input

None

Output

StripIntercrop.InitializeModelComponents() returns the Model object that calls it, after it is initialized.

6.4.3 ModelRunner.StripIntercropModel.InitializeSoil

Description

The purpose of the class variable StripIntercropModel.InitializeSoil(string soilFilePath) is to initialize the class Soil object soil. It initializes the class variable soil and then calls the method soil.ReadSoilFile. This methods uses the soil file with file path soilFilePath to further initialize the soil object.

Input

Variable name	**Type**	**Description**
soilFilePath	string	File path of a soil file for which the data are used to initialize the Soil object soil.

Output

StripIntercrop.InitializeSoil(string soilFilePath) returns an initialized Soil object soil. It also initializes the class object soil in the StripIntercropModel class that calls it.

6.4.4 ModelRunner.StripIntercrop.InitializeIntercrop

Description

The method StripIntercrop.InitializeIntercrop(Farmer farmer) declares an intercrop object. It uses the class variable cropFilePaths and the input farmer object to find information to initialize the intercrop object and its Crop array crops.

Input

Variable name	Type	Description
farmer	Farmer	Object that contains management information of an StripIntercropo object

Output

The StripIntercrop.InitializeIntercrop(Farmer farmer) returns an initialized intercrop object.

'

6.4.5 ModelRunner.StripIntercrop.InitializeTimer

Description

The purpose of the method StripIntercrop.InitializeTimer(Farmer farmer) is to declare, initialize and return a Timer object timer. It uses the farmer input object to read the sowing dates and the harvest and subsequently to set the start and the end date of the simulation equal to the sowing and the harvest date, respectively.

Input

Variable name	Type	Description
farmer	Farmer	Object that contains management information of a StripIntercrop object

Output

The method StripIntercrop.InitializeTimer(Farmer farmer) returns an initialized Timer object.

6.4.6 StripIntercrop.InitializeWeatherStation

Description

The method StripIntercrop.InitializeWeatherStation(string weatherFilePath, string weatherStationPath) initializes the WeatherStation class object weatherStation. In order to do so, it first initializes a WeatherFileReader and a WeatherStationReader object. Methods in these objects are used to, respectively, initialize a WeatherStation object weatherStation and an array of DailyWeather objects dailyWeatherRecords. For this purpose, information from the input strings weatherFilePath and weatherStationPath is exploited. The class variable weatherStation.weatherRecords is set equal to the array dailyWeatherRecords. The initialized Weathe rStation object weatherStation is returned.

Input

Variable name	Type	Description
weatherFilePath	string	File path of a weather file for which the data are used to initialize the weather station

weatherStationPath	string	File path that contains information of the WeatherStation object weather station that is initialized

Output

The method StripIntercrop.InitializeWeatherStation(string weatherFilePath, string weatherStationPath) returns the WeatherStation object that was initialized.

6.4.7 StripIntercropModel.SaveHeader()

Description

The purpose of the method StripIntercropModel.SaveHeader() is to store the column titles of a model output file return these titles. The title of each column is hard coded. First, the method initializes a string array header that holds the column titles. Each entry contains a column title. It initializes each string in header and returns it.

Input

None

Output

StripIntercropModel.SaveHeader() returns a string array. The entries of the string array contain the column titles of the output file.

6.4.8 ModelRunner.StripIntercropModel.SaveOutputRecord()

The purpose of the method StripIntercropModel.SaveOutputReord() is to collect all output variables and store them in a output variable record. First, it initializes a string array outputRecord to store all output of a simulated day. The size of this array equals the total number of output variables. Second, it collects all output variables that are not specific to one crop species (output variables from the Timer object timer and from the Soil object soil). Third, it collects crop specific output variables for each crop species in the intercrop and stores them in the string array outputRecord as well. Finally, the output record is returned.

Input

None

Output

The method StripIntercropModel.SaveOutputReord() returns a string array that contains the model output variables of one simulated day.

6.4.9 ModelRunner.StripIntercropModel.SimulateCropGrowth

Description

The purpose of the method StripIntercropModel.SimulateCropGrowth() is to run simulations and generate output files to store the results of these simulations. It first collects input data for the simulations. These data include of fertilization dates and fertilier amounts from the Farmer object farmer. They also include the start date of the simulation (timer.StartSimulationDate) and the end date of the simulation (timer.EndSimulationDate) from the Timer object timer. These dates are used to calculate the total number of simulated days. Next, the total number of output variables is calculated from the number of crops, the number of output variables that are not specified per crop species, and the number of output variables that are specified per crop. A list of string arrays (outputRecordList) is initialized in order to store the daily output records.

Next, the actual calculations begin using a while loop. For each date between the start and the end of the simulation, the daily weather data are obtained from the DailyWeather class variable dailyWeather. At each simulated day, the amount of mineral nitrogen that is added to the soil mineral nitrogen is initially set to 0. Then, it is checked whether the simulated day is one of the days at which fertilization is applied (i.e. the current date is equal to one of the dates in the array fertilizationDates). If this is the case, the amount of mineral nitrogen that is applied at that date is updated. The effective temperature of each crop species is calculated. If the crop species has not emerged before the simulated day and the temperature sum from sowing is larger than the parameter TemperatureSumSowingToEmergence, the crop emerges at that day (i.e. the crop class variable isEmerged is set to true). If the simulated date equals the crop class variable harvest date (harvestDate), the crop is harvested. Next, the growth rate of the state variables is calculated. The output variables are then stored in a string array outputRecord, which is added to the string list outputRecordList. Once the end of the simulation is reached, the while loop closes, the list of output records (outputRecordList) is converted in a string array (outputRecords) and the method WriteOutput is called to write the content of outputRecords to an output file.

Input

None

Output

None

6.4.10 ModelRunner.StripIntercrop.WriteOutput

Description

The method StripIntercrop.WriteOutput() writes a model output file. The data for this output file are obtained from the jagged string array input variable outputRecords and the column titles of the model output file are obtained from the string array header. It opens a StreamWriter object sw. The streamWriter first writes a line for the header; each column is separated by a comma. Next, sw writes for each day a line with the values of all output variables below the header. The values of the output variables are separated by commas.

Input

Variable name	Type	Description
header	string array	String array for which the column titles contain the column titles of the output file
outputRecords	string jagged array	2-dimensional string array with model output. Each column represents an output variable. Each row represents the output for a certain date

6.5 M3_library.Crop

The file Crop.cs contains the class Crop. The class Crop contains initial conditions, state variables, intermediate variables, rates of change, indices and parameters at the crop level. Each Crop object contains instances of the classes CropNitrogen, Leaf, Phenology, Stem, and StorageOrgan. The class Crop.cs contains also functions that update the state variables that are hold by these classes. The Crop class containsproperties for each class variable. Finally, it contains methods to calculate the biomass production and mortality at the crop level. The tables below show the class variables, their type, and their units.

Name	Type	Description	Unit
cropID	int	Identifier of crop within the group of species that are intercropped	
cropNitrogen	CropNitrogen	CropNitrogen object	
abovegroundDryWeight	double	Aboveground dry weight	kg m^{-2}
abovegroundBiomassProduction	double	Aboveground dry weight production rate	kg m^{-2} d^{-1}
abovegroundBiomassMortality	double	Aboveground dry weight mortality rate	kg m^{-2} d^{-1}
developmentStateDeclinePartitioningToLeaves	double	Developmental stage above which the partitioning fraction to the leaves starts to decline	
developmentStateFullPartitioningToStorageOrgansAndLeaves	double	Developmental stage above which all biomass is partitioned to the storage organs	
developmentStateIncreasePartitioningToStorageOrgans	double	Developmental stage above which biomass is partitioned to the storage organs	
developmentStateNoPartitioningToLeaves	double	Developmental stage above which no more biomass is partitioned to the leaves	
fractionOfLightInterception	double	Fraction of global radiation that is intercepted by the crop	
harvestDate	DateTime	Harvest date	
isCropEmerged	bool	Indicates whether the crop has emerged	
isCropHarvested	bool	Indicates whether the crop is harvested	
leaf	Leaf	Leaf object	
partitioningFractionToLeaves	double	Fraction of newly produced biomass that is partitioned to the leaves	
partitioningFractionToStorageOrgans	double	Fraction of newly produced biomass that is partitioned to the storage organs	
partitioningFractionToLeavesBeforeDecline	double	Fraction of newly produced biomass that is partitioned until the developmental stage is reached at which it start to decline.	
phenology	Phenology	Phenology object	
sowingDensity	double	Sowing density	m^{-2}
stripWidth	double	Strip width	m
stem	Stem	Stem object	
storageOrgan	StorageOrgan	StorageOrgan object	

6.5.1 M3_library.Crop.CalculateCropProduction

Description

CalculateCropProduction(DailyWeather dw) calculates the fraction of photosynthetic radiation by crop species *i* that is intercepted by the crop as:

$$I_{\mathrm{int,i}}(t) = f_{\mathrm{par}} \cdot f_{\mathrm{i}}(t) \cdot I_0(t)$$

Symbol	Description	Unit
$f_{\mathrm{i}}(t)$	Fraction of global radiation that is intercepted by species *i*	
f_{par}	Fraction of photosynthetic radiation in global radiation	
$I_0(t)$	Global radiation	J m^{-2} d^{-1}
$I_{\mathrm{int,i}}(t)$	Amount of intercepted light by species *i* at time *t*	J m^{-2} d^{-1}
t	Time	d

Input

Variable name	Type	Description
dw	DailyWeather	DailyWeather object

Output

CalculateCropProduction(DailyWeather dw) updates its state and returns the updated crop object.

6.5.2 M3_library.Crop.CalculatePartitioningFractionToLeaves

Description

The function CalculatePartitioningFractionToLeaves() calculates what fraction of newly produced biomass is partitioned to the leaves. Before the crop has emerged, no biomass is assigned to the leaves. At the day of crop emergence, the fraction of newly produced biomass is set equal to the value of the double variable $f_{\mathrm{L,0,i}}$. This fraction remains constant until the developmental stage is equal to $d_{1,i}$. From this developmental stage, the fraction of biomass partitioned to the leaves declines linearly until the developmental stage equals $d_{2,\mathrm{i}}$ at which point no more biomass is partitioned to the leaves.

$$f_{\mathrm{L,i}}(t) = \begin{cases} 0 & | \quad \delta_{\mathrm{i}}(t) = 0 \\ f_{\mathrm{L,0,i}} & | \quad 0 \leq \delta_{\mathrm{i}}(t) < d_{1,i} \\ f_{\mathrm{L,0,i}} \cdot \dfrac{d_{2,i} - \delta_{\mathrm{i}}(t)}{d_{2,i} - d_{1,\mathrm{i}}} & | \quad d_{1,i} \leq \delta_{\mathrm{i}}(t) < d_{2,i} \\ 0 & | \quad \delta_{\mathrm{i}}(t) > d_{\mathrm{i},2} \end{cases}$$

Symbol	Description	Unit
$f_{\mathrm{L},0,\mathrm{i}}$	Fraction of newly produced biomass partitioned to the leaves after emergence until developmental stage $\delta_{1,i}$ is reached	
$f_{\mathrm{L},\mathrm{i}}(t)$	Fraction of newly produced biomass that is partitioned to the leaves at time t	
$\delta_{\mathrm{i}}(t)$	Developmental stage	
$d_{1,i}$	Developmental stage above which the fraction of newly produced biomass partitioned to the leaves starts to decline	
$d_{\mathrm{i},2}$	Developmental stage above which no more newly produced biomass is partitioned to the leaves	

Input

None

Output

CalculatePartitioningFractionToLeaves () updates its state and returns the updated crop object.

6.5.3 M3_library.Crop.CalculatePartitioningFactorToStorageOrgans()

The function Crop.CalculatePartitioningFactorToStorageOrgans() calculates what fraction of newly produced biomass is assigned to the storage organs. Before the developmental stage $d_{4,\mathrm{i}}$ is reached, no biomass is partitioned to the storage organs. After this developmental stage has been reached, the fraction of biomass that is partitioned to the storage organs starts to increase. From the developmental stage $d_{4,\mathrm{i}}$, all biomass is partitioned to the storage organs and the leaves until phyisological maturity is reached.

$$f_{\mathrm{y},\mathrm{i}}(t) = \begin{cases} 0 & | \quad d_{\mathrm{i}}(t) < d_{3,\mathrm{i}} \\ \left(1 - f_{\mathrm{L},\mathrm{i}}(t)\right) \cdot \dfrac{\delta_{\mathrm{i}}(t) - d_{3,\mathrm{i}}}{d_{4,\mathrm{i}} - d_{3,\mathrm{i}}} & | \quad d_{3,i} < \delta_{\mathrm{i}}(t) < d_{4,\mathrm{i}} \\ 1 - f_{\mathrm{L},\mathrm{i}}(t) & | \quad \delta_{\mathrm{i}}(t) > d_{4,\mathrm{i}} \end{cases}$$

Symbol	Description	Unit
$f_{\mathrm{y},\mathrm{i}}(t)$	Fraction of biomass assigned to the storage organs at time t	
$\delta_{\mathrm{i}}(t)$	Developmental stage	
i	Species index	
t	Time	
$d_{3,i}$	Developmental stage below which no newly produced biomass is assigned to the storage organs	

$d_{4,\mathrm{i}}$	Developmental stage above which all newly produced biomass is assigned to the storage organs and the leaves	

Input

None

Output

CalculatePartitioningFactorToStorageOrgans() updates its state and returns itself.

6.5.4 M3_library.Crop.CalculateCropGrowth

The purpose of the function Crop.CalculateCropGrowth (StripIntercrop ic, Soil s, Timer t, WeatherStation w, DailyWeather dw) is to call all a sequence of functions that calculate the rates of change of the Crop object itself and the state of its class variables leaf, storageOrgan, stem, cropNitrogen, and phenology.

Input

Variable name	**Type**	**Description**
ic	StripIntercrop	The StripIntercropObject from which the Crop object is a part
s	Soil	The SoilObject that describes the state of the soil at which the crop object is grown.
t	Timer	Timer object
w	WeatherStation	WeatherStation object
dw	DailyWeather	DailyWeather object

Output

Crop.CalculatePartitioningFactorToStorageOrgans() updates its state and returns the updated crop object.

6.5.5 M3_library.Crop.EmergeCrop()

The purpose of the function Crop.EmergeCrop(StripIntercrop ic) is to set the state of the Crop object itself and its leaf, stem, and cropNitrogen class variables equal to their state at emergence. It is assumed that the initial crop nitrogen concentration equals $\nu_{\mathrm{max},0}$.

$$H_{\mathrm{i}}(t) = H_{0,\mathrm{i}}$$

$$L_{\mathrm{i}}(t) = A_{0,\mathrm{i}} \cdot \rho_{\mathrm{i}}$$

$$B_{\mathrm{i}}(t) = \frac{A_{0,\mathrm{i}} \cdot \rho_{\mathrm{i}}}{S_{\mathrm{la,i}}}$$

$$N_{\mathrm{c,i}}(t) = \frac{A_{0,\mathrm{i}} \cdot \rho_{\mathrm{i}}}{S_{\mathrm{la,i}}} \cdot \nu_{\mathrm{max},0,\mathrm{i}}$$

Symbol	Description	Unit
$A_{0,\mathrm{i}}$	Leaf area per plant at emergence	m^2
$B_{\mathrm{i}}(t)$	Aboveground dry weight at time t	kg m^{-2}
$H_{0,\mathrm{i}}$	Crop height at emergence	m
$H_{\mathrm{i}}(t)$	Crop height at time t	m
i	Crop species index	
$N_{\mathrm{c,i}}(t)$	Crop nitrogen amount at time t	kg m^{-2}
$S_{\mathrm{la,i}}$	Specific leaf area	m^2 kg^{-1}
t	Time	d
$\nu_{\mathrm{max,0,i}}$	Crop nitrogen concentrations at low biomasses	kg kg^{-1}
ρ_{i}	Sowing density	m^{-2}

Input

Variable name	Type	Description
ic	StripIntercrop	The StripIntercropObject of which the Crop object is a part

Output

Crop.EmergeCrop () updates its state and returns the updated crop object.

6.5.6 M3_library.Crop.GrowCrop

The purpose of the method Crop.GrowCrop() is to calculate the state of the crop and its leaf, stem, and storageOrgan class variables at the next time step and set the state of these objects equal to those values. It is always assumed that the time step is one day. It also calls a function that calculates the new state of the cropNitrogen class variable and updates this state.

$$B_{\mathrm{i}}(t+\Delta t) = B(t) + \Delta t \cdot \left(P_{\mathrm{B,i}}(t) - M_{\mathrm{B,i}}(t)\right)$$
$$L_{\mathrm{i}}(t+\Delta t) = L_{\mathrm{i}}(t) + \Delta t \cdot \left(P_{\mathrm{L,i}}(t) - M_{\mathrm{L,i}}(t)\right)$$
$$B_{\mathrm{BL,i}}(t+\Delta t) = B_{\mathrm{BL,i}}(t) + \Delta t \cdot \left(P_{\mathrm{BL,i}}(t) - M_{\mathrm{BL,i}}(t)\right)$$
$$H_{\mathrm{i}}(t+\Delta t) = H_{\mathrm{i}}(t) + \Delta t \cdot P_{\mathrm{H,i}}(t)$$
$$Y_{\mathrm{i}}(t+\Delta t) = Y_{\mathrm{i}}(t) + \Delta t \cdot P_{\mathrm{Y,i}}(t)$$

Symbol	Description	Unit
$B(t)$	Aboveground dry weight at time *t*	kg m^{-2}
$B_{\mathrm{BL,i}}(t)$	Leaf dry weight at time *t*	kg m^{-2}
$H_{\mathrm{i}}(t)$	Crop height at time *t*	m
i	Species index	
$L_{\mathrm{i}}(t)$	Leaf area index mortality	
$M_{\mathrm{B,i}}(t)$	Aboveground dry weight mortality	kg m^{-2} d^{-1}
$M_{\mathrm{BL,i}}(t)$	Leaf dry weight mortality	kg m^{-2} d^{-1}
$M_{\mathrm{L,i}}(t)$	Leaf area index mortality	d^{-1}
$P_{\mathrm{B,i}}(t)$	Daily production of aboveground dry weight at time *t*	kg m^{-2} d^{-1}
$P_{\mathrm{H,i}}(t)$	Daily height growth	m d^{-1}
$P_{\mathrm{L,i}}(t)$	Daily production of leaf area index	d^{-1}
$P_{\mathrm{Y,i}}(t)$	Daily production of storage organ dry weight at time *t*	kg m^{-2} d^{-1}
t	Time	d
$Y_{\mathrm{i}}(t)$	Storage organ weight at time *t*	kg m^{-2}
Δt	Time step	d

Input

None

Output

Crop.GrowCrop() calculates the new state of the crop and returns itself.

6.5.7 M3_library.Crop.HarvestCrop

The purpose of the method Crop.HarvestCrop() is to calculate the state of the crop and its leaf, stem and storageOrgan class variables after the crop is harvested and set the state of these objects equal to those values.

$$B_{\mathrm{BL,i}}(t) = 0$$
$$H_{\mathrm{i}}(t) = 0$$
$$L_{\mathrm{i}}(t) = 0$$
$$B_{\mathrm{i}}(t) = 0$$
$$Y_{\mathrm{i}}(t) = 0$$

Variable	Description	Unit
$B_{\mathrm{i}}(t)$	Aboveground dry weight at time *t*	kg m^{-2} d^{-1}
$B_{\mathrm{BL,i}}(t)$	Leaf dry weight at time *t*	kg m^{-2} d^{-1}
$H_{\mathrm{i}}(t)$	Crop height at time *t*	kg m^{-2} d^{-1}
i	Crop species index	
$L_{\mathrm{i}}(t)$	Leaf area index at time *t*	
t	Time	d
$Y_{\mathrm{i}}(t)$	Storage organ weight at time *t*	kg m^{-2} d^{-1}

Input

None

Output

Crop.HarvestCrop() calculates the new state of the crop and returns the updated crop object.

6.5.8 M3_library.Crop.InitializeCrop

The purpose of Crop.InitializeCrop() is to initialize the Crop object at the start of the simulation and the leaf, phenology, stem, and storageOrgan class variables of the crop. It also calls a function to initialize the cropNitrogen class variable.

$B_{\mathrm{i}}(t) = 0$
$P_{\mathrm{B,i}}(t) = 0$
$M_{\mathrm{B,i}}(t) = 0$
$L_{\mathrm{i}}(t) = 0$
$B_{\mathrm{BL,i}} = 0$
$P_{\mathrm{L,i}}(t) = 0$
$M_{\mathrm{L,i}}(t) = 0$
$\delta_{\mathrm{i}}(t) = 0$
$S_{\mathrm{i}}(t) = 0$
$H_{\mathrm{i}}(0)$
$P_{\mathrm{H,i}}(0)$
$P_{\mathrm{Y,i}}(0)$
$Y_{\mathrm{i}}(t) = 0$

Symbol	**Description**	**Unit**
$B(t)$	Aboveground dry weight at time *t*	kg m^{-2}
$B_{\mathrm{BL,i}}(t)$	Leaf dry weight at time *t*	kg m^{-2}
$H_{\mathrm{i}}(t)$	Crop height at time *t*	m
i	Species index	
$L_{\mathrm{i}}(t)$	Leaf area index	
$M_{\mathrm{B,i}}(t)$	Aboveground dry weight mortality	kg m^{-2} d^{-1}
$M_{\mathrm{L,i}}(t)$	Leaf area index mortality	d^{-1}
$P_{\mathrm{B,i}}(t)$	Daily production of aboveground dry weight at time *t*	kg m^{-2} d^{-1}
$P_{\mathrm{H,i}}(t)$	Daily height growth	m d^{-1}
$P_{\mathrm{L,i}}(t)$	Daily production of leaf area index	d^{-1}
$P_{\mathrm{Y,i}}(t)$	Daily production of storage organ dry weight at time *t*	kg m^{-2} d^{-1}
$S_{\mathrm{i}}(t)$	Temperature sum from sowing at time *t*	°C d
t	Time	d
$Y_{\mathrm{i}}(t)$	Storage organ weight at time *t*	kg m^{-2}
$\delta_{\mathrm{i}}(t)$	Developmental state at time *t*	

Input

None

Output

Crop.InitializeCrop() calculates the new state of the crop and returns the updated crop object.

6.6 M3_library.CropFileReader

The file CropFileReader.cs contains the class CropFileReader. The purpose of CropFilerReader is to read parameter values from crop input files and use those values to initialize one Crop object. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Variable name	Type	Description	Unit
baseTemperature	double	Average daily below which the temperature sum from sowing does not increase	^{o}C
coefficientACriticalNitrogenConcentration	double	Biomass at which the critical crop nitrogen concentration would be 1, in case the critical nitrogen concentration would not be assumed constant at low biomass	$kg\ m^{-2}$
coefficientAMaximumNitrogenConcentration	double	Biomass at which the maximum crop nitrogen concentration would be 1, in case the maximum nitrogen concentration would not be assumed constant at low biomass	$kg\ m^{-2}$
coefficientAMinimumNitrogenConcentration	double	Biomass at which the minimum crop nitrogen concentration would be 1, in case the minimum nitrogen concentration would not be assumed constant at low biomass	$kg\ m^{-2}$
coefficientBCriticalNitrogenConcentration	double	Shape parameter of the nitrogen dilution curve for the critical concentration	
coefficientBMaximumNitrogenConcentration	double	Shape parameter of the nitrogen dilution curve for the maximum concentration	
coefficientBMinimumNitrogenConcentration	double	Shape parameter of the nitrogen dilution curve for the minimum concentration	
developmentStateDeclinePartitioningToLeaves	double	Developmental state above which the fraction of newly produced biomass assigned to the leaves starts to decrease	
developmentStateNoPartitioningToLeaves	double	Developmental state above which no newly produced biomass is assigned to the leaves	
developmentStateFullPartitioningToStorageOrgansAndLeaves	double	Developmental rate above which all newly produced biomass is partitioned to the leaves and storage organs	
developmentStateIncreasePartitioningToStorageOrgansAndLeaves	double	Developmental state above which all newly produced biomass is partitioned to the leaves and storage organs	
extinctionCoefficient	double	Light extinction coefficient of the canopy	
fractionCriticalToMaximumNitrogenConcentration	double	Fraction of the maximum critical nitrogen concentration to the maximum nitrogen concentration	
fractionMinimumToMaximumNitrogenConcentration	double	Fraction of the maximum minimum nitrogen concentration to the maximum nitrogen concentration	
fractionOfNitrogenDemandFixed	double	Fraction of the crop nitrogen demand that can be acquired by N2 fixation	
initialCropHeight	double	Crop height at emergence	m
initialLeafArea	double	Leaf area at emergence	m^2
maximumCropHeight	double	Maximum crop height	m
maximumCropNitrogenConcentrationEarlyGrowth	double	Maximum nitrogen concentration at low biomasses	$kg\ kg^{-1}$

minimumTemperatureSenescence	double	Minimum temperature at which leaf senescence starts	^{o}C
partitioningFractionToLeavesBeforeDecline	double	Fraction of newly produced biomass that is partitioned to the leaves from emergence until the developmental stage where this fraction starts to decrease.	
radiationUseEfficiency	double	Radiation use efficiency (of photosynthetically active radiation)	$J\ m^{-2}\ d^{-1}$
relativeGrowthRateHeight	double	Relative height growth rate	$^{o}C\ d$
slopeSenescenceTemperature	double	Slope of the relationship between the senescence rate and the average daily temperature	$^{o}C\ d$
sowingDensity	double	Sowing density	$seed\ m^{-2}$
specificLeafArea	double	Specific leaf area	$m^2\ kg^{-1}$
temperatureSumSowingToEmergence	double	Temperature sum from sowing to emergence	$^{o}C\ d$
temperatureSumEmergenceToFlowering	double	Temperature sum from emergence to flowering	$^{o}C\ d$
temperatureSumFloweringToMaturity	double	Temperature sum from flowering to maturity	$^{o}C\ d$

6.6.1 M3_library.CropFileReader.ReadCropFile

The method CropFileReader.ReadCropFile(string filePath) reads a crop file, extracts crop parameters values from the file, assigns the parameter values to a Crop object and returns this Crop object. In order to do this, it first declares a StreamReader object sr. The string input variable filePath of the function CropFileReader.ReadCropFile(string filePath) specifies the path of the Crop file that is read by the StreamReader object sr. Next, sr reads each line of the Crop file. The first column (index number 0) of each crop file includes a symbol that represents a parameter. The second column (index number 1) contains a string that describes the parameter. The third column (index number 2) contains the parameter value. The fourth column (index number 3) contains the unit. For each line, it is checked whether the line contains the parameter symbol using switch-case statements. If it does, its parameter value is stored in the corresponding class variables in CropFileReader. After sr reads the last line of the crop file, a Crop object is declared and initialized. After initialization, CropFileReader.ReadCropFile returns the Crop object.

Input

Variable name	**Type**	**Description**
filePath	String	The path that describes the location of the cropFile that is to be read.

Output

CropFileReader.ReadCropFile returns the Crop object that it initialized with the parameter values in a crop file.

6.7 M3_library.CropNitrogen.cs

The file CropNitrogen.cs contains the class CropNitrogen. The class CropNitrogen contains initial variables, intermediate variables, parameters and a state variable. The CropNitrogen class contains methods that update the state of the object. Finally, the CropNitrogen class contains properties. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Name	Type	Description	Unit
fractionCriticalToMaximumNitrogenConcentration	double	Fraction of critical nitrogen concentration to maximum nitrogen concentration at low biomass	
fractionMinimumToMaximumNitrogenConcentration	double	Fraction of minimum nitrogen concentration to maximum nitrogen concentration at low biomass	
fractionOfNitrogenDemandFixed	double	Fraction of the nitrogen demand that can be obtained by N_2 fixation.	
coefficientACriticalNitrogenConcentration	double	Biomass at which the critical crop nitrogen concentration would be 1, in case the critical nitrogen concentration would not be assumed constant at low biomass	kg m^{-2}
coefficientAMaximumNitrogenConcentration	double	Biomass at which the maximum crop nitrogen concentration would be 1, in case the maximum nitrogen concentration would not be assumed constant at low biomass	kg m^{-2}
coefficientAMinimumNitrogenConcentration	double	Biomass at which the minimum crop nitrogen concentration would be 1, in case the minimum nitrogen concentration would not be assumed constant at low biomass	kg m^{-2}
coefficientBCriticalNitrogenConcentration	double	Shape parameter of the nitrogen dilution curve	
coefficientBMaximumNitrogenConcentration	double	Shape parameter of the nitrogen dilution curve	
coefficientBMinimumNitrogenConcentration	double	Shape parameter of the nitrogen dilution curve	
cropNitrogenAmount	double	Amount of nitrogen in crop	kg m^{-2}
cropNitrogenDemand	double	Total nitrogen demand	kg m^{-2}
cropNitrogenDemandFromSoil	double	Part of the nitrogen demand that needs to be obtained by soil mineral nitrogen uptake	kg m^{-2}
cropNitrogenFixationRate	double	Rate of nitrogen fixation	kg m^{-2} d^{-1}
cropNitrogenGrowth	double	Rate of crop nitrogen accumulation	kg m^{-2} d^{-1}
cropNitrogenConcentration	double	Crop nitrogen concentration	kg kg^{-1}
cropNitrogenLoss	double	Rate of crop nitrogen loss	kg m^{-2} d^{-1}
criticalNitrogenConcentration	double	Critical crop nitrogen concentration	kg kg^{-1}
minimumNitrogenConcentration	double	Minimum crop nitrogen concentration	kg kg^{-1}
maximumNitrogenConcentration	double	Maximum crop nitrogen concentration	kg kg^{-1}
maximumNitrogenConcentrationEarlyGrowth	double	Maximum nitrogen concentration at low biomass	kg kg^{-1}
nitrogenGrowthReductionFactor	double	Growth reduction factor due to nitrogen stress	
nitrogenUptakeFromSoil	double	Rate of nitrogen uptake from the soil by the crop	kg m^{-2} d^{-1}
timeCoefficient	double	Time coefficient	d^{-1}

6.7.1 M3_library.CropNitrogen.CalculateCriticalNitrogenConcentration

The function CropNitrogen.CalculateCriticalCropNitrogenConcentration(StripIntercrop ic, Crop c) calculates the CropCriticalNitrogenConcentration as:

$$\ell_{\mathrm{tot}} = \sum_{\mathrm{j}=1}^{n_{\mathrm{crop}}} \ell_{\mathrm{j}}$$

$$\nu_{\mathrm{crit,i}}(t) = f_{\mathrm{crit,i}} \cdot \nu_{\mathrm{max,0,i}} \cdot \min\left(1, \left(\frac{B_{\mathrm{i}}(t)}{\alpha_{\mathrm{crit,i}} \cdot \frac{\ell_{\mathrm{i}}}{\ell_{\mathrm{tot}}}}\right)^{-\beta_{\mathrm{crit,i}}}\right)$$

Symbol	Description	Unit
$B_{\mathrm{i}}(t)$	Aboveground dry weight	kg m^{-2}
$f_{\mathrm{crit,i}}$	Fraction of critical to maximum nitrogen concentration at low biomasses	
i	Species index	
j	Species index	
ℓ_{i}	Strip width	m
ℓ_{tot}	Sum of strip width of all crops in the intercrop	m
n_{crop}	Number of crop species in the intercrop	
$\alpha_{\mathrm{crit,i}}$	Aboveground dry weight at which the critical crop nitrogen concentration would be 1, in case the crop critical nitrogen concentration would not be assumed to be constant at low biomass	kg m^{-2}
$\beta_{\mathrm{crit,i}}$	Shape parameter critical nitrogen dilution curve	
$\nu_{\mathrm{crit,i}}(t)$	Critical nitrogen concentration	kg kg^{-1}
$\nu_{\mathrm{max,0,i}}$	Maximum nitrogen concentration at low biomass	kg kg^{-1}

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs
ic	StripIntercrop	The StripIntercropObject from which the Crop object is a part

Output

CropNitrogen.CalculateCriticalNitrogenConcentration(StripIntercrop ic, Crop c) calculates the criticalNitrogenConcentration, updates the state of this class variable, and returns the calculated value.

6.7.2 M3_library.CropNitrogen.CalculateCropNitrogenConcentration

The function of CropNitrogen.CalculateCropNitrogenConcentration(Crop c) is to calculate the crop nitrogen concentration. It sets this concentration equal to 0, if the crop is not present in the field (i.e. if it is either harvested or has not emerged yet).

$$\nu_{\mathrm{i}}(t) = \begin{cases} 0 & | \quad B_{\mathrm{i}}(t) = 0 \\ \dfrac{N_{\mathrm{c,i}}(t)}{B_{\mathrm{i}}(t)} & | \quad B_{\mathrm{i}}(t) > 0 \end{cases}$$

Symbol	Description	Unit
$B_{\mathrm{i}}(t)$	Aboveground dry weight	kg m^{-2}
$N_{\mathrm{c,i}}(t)$	Crop nitrogen amount	kg m^{-2}
$\nu_{\mathrm{i}}(t)$	Crop nitrogen concentration	kg kg $^{-1}$

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs.

Output

CropNitrogen.CalculateCropNitrogenConcentration(Crop c) calculates the value of cropNitrogenConcentration, updates the state of this class variable, and returns the calculated value of cropNitrogenConcentration.

6.7.3 M3_library.CropNitrogen.CalculateCropNitrogenDemand

The method CropNitrogen.CalculateCropNitrogenDemand(Crop c, StripIntercrop ic) calculates the nitrogen demand. The demand is assumed to be 0 if either 1) the crop has not yet emerged, 2) the crop nitrogen amount is equal or larger than the maximum nitrogen amount, or 3) the crop has been harvested:

$$N_{\mathrm{c,max,i}}(t) = \nu_{\mathrm{i,max}}(t) \cdot B_{\mathrm{i}}(t)$$

$$D_{\mathrm{i}}(t) = \begin{cases} 0 & | \quad \delta_{\mathrm{i}}(t) = 0 \\ 0 & | \quad 0 < \delta_{\mathrm{i}}(t) < 2 \wedge N_{\mathrm{c,i}}(t) > N_{\mathrm{c,max,i}}(t) \\ N_{\mathrm{c,max,i}}(t) - N_{\mathrm{c,i}}(t) & | \quad 0 < \delta_{\mathrm{i}}(t) < 2 \wedge N_{\mathrm{c,i}}(t) \leq N_{\mathrm{c,max,i}}(t) \\ 0 & | \quad \delta_{\mathrm{i}}(t) = 2 \end{cases}$$

Symbol	Description	Unit
$B_{\mathrm{i}}(t)$	Aboveground dry weight	kg m^{-2}
$D_{\mathrm{i}}(t)$	Crop nitrogen demand	kg m^{-2}
i	Crop species index	
$N_{\mathrm{c,i}}(t)$	Crop nitrogen amount	kg m^{-2}
$N_{\mathrm{c,i,max}}(t)$	Maximum crop nitrogen amount	kg kg $^{-1}$
$\delta_{\mathrm{i}}(t)$	Developmental stage	
$\nu_{\mathrm{max,i}}(t)$	Maximum crop nitrogen concentration	kg kg $^{-1}$

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs.

Output

CropNitrogen.CalculateCropNitrogenDemand(StripIntercrop ic, Crop c) calculates the value of cropNitrogenDemand, updates the state of this class variable, and returns the calculated value of cropNitrogenDemand.

6.7.4 M3_library.CropNitrogen.CalculateCropNitrogenDemandFromSoil

The method CropNitrogen.CalculateCropNitrogenDemandFromSoil() calculates the amount of the nitrogen demand that the crop cannot obtained by N_2 fixation and, consequently, must take up from the soil in order to maintain its maximum nitrogen concentration.

$$D_{\mathrm{soil,i}}(t) = \left(1 - f_{\mathrm{fix,i}}\right) \cdot D_{\mathrm{i}}(t)$$

Symbol	Description	Unit
$D_{\mathrm{i}}(t)$	Crop nitrogen demand	kg m^{-2}
$D_{\mathrm{soil,i}}(t)$	Amount of nitrogen that the crop need to take up from the soil in order to fulfill its nitrogen demand.	kg m^{-2}
$f_{\mathrm{fix,i}}$	Fraction of nitrogen demand that the crop can obtain by N_2 fixation.	
i	Species index	

Input

None

Output

CropNitrogen.CalculateCropNitrogenDemandFromSoil() calculates the amount of nitrogen that a plant has to take up from the soil in order to fulfill its nitrogen demand. This value is stored in the variable cropNitrogenDemandFromSoil. CropNitrogen.CalculateCropNitrogenDemandFromSoil() calculates this value and updates the state of the CropNitrogen object.

6.7.5 M3_library.CropNitrogen.CalculateCropNitrogenFixed

The method CropNitrogen.CalculateCropNitrogenFixed calculates the amount of nitrogen that is daily fixed by the crop.

$$\Phi_{\mathrm{i}}(t) = \left(1 - f_{\mathrm{fix,i}}\right) \cdot D_{\mathrm{i}}(t)$$

Symbol	Description	Unit
$D_{\mathrm{i}}(t)$	Crop nitrogen demand	kg m^{-2}
$f_{\mathrm{fix,i}}$	Fraction of nitrogen demand that the crop can obtain by N_2 fixation.	
i	Species index	
$\Phi_{\mathrm{i}}(t)$	Nitrogen fixation rate	kg m^{-2} d^{-1}

Input

None

Output

CropNitrogen.CalculateCropNitrogenFixed calculates the amount of nitrogen that is daily fixed, updates the value of cropNitrogenFixationRate, and returns the calculated value.

6.7.6 M3_library.CropNitrogen.CalculateCropNitrogenGrowth

The method CropNitrogen.CalculateCropNitrogenGrowth(Crop c) calculates the net growth rate of the amount of crop nitrogen.

$$\frac{\Delta N_{\mathrm{c,i}}(t)}{\Delta t} = U_{\mathrm{i}}(t) + \Phi_{\mathrm{i}}(t) - \mu_{\mathrm{N,i}}(t)$$

Symbol	Description	Unit
$U_{\mathrm{i}}(t)$	Nitrogen uptake rate from the soil	kg m^{-2} d^{-1}
$\frac{\Delta N_{\mathrm{c,i}}(t)}{\Delta t}$	Rate of change crop nitrogen	kg m^{-2} d^{-1}
$\mu_{\mathrm{N,i}}(t)$	Crop nitrogen loss rate due to leaf senescence	kg m^{-2} d^{-1}
$\Phi_{\mathrm{i}}(t)$	N_2 fixation rate	kg m^{-2} d^{-1}

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs.

Output

CropNitrogen.CalculateCropNitrogenGrowth(Crop c) calculates the value of cropNitrogenGrowth, updates the CropNitrogen object, and returns the calculated value of cropNitrogenGrowth.

6.7.7 M3_library.CropNitrogen.CalculateCropNitrogenLoss

The function CropNitrogen.CalculateCropNitrogenLoss(Crop c) calculates the amount of nitrogen that is lost due to senescence.

$$N_{\mathrm{min,i}}(t) = \nu_{\mathrm{min,i}}(t) \cdot B_{\mathrm{i}}(t)$$
$$\mu_{\mathrm{N,i}}(t) = \nu_{\mathrm{min,i}}(t) \cdot \mu_{\mathrm{i}}(t) \cdot B_{\mathrm{i}}(t)$$

Symbol	Description	Unit
$B_{\mathrm{i}}(t)$	Aboveground dry weight	kg m^{-2}
$N_{\mathrm{min,i}}(t)$	Minimum amount of crop nitrogen	kg m^{-2}
$\mu_{\mathrm{i}}(t)$	Mortality rate of biomass	d-1
$\mu_{\mathrm{N,i}}(t)$	Amount of nitrogen that is lost due to senescence	kg m^{-2}
$\nu_{\mathrm{min,i}}(t)$	Minimum nitrogen concentration	kg kg^{-1}

6.7.8 M3_library.CropNitrogen.CalculateGrowthReductionFactor

The method CalculateGrowthReductionFactor(Crop c) calculates the reduction factor of the production of aboveground biomass due to nitrogen deficiencies. This factor is assumed to be 1 if the crop has not emerged yet or has been already harvested.

$$N_{min,i}(t) = \nu_{\mathrm{min,i}}(t) \cdot B_{\mathrm{i}}(t)$$
$$N_{crit,i}(t) = \nu_{\mathrm{crit,i}}(t) \cdot B_{\mathrm{i}}(t)$$

$$\sigma_{\mathrm{N,i}}(t) = \begin{cases} \max\left(0, \min\left(1, \frac{N_{c,i}(t) - N_{min,i}(t)}{N_{crit,i}(t) - N_{min,i}(t)}\right)\right) & | \quad 0 < \delta_{\mathrm{i}} < 2 \\ 1 & | \quad \delta_{\mathrm{i}} = 0 \vee \delta_{\mathrm{i}} = 2 \end{cases}$$

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs.

Output

CropNitrogen.CalculateGrowthReductionFactor(Crop c) calculates nitrogenGrowthReductionFactor, updates the CropNitrogen object and returns the value of nitrogenGrowthReductionFactor.

6.7.9 M3_library.CropNitrogen.CalculateMaximumNitrogenConcentration

The function CropNitrogen.CalculateMaximumCropNitrogenConcentration(StripIntercrop ic, Crop c) calculates the maximum crop nitrogen concentration as:

$$\ell_{\mathrm{tot}} = \sum_{\mathrm{j=1}}^{n_{\mathrm{crop}}} \ell_{\mathrm{j}}$$

$$\nu_{\mathrm{max,i}}(t) = \nu_{\mathrm{max,0,i}} \cdot \min\left(1, \left(\frac{B_{\mathrm{i}}(t)}{\alpha_{\mathrm{max,i}} \cdot \frac{\ell_{\mathrm{i}}}{\ell_{\mathrm{tot}}}}\right)^{-\beta_{\mathrm{max,i}}}\right)$$

Symbol	Description	Unit
$B_{\mathrm{i}}(t)$	Aboveground dry weight	kg m^{-2}
i	Species index	
j	Species index	
ℓ_{i}	Strip width	m
ℓ_{tot}	Sum of strip width of all crops in the intercrop	m
n_{crop}	Number of crop species in the intercrop	
$\alpha_{\mathrm{max,i}}$	Aboveground dry weight at which the maximum crop nitrogen concentration would be 1, in case the maximum crop nitrogen concentration would not be assumed to be constant at low biomass.	kg m^{-2}
$\beta_{\mathrm{max,i}}$	Shape parameter maximum nitrogen dilution curve	
$\nu_{\mathrm{max,i}}(t)$	Maximum nitrogen concentration	kg kg^{-1}
$\nu_{\mathrm{max,0}}$	Maximum nitrogen concentration at low biomass	

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs
ic	StripIntercrop	The StripIntercropObject of which the Crop object is a part

Output

CropNitrogen.CalculateMaximumNitrogenConcentration(StripIntercrop ic, Crop c) calculates the value of the class variable maximumNitrogenConcentration, updates the state of this class variable, and returns the calculated value.

6.7.10 M3_library.CropNitrogen.CalculateMinimumNitrogenConcentration

The function CropNitrogen.CalculateMinimumCropNitrogenConcentration(StripIntercrop ic, Crop c) calculates the minimum crop nitrogen concentration as:

$$\ell_{\mathrm{tot}} = \sum_{\mathrm{j}=1}^{n_{\mathrm{crop}}} \ell_{\mathrm{j}}$$

$$\nu_{\mathrm{min,i}}(t) = f_{\mathrm{min,i}} \cdot \nu_{\mathrm{max,0,i}} \cdot \min\left(1, \left(\frac{B_{\mathrm{i}}(t)}{\alpha_{\mathrm{min,i}} \cdot \frac{\ell_{\mathrm{i}}}{\ell_{\mathrm{tot}}}}\right)^{-\beta_{\mathrm{min,i}}}\right)$$

Symbol	Description	Unit
$B_{\mathrm{i}}(t)$	Aboveground dry weight	kg m^{-2}
$f_{\mathrm{min,i}}$	Fraction of minimum to maximum nitrogen concentration at low biomass	
i	Species index	
j	Species index	
ℓ_{i}	Strip width	m
ℓ_{tot}	Sum of strip width of all crops in the intercrop	m
n_{crop}	Number of crop species in the intercrop	
$\alpha_{\mathrm{min,i}}$	Aboveground dry weight at which the minimum crop nitrogen concentration would be 1, in case the crop minimum nitrogen concentration would not be assumed constant at low biomass	kg m^{-2}
$\beta_{\mathrm{min,i}}$	Shape parameter of the minimum nitrogen dilution curve	
$\nu_{\mathrm{min,i}}(t)$	Minimum nitrogen concentration	kg kg^{-1}
$\nu_{\mathrm{max,0,i}}$	Maximum nitrogen concentration at low biomass	kg kg^{-1}

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs.
ic	StripIntercrop	The StripIntercropObject from which the Crop object is a part.

Output

CropNitrogen.CalculateMinimumNitrogenConcentration(StripIntercrop ic, Crop c) calculates the value of the class variable minimumNitrogenConcentration, updates the state of this class variable, and returns the calculated value.

6.7.11 M3_library.CropNitrogen.GrowCropNitrogen

The method CropNitrogen.GrowCropNitrogen() calculates the state of the cropNitrogenObject at the next time step. The time step is assumed to be 1 d.

$$N_{\mathrm{c,i}}(t+\Delta t) = N_{\mathrm{c,i}}(t) + \Delta t \cdot \frac{\Delta N_{\mathrm{c,i}}(t)}{\Delta t}$$

Symbol	Description	Unit
$N_{\mathrm{c,i}}(t)$	Amount of crop nitrogen	kg m^{-2}
Δt	Time step	d

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs.

Output

CropNitrogen.GrowCropNitrogen() calculates cropNitrogenAmount for the next time step and updates the CropNitrogen object. It returns this updated object.

6.7.12 M3_library.CropNitrogen.InitializeCropNitrogen

CropNitrogen.InitializeCropNitrogen() initializes the CropNitrogen object at the start of the simulation

$$D_{\mathrm{i}}(t) = 0$$
$$N_{\mathrm{c,i}}(t) = 0$$
$$\nu_{\mathrm{i}}(t) = 0$$
$$\nu_{\mathrm{crit,i}}(t) = 0$$
$$\nu_{\mathrm{max,i}}(t) = 0$$
$$\nu_{\mathrm{min,i}}(t) = 0$$

Symbol	Description	Unit
$D_{\mathrm{i}}(t)$	Crop nitrogen demand	kg m^{-2}
$N_{\mathrm{c,i}}(t)$	Crop nitrogen amount	kg m^{-2}
$\nu_{\mathrm{i}}(t)$	Crop nitrogen concentration	kg kg^{-1}
$\nu_{\mathrm{crit,i}}(t)$	Critical crop nitrogen concentration	kg kg^{-1}
$\nu_{\mathrm{max,i}}(t)$	Maximum crop nitrogen concentration	kg kg^{-1}
$\nu_{\mathrm{min,i}}(t)$	Minimum crop nitrogen concentration	kg kg^{-1}

6.8 M3_library.DailyWeather

The file DailyWeather.cs contains the class DailyWeather. The purpose of this class is to store variables that describe meteorological variables of a single day. This class does not contain methods. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Variable name	Type	Meaning	Unit
dayOfMonth	int	Day of month	
dayOfYear	int	Day of year	
date	DateTime	Date	
maximumTemperature	double	Maximum daily temperature	^{o}C
minimumTemperature	double	Minimum daily temperature	^{o}C
month	int	Month	
temperature	double	Average daily temperature	^{o}C
radiation	double	Global daily radiation	MJ m^{-2} d^{-1}
weatherStationID	int	Identifier of the weather station where the observations were conducted	
year	int	Year	

6.9 M3_library.Farmer

The file Farmer.cs contains the management parameters. Additionally, the class contains a method to harvest the crop. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Variable name	Type	Description	Unit
cropFileDirectories	string array	Array that holds the directories of the crop files of the species in the intercrop	
cropFileNames	string array	Array that holds the names of the crop files of the species in the intercrop	
cropParameterFilePaths	string array	Array that holds the paths of the crop files of the species in the intercrop	
fertilizationAmounts	double array	Array that holds amount of applied nitrogen per fertilization event	kg m^{-2} d^{-1}
fertilizationDates	DateTime array	Array that holds the dates of all fertilization events	
harvestDates	DateTime array	Array that holds the harvest date per crop species in the intercrop	
irrigationAmounts	double array	Array that holds the amount of applied water per irrigation event'	m
irrigationDates	DateTime array	Array that holds the dates of all irrigation events	
stripWidths	double array	Array that holds the strip widths per crop species in the intercrop	m^{-2}
sowingDates	DateTime array	Array that holds the sowing dates per crop species in the intercrop	
sowingDensities	double array	Array that holds the sowing dates per crop species in the intercrop	m^{-2}

6.9.1 M3_library.Farmer.Harvest

The class Farmer.Harvest(Crop c) class the method Crop.Harvest().

Input

Variable name	Type	Description
c	Crop	Crop object to which the cropNitrogen object belongs

Output

The Crop object of the crop that was harvested.

6.10 M3_library.FarmerFileReader

The file FarmerReader.cs contains the class FarmerReader The purpose of the class FarmerReader is to read management parameters from farmer files and use them to initialize a Farmer object. It does not contain class variables. This class has no class variables.

6.10.1 M3_library.FarmFileReader.ReadFarmerFile

The method FarmerReader.ReadFarmerFile reads a farmer file, extracts the crop files, management parameters and the soil file from this file, uses this information to initialize a farmer file, and returns the farmer object. The input string filePath is the file path of the farmer file to be opened. It first declares and initializes the lists necessary to temporary store the file path of the crop file parameters, amount of fertilizers, fertilization dates, harvest dates, sowing densities sowing dates, and the strip widths of all crops in the cropping system. Next, it declares and initializes a StreamReader object sr, which reads the farmer file with the path stored in the method's input variable filePath. The StreamReader sr reads through the file and checks whether the first column of this line contains the substring "date_sow". If it does, it indicates that the line contains information about the sowing date, sowing densities, harvest date, strip widths, and the directory of the crop file parameters, and the file name of the crop file parameters. Those variables are then found in the third column (index 2), seventh column (index 6), tenth column (index 10), thirteenth column (index 14), and fifteenth column (index 16), respectively. Those variables are read and added to the previously defined lists. If the line contains the word "date_n_fert", it indicates that the lines contains information about the fertilization dates and amounts in the third (index 2) and seventh column (index 6), respectively. Those variables are red and stored in the corresponding lists. After all variables are read, the lists are converted into arrays. These arrays are used to declare and initialize a Farmer object. Finally, FarmerReader.ReadFarmerFile(string filePath) returns this Farmer object.

Input

Variable name	Type	Description
filePath	string	Location of the farmer file that is read

6.11 M3_library.Leaf.cs

The file Leaf.cs contains the class Leaf. The class Leaf contains initial conditions, state variables, intermediate variables, rates of change, indices and parameters at the leaf level. It contains methods to calculate the production and mortality of leaf tissue. The table below shows the class variables, their types, their meaning, and, when applicable, their units.

Variable name	Type	Description	Unit
compressedLeafAreaIndex	double	Compressed leaf area index	$m^2\ m^{-2}$
extinctionCoefficient	double	Extinction coefficient	$m^2\ m^{-2}$
leafAreaIndex	double	Leaf area index	$m^2\ m^{-2}$
leafAreaIndexProduction	double	Leaf area index production rate	$m^2\ m^{-2}\ d^{-1}$
leafAreaIndexMortality	double	Leaf area index mortality rate	$m^2\ m^{-2}\ d^{-1}$
leafProduction	double	Leaf dry matter production rate	$kg\ m^{-2}\ d^{-1}$
leafMortality	double	Leaf dry matter mortality rate	$kg\ m^{-2}\ d^{-1}$
minimumTemperatureSenescence	double	Temperature above which leaf senescence can take place	^oC
initialLeafArea	double	Initial leaf area per plant	m^2
leafWeight	double	Leaf dry matter	$kg\ m^{-2}$
radiationUseEfficiency	double	Radiation use efficiency (based on photosynthetically active radiation)	$kg\ J^{-1}$
specificLeafArea	double	Specific leaf area	$m^2\ kg^{-1}$
slopeSenescenceTemperature	double	Sensitivity of leaf senescence to temperature	$^oC\ d$

6.11.1 M3_library.Leaf.CalculateCumulativeLeafAreaIndex

The function Leaf.CalculateCumulativeLeafAreaIndex(Crop c, h) calculates the cumulative leaf are index between the crop height and a certain height, measured from the ground level.

$$L_{c,i}(t,h) = \begin{cases} 0 & | \quad h > H_i(t) \\ L_i(t) \cdot \dfrac{H_i(t) - h}{H_i(t)} & | \quad h \leq H_i(t) \end{cases}$$

Symbol	Description	Unit
h	Height from which the cumulative leaf area index is calculated.	m
$H_i(t)$	Crop height	m
i	Species index	
$L_i(t)$	Leaf area index	$m^2\ m^{-2}$
$L_{c,i}(t)$	Cumulative leaf area index	$m^2\ m^{-2}$

Input

Variable name	Type	Description
c	Crop	Crop object for which the cumulative leaf area index is calculated
h	double	Height for which the cumulative leaf area index is calculated

Output

Leaf.CalculateCumulativeLeafAreaIndex calculates the cumulative leaf are index and returns the calculated value.

6.11.2 M3_library.Leaf.CalculateLeafProduction

The method Leaf.CalculateLeafProduction(Crop c, DailyWeather dw) calculates the leaf dry matter production rate and the leaf area index production rate as:

$$I_{\mathrm{int,i}}(t) = f_{\mathrm{PAR}} \cdot f_{\mathrm{int,i}}(t) \cdot I_0(t)$$
$$P_{\mathrm{BL,i}} = \sigma_{\mathrm{N,i}}(t) \cdot \eta_{\mathrm{i}} \cdot I_{\mathrm{int,i}}(t) \cdot f_{\mathrm{l,i}}(t)$$
$$P_{\mathrm{L,i}} = P_{\mathrm{BL,i}} \cdot S_{\mathrm{la,i}}$$

Symbol	Description	Unit
f_{PAR}	Fraction of photosynthetically active radiation in global radiation	
$f_{\mathrm{int,i}}(t)$	Fraction of radiation intercepted by species i	
$f_{\mathrm{l,i}}(t)$	Fraction of newly produced aboveground dry matter partitioned to the leaves	
i	Species index	
$I_0(t)$	Global radiation	J m^{-2} d^{-1}
$L_{\mathrm{i}}(t)$	Leaf area index	m^2 m^{-2}
$P_{\mathrm{BL,i}}$	Leaf dry matter production rate	kg m^{-2} d^{-1}
$P_{\mathrm{L,i}}$	Leaf area index production rate	m^2 m^{-2} d^{-1}
$S_{\mathrm{la,i}}$	Specific leaf area	m^2 kg^{-1}
η_{i}	Radiation use efficiency (based on photosynthetically active radiation).	J kg^{-1}
$\sigma_{\mathrm{N,i}}(t)$	Crop growth reduction factor due to nitrogen deficiencies	

Input

Variable name	Type	Description
c	Crop	Crop object for which the leaf dry matter and leaf area index production are calculated
dw	DailyWeather	DailyWeather object

Output

Leaf.CalculateLeafProduction calculates the leaf dry matter production rate and the leaf area index, updates the state of the Leaf object, and returns this object.

6.11.3 M3_library.Leaf.CalculateLeafMortality

The method Leaf.CalcucluateLearMortality(Crop c, DailyWeather dw) calculates the mortality rate of the leaf dry matter and the leaf area index as:

$$\mu_{\mathrm{BL,i}}(t) = \min\left(1, max\left(s_{\mathrm{sen,i}} \cdot (T(t) - T_{\mathrm{sen}})\right)\right) \quad | \quad \delta_{\mathrm{i}}(t) \geq 1$$
$$M_{\mathrm{BL,i}}(t) = \mu_{\mathrm{BL,i}}(t) \cdot B_{\mathrm{BL,i}}(t)$$
$$M_{\mathrm{L,i}}(t) = M_{\mathrm{BL,i}}(t) \cdot S_{\mathrm{la,i}}$$

Symbol	Description	Unit
$B_{\mathrm{BL,i}}(t)$	Leaf dry matter	kg m^{-2}
$M_{\mathrm{BL,i}}(t)$	Leaf dry matter mortality	kg m^{-2} d^{-1}
$M_{\mathrm{L,i}}(t)$	Leaf area index mortality	kg m^{-2} d^{-1}
$s_{\mathrm{sen,i}}$	Sensitivity of leaf mortality to temperature	$^oC^{-1}$ d^{-1}
$S_{\mathrm{la,i}}$	Specific leaf area	m^2 kg^{-1}
$T(t)$	Daily average temperature	^{o}C
T_{sen}	Temperature above which leaf mortality takes place	^{o}C
$\delta_{\mathrm{i}}(t)$	Developmental stage	
$\mu_{\mathrm{BL,i}}(t)$	Relative leaf dry matter mortality rate	d^{-1}

Input

Variable name	Type	Description
c	Crop	Crop object for which the leaf dry matter and leaf area index production are calculated.
dw	DailyWeather	DailyWeather object

Output

The method Leaf.CalculateLeafMortality(Crop c, DailyWeather dw) calculates the mortality rate of the leaf dry matter and the leaf area index, updates the Leaf object and returns this object.

6.12 M3_library.Phenology.cs

The file Phenology.cs contains the class Phenology. The class Phenology contains initial conditions, state variables, intermediate variables, rates of change, indices and parameters of the crop phenology. The Phenology class also contains properties for each class variable. It also contains function for calculations related to the crop's phenology. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Variable name	Type	Description	Unit
baseTemperature	double	Base temperature: temperature below which temperature sums does not increase with the temperature	^{o}C d
sowingDate	DateTime	Sowing date	
developmentState	double	Developmental stage	
temperatureSumFloweringToMaturity	double	Temperature sum from flowering to maturity	^{o}C d
temperatureSumFromFlowering	double	Temperature sum from flowering	^{o}C d
temperatureSumEmergenceToFlowering	double	Temperature sum from emergence to flowering	^{o}C d
temperatureSumFromEmergence	double	Temperature sum from emergence	^{o}C d
temperatureSumFromSowing	double	Temperature sum from sowing	^{o}C d
temperatureSumSowingToEmergence	double	Temperature sum from sowing to emergence	^{o}C d

6.12.1 M3_library.Phenology.CalculateDevelopmentState

The method Phenology.CalculateDevelopmentState(Timer t, DailyWeather dw) calculates the development stage of the crop.

$$\delta_{\mathrm{i}}(t) = \begin{cases} 0 & | \quad S_{\mathrm{i}}(t) \leq S_{\mathrm{e,i}} \\ \dfrac{S_{\mathrm{i}}(t) - S_{\mathrm{e,i}}}{S_{\mathrm{a,i}}} & | \quad S_{\mathrm{e,i}} < S_{\mathrm{i}}(t) \leq S_{\mathrm{e,i}} + S_{\mathrm{a,i}} \\ 1 + \dfrac{S_{\mathrm{i}}(t) - S_{\mathrm{a,i}} - S_{\mathrm{e,i}}}{S_{\mathrm{m,i}}} & | \quad S_{\mathrm{e,i}} + S_{\mathrm{a,i}} < S_{\mathrm{i}}(t) \leq S_{\mathrm{e,i}} + S_{\mathrm{a,i}} + S_{\mathrm{m,i}} \\ 2 & | \quad S_{\mathrm{i}}(t) \geq S_{\mathrm{e,i}} + S_{\mathrm{m,i}} + S_{\mathrm{a,i}} \end{cases}$$

Symbol	Description	Unit
$S_{\mathrm{a,i}}$	Temperature sum from emergence until anthesis	°C d
$S_{\mathrm{e,i}}$	Temperature sum from sowing to emergence	°C d
$S_{\mathrm{i}}(t)$	Temperature sum from sowing	°C d
$S_{\mathrm{m,i}}$	Temperature sum from anthesis to maturity	°C d
$\delta_{\mathrm{i}}(t)$	Developmental stage	-

Input

Variable name	Type	Description
t	Timer	Timer object
w	WeatherStation	WeatherStation object

Output

The method Phenology.CalculateDevelopmentState(Timer t, DailyWeather dw) calculates the temperature sum from sowing, updates the Phenology object, and returns this object.

6.12.2 M3_library.Phenology.CalculateTemperatureSumFromSowing

The method Phenology.CalculateTemperatureSumFromSowing(Timer t, WeatherStation w) calculates the temperature from sowing to a certain time:

$$\bar{T}(t) = \frac{T_{\min}(t) + T_{\max}(t)}{2}$$
$$T_{\mathrm{e}}(t) = \min(0, \bar{T}(t) - T_{\mathrm{b}})$$
$$S_{\mathrm{i}}(t) = \sum_{\tau=t_{\mathrm{s}}}^{t} T_{\mathrm{e}}(\tau)$$

Symbol	Description	Unit
$S_{\mathrm{i}}(t)$	Temperature sum from sowing	°C d
t_{s}	Sowing date	d
$\bar{T}(t)$	Average temperature	°C
T_{b}	Base temperature	°C
$T_{\mathrm{e}}(t)$	Effective temperature	°C

Input

Variable name	Type	Description
t	Timer	Timer object
w	WeatherStation	WeatherStation object

Output

The method Phenology.CalculateTemperatureSumFromSowing(Timer t, WeatherStation w) calculates the temperature sum from sowing and returns the Phenology object.

6.12.3 M3_library.Phenology.CalculateTemperatureSumFromEmergence

The method Phenology.CalculateTemperatureSumFromEmergence (Timer t, WeatherStation w) calculates the temperature from emerge to a certain time:

$$S_{t_e \to t}(t) = \max(0, S_i(t) - S_e)$$

Symbol	Description	Unit
S_e	Temperature sum from emergence	°C d
$S_i(t)$	Temperature sum from sowing	°C d
$S_{t_e \to t}(t)$	Temperature sum from emergence to a certain time *t*	°C d
t_e	Emergence date	d

Input

Variable name	Type	Description
t	Timer	Timer object
w	WeatherStation	WeatherStation object

Output

The method Phenology.CalculateTemperatureSumFromEmergence(Timer t, WeatherStation w) calculates the temperature sum from emergence and returns the Phenology object.

6.12.4 M3_library.Phenology.CalculateTemperatureSumFromAnthesis

The method Phenology.CalculateTemperatureSumFromAnthesis (Timer t, WeatherStation w) calculates the temperature from emerge to a certain time:

$$S_{t_a \to t}(t) = S_{t_a \to t}(t) - S_{t_e \to t}(t)$$

Symbol	Description	Unit
S_e	Temperature sum from emergence	°C d
$S_i(t)$	Temperature sum from sowing	°C d
$S_{t_e \to t}(t)$	Temperature sum from emergence to a certain time *t*	°C d
$S_{t_a \to t}(t)$	Temperature sum from anthesis to a certain time *t*	°C d

Input

Variable name	Type	Description
t	Timer	Timer object
w	WeatherStation	WeatherStation object

Output

The method Phenology.CalculateTemperatureSumFromAnthesis(Timer t, WeatherStation w) calculates the temperature sum from anthesis and returns the Phenology object.

6.13 M3_library.PhysicalConstants

The file PhysicalConstants.cs contains the class PhysicalConstants. The purpose of this file is to hold physical constants. These constants are hard-coded. This class does not contain methods. The tables below show the class variable, its types, its meaning, and its unit.

Variable name	Type
fractionOfPARInRadiation	const double

Variable name	Description
fractionOfPARInRadiation	Fraction of photosynthetically active radiation in the global radiation

Variable name	Unit
fractionOfPARInRadiation	$J\ J^{-1}$

6.14 M3_library.Soil

The file Soil.cs contains the class Soil. Its purpose is to hold a SoilMineralNitrogen object. The table below shows its class variable, its type, and its meaning.

Variable name	Type	Description
soilMineralNitrogen	SoilMineralNitrogen	Soil mineral nitrogen object

6.15 M3_library.SoilMineralNitrogen

The file SoilMineralNitrogen.cs contains the class SoilMineralNitrogen. The class SoilMineralNitrogen contains initial conditions, state variables, intermediate variables, rates of change, indices and parameters related to nitrogen in the soil. It also contains functions to calculate the soil mineral nitrogen amount and the daily production of soil mineral nitrogen by mineralization and fertilization and the loss of soil mineral nitrogen by uptake. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Variable name	Type	Description	Unit
initialNitrogenAmount	double	Amount of soil mineral nitrogen at the start of the simulation	kg m^{-2}
fertilizerAmount	double	Amount of fertilizer that was applied	kg m^{-2} d^{-1}
fractionOfFertilizerRecovered	double	Fraction of applied fertilizer that is recovered.	
netMineralizationRate	double	Net production of soil mineral nitrogen by mineralization	kg m^{-2} d^{-1}
soilNitrogenNetGrowth	double	Net growth of soil mineral nitrogen	kg m^{-2} d^{-1}
soilMineralNitrogenAmount	double	Amount of soil mineral nitrogen	kg m^{-2}
soilMineralNitrogenProduction	double	Production of soil mineral nitrogen	kg m^{-2} d^{-1}
soilMineralNitrogenUptake	double	Removal of soil mineral nitrogen by crop uptake	kg m^{-2} d^{-1}

6.15.1 M3_library.SoilMineralNitrogen.GrowNitrogen

The method SoilMineralNitrogen.GrowNitrogen calculates the value of the soil mineral nitrogen amount in the next time step as:

$$N_s(t + \Delta t) = N_s(t) + \Delta t \cdot \frac{\Delta N_s(t)}{\Delta t}$$

Symbol	Description	Unit
$N_s(t)$	Amount of soil mineral nitrogen	kg m^{-2}
Δt	Time step	d

Input

None

Output

The method SoilMineralNitrogen.GrowNitrogen calculates the new value of the variable soilMineralNitrogen in the next time step, updates the SoilMineralNitrogen object and returns the new value of soilMineralNitrogen.

6.16 M3_library.Stem

The file Stem.cs contains the class Stem. The class Stem contains initial variables, intermediate variable, rates of change, parameters and state variables related to the height of the crop. It calculates the change in crop height and records the stem weight. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Variable name	Type	Description	Unit
shootHeightGrowth	double	Shoot height growth rate	m d^{-1}
shootHeight	double	Shoot height	m
initialShootHeight	double	Shoot height at emergence	m
maximumShootHeight	double	Maximum shoot height	m

relativeGrowthRateShootHeight	double	Relative temperature sensitive height growth rate	(°C d)$^{-1}$

6.16.1 M3_library.Stem.CalculateShootHeightGrowth

The method Stem.CalculateShootHeightGrowth(Crop c, DailyWeather dw) calculates the height growth rate as:

$$T_{\mathrm{e}}(t) = \max(0, \bar{T}(t) - T_{\mathrm{b}})$$
$$P_{\mathrm{H,i}}(t) = H_{\mathrm{i}}(t) \cdot r_{\mathrm{h,i}} \cdot \left(1 - \frac{H_{\mathrm{i}}(t)}{H_{\mathrm{m,i}}}\right) \cdot T_{\mathrm{e}}(t)$$

Symbol	Description	Unit
$H_{\mathrm{i}}(t)$	Crop height	m
$H_{\mathrm{m,i}}$	Maximum crop height	m
$P_{\mathrm{H,i}}(t)$	Height growth rate	m d^{-1}
$r_{\mathrm{h,i}}$	Temperature dependent relative height growth rate	°C^{-1} d^{-1}
$\bar{T}(t)$	Average temperature	°C
$T_{\mathrm{e}}(t)$	Effective temperature	°C

Input

Variable name	Type	Description
c	Crop	Crop object for which the height growth is calculated.
dw	DailyWeather	DailyWeather object

Output

Stem.CalculateShootHeightGrowth(Crop c, DailyWeather dw) calculates the rate of crop height growth, updates the Stem object and returns this stem object.

6.17 M3_library.StorageOrgan

The file StorageOrgan.cs contains the class StorageOrgan. The class StorageOrgan contains initial variables, intermediate variable, rates of change, parameters and state variables related to the storage organ. It calculates the growth rate of the weight of the storage organs of the crop per time step. The tables below show the class variables, their types, their meaning, and, when applicable, their units.

Name	Type
storageOrganWeight	double
storageOrganProduction	double

Name	Description
storageOrganWeight	Storage organ dry matter weight

storageOrganProduction	Storage organ dry matter production rate

Name	Unit
storageOrganWeight	kg m^{-2}
storageOrganProduction	kg m^{-2} d^{-1}

6.17 M3_library.StorageOrgan.CalculateStorageOrganProduction

The method StorageOrgan.CalculateStorageOrganProduction(Crop c, DailyWeather dw) calculates the rate of storage organ dry matter production:

$$I_{\mathrm{int,i}}(t) = f_{\mathrm{int,i}}(t) \cdot f_{\mathrm{PAR}} \cdot I_0(t)$$

$$P_{\mathrm{Y,i}}(t) = \begin{cases} \sigma_{\mathrm{N,i}}(t) \cdot \eta_{\mathrm{i}}(t) \cdot I_{\mathrm{int,i}}(t) \cdot f_{\mathrm{y,i}}(t) & | \quad \delta_{\mathrm{i}}(t) < 2 \\ 0 & | \quad \delta_{\mathrm{i}}(t) = 2 \end{cases}$$

Symbol	Description	Unit
$f_{\mathrm{i}}(t)$	Fraction of global radiation that is intercepted by species i	
$f_{\mathrm{y,i}}(t)$	Fraction of newly produced aboveground dry matter partitioned to the storage organs	
f_{PAR}	Fraction of photosynthetic radiation in global radiation	
i	Crop species index	
$I_0(t)$	Global radiation	J m^{-2} d^{-1}
$I_{\mathrm{int,i}}(t)$	Amount of intercepted light by species *i* at time *t*	J m^{-2} d^{-1}
$P_{\mathrm{Y,i}}(t)$	Storage organ dry matter production rate	kg m^{-2} d^{-1}
$\delta_{\mathrm{i}}(t)$	Developmental stage	
$\eta_{\mathrm{i}}(t)$	Radiation use efficiency based on photosynthetically active radiation.	J kg^{-1}

Input

Variable name	Type	Description
c	Crop	Crop object for which the storage organ growth weight is calculated.
dw	DailyWeather	DailyWeather object

Output

StorageOrgan.CalculateStorageOrganProduction(Crop c, DailyWeather dw) calculates the rate of storage organ dry matter production, updates the StorageOrgan object, and returns the updated StorageOrgan object.

6.18 M3_library.StripIntercrop

The file StripIntercrop.cs contains the class StripIntercrop. One purpose of the class StripIntercrop is that it holds an array with Crop objects. This array contains the state variables, rates of change, parameters, and initial conditions for all crop within the intercrop. Only intercrops of two or one (which is actually a pure culture) can be simulated. The class StripIntercrop also contains methods to calculate the fraction of solar

radiation that are daily intercepted by the crops in the intercrop. The table below shows an overview of the class variables, their types, and their units.

Variable name	Type	Description	Unit
crop1IsShorterCrop	bool	Indicates whether the first Crop object in the object Crops is shorter than the second one	
crop1IsTallerCrop	bool	Indicates whether the first Crop object in the object Crops is taller than the second one	
crops	Crop array	Crop array that holds the Crop objects that describe the state of the crop in the intercrop	
weightFunction	double	Weight factor that indicates to which extend the strip intercrop resembles a compressed intercrop	

6.18.1 M3_library.StripIntercrop.CalculateCumlativeCompressedLeafAreaIndexUpper CanopyTallerCrop

The method StripIntercrop.CalculateCumulativeCompressedLeafAreaIndexUpperCanopyTallerCrop(double H, double H_diff, double L_comp) calculates the leaf area index between the top of the canopy of the taller crop in the intercrop and a certain height:

$$L_{\mathrm{c,u,h}}(t) = \frac{\Delta H_{\mathrm{i}}(t)}{H_{\mathrm{i}}(t)} \cdot L_{\mathrm{c,u}}(t)$$

Symbol	Description	Unit
$H_{\mathrm{i}}(t)$	Crop height	m
$L_{\mathrm{c,u}}(t)$	Compressed leaf area index	$m^2\ m^{-2}$
$L_{\mathrm{c,u,h}}(t)$	Cumulative leaf area index between the crop height and height $H_{\mathrm{i}}(t) - \Delta H_{\mathrm{i}}(t)$.	$m^2\ m^{-2}$
$\Delta H_{\mathrm{i}}(t)$	Difference between the crop height $H_{\mathrm{i}}(t)$ and the height for which the cumulative leaf area index is calculated.	m

Input

Variable name	Type	Description
H	double	Crop height
H_diff	double	Difference between the crop height and the height for which the cumulative leaf area index is calculated.
L_comp	double	Compressed leaf area index

Output

StripIntercrop.CalculateCumulativeCompressedLeafAreaIndexUpperCanopyTallerCrop(double H, double H_diff, double L_comp) calculates the leaf area index between the top of the canopy of the taller crop in the intercrop and a certain height and returns this value.

6.18.2 M3_library.StripIntercrop.CalculateCompressedLeafAreaIndices

StripIntercrop.CalculateCompressedLeafAreaIndices() calculates for each Crop object in the StripIntercrop object what the compressed leaf area index is:

$$L_{c,i}(t) = L_i(t) \cdot \frac{\ell_i + \ell_j}{\ell_i}$$

Symbol	Description	Unit
i	Crop species index of the species for which the compressed leaf area index is calculated	
j	Crop species index of the species that is intercropped with the species for which the compressed leaf area index is calculated	
ℓ_i	Strip width of the crop species for which the compressed leaf area index is calculated	m
ℓ_j	Strip width of the species which is intercropped with the species for which the compressed leaf area index is calculated	m
$L_{c,i}(t)$	Compressed leaf area index	$m^2\ m^{-2}$
$L_i(t)$	Leaf area index	$m^2\ m^{-2}$

6.18.3 M3_library.StripIntercrop.CalculateFractionOfRadiationInterceptionPerCrop

StripIntercrop.CalculateFractionOfRadiationInterceptionPerCrop() calculates for each crop species in the intercrop the fraction of the global radiation that is intercepted by this crop. It does this following these steps:

1) Check which of the two species in the Crop array crops is the taller species.

2) Collect for both the shorter and the tallest species their height, extinction coefficient, leaf area index, and the strip widths.

3) Calculate the compressed leaf area indices of both species.

4) Calculate the cumulative leaf area index of the upper part of the canopy of the taller species.

5) Calculate the height difference between the height of the upper and the lower canopy:

6) Calculate the compressed cumulative leaf area index of the upper canopy and the lower canopy of the taller crop.

7) Calculate the view factors of the sky for both the shortest and tallest crop.

8) Calculate the fraction of the radiation that reaches i) the top of the canopy of the shorter crop in the intercrop and ii) the top of the lower canopy of the taller crop in the intercrop.

9) Calculate the weight factor.

10) Calculate the fractions of the global radiation that are intercepted by i) the upper canopy by a compressed canopy, ii) the upper canopy by a homogeneous canopy, iii) the upper canopy of the taller crop, iv) the lower canopy of the taller crop, v) the canopy of the shorter canopy.

11) The total fraction of the global radiation that is intercepted by the taller crop.

Input

None

Output

StripIntercrop.CalculateFractionOfRadiationInterceptionPerCrop() calculates for the shorter and the taller crop what the fraction of light interception is. The calculated fractions are assigned to the corresponding Crop objects. The function returns the updated StripIntercrop object.

6.18.4 M3_library.StripIntercrop.CalculateGrowthIntercrop

The method StripIntercrop.CalculateGrowthIntercrop(Soil s, Timer t, WeatherStation w, DailyWeather dw) calculates the growth rates in each crop object that the Crops array crop holds.

Input

Variable name	Type	Description
s	Soil	Soil object that describes the state of the soil at which the intercrop is grown
t	Timer	Timer object
w	WeatherStation	WeatherStation object that holds information about the weather station of the site where the intercrop is grown
dw	DailyWeathr	DailyWeather object that holds information about the weather of that day

Output

The method StripIntercrop.CalculateGrowthIntercrop(Soil s, Timer t, WeatherStation w, DailyWeather dw) calculates the growth rates in each crop object that the Crops array crop holds, updates the StripIntercrop object, and returns the StripIntercrop object.

6.18.5 M3_library.StripIntercrop.CalculateHeightDifference

The method StripIntercrop.CalculateHeightDifference() calculates the height difference between the taller crop and the shorter crop.

$$\Delta H(t) = \left| H_{\mathrm{i}}(t) - H_{\mathrm{j}}(t) \right|$$

Symbol	Description	Unit
$H_{\mathrm{i}}(t)$	Crop height of the first species in the intercrop	m
$H_{\mathrm{j}}(t)$	Crop height of the second species in the intercrop	m
i	Crop species index	
j	Crop species index	
$\Delta H(t)$	Difference between the crop heights of both species in the intercrop.	m

Input

None

Output

StripIntercrop.CalculateHeightDifference() returns the calculated height differences.

6.18.6 M3_library.StripIntercrop.CalculateInterceptionFractionLowerCanopyTallerCrop

The method StripIntercrop.CalculateInterceptionFractionLowerCanopyTallerCrop (double k_t, double L_l_t_c, double P, double SR_ni, double W) calculates the fraction of the global radiation that is intercepted by the lower canopy of the taller crop:

$$f_{\mathrm{t,l}}(t) = f_{\mathrm{trans,t}} \cdot \left(1 - \mathrm{e}^{-k_{\mathrm{t}}(t) \cdot L_{\mathrm{t,l,c}}(t)}\right) \cdot \frac{\ell_{\mathrm{t}}(t)}{\ell_{\mathrm{s}}(t) + \ell_{\mathrm{t}}(t)}$$

Symbol	Description	Unit
$f_{\mathrm{t,l}}(t)$	Fraction of global radiation that is intercepted by the lower canopy of the taller crop.	
$f_{\mathrm{trans,t}}$	Fraction of global radiation that is transmitted through the top canopy of the taller crop and enter the top of the lower canopy of the taller crop.	
$k_{\mathrm{t}}(t)$	Extinction coefficient of the taller crop	
$\ell_{\mathrm{s}}(t)$	Strip width of lower crop	m
$\ell_{\mathrm{t}}(t)$	Strip width of taller crop	m
$L_{\mathrm{t,l,c}}(t)$	Cumulative leaf area index of the lower canopy of the taller crop	$m^2\ m^{-2}$

Input

Variable name	Type	Description
k_t	double	Extinction coefficient of the taller species
L_l_t_c	double	Cumulative leaf area index of taller species
P	double	Cumulative leaf area index of shorter species
SR_ni	double	Fraction of global radiation that reaches the top of the lower canopy of the taller crop
W	double	Strip width of the taller crop

Output

The method StripIntercrop.CalculateInterceptionFractionLowerCanopyTallerCrop (double k_t, double L_l_t_c, double P, double SR_ni, double W) calculates the fraction of the global radiation that is intercepted by the lower canopy of the taller crop and returns its value.

6.18.7 M3_library.StripIntercrop.CalculateInterceptionFractionUpperCanopyTallerCrop

The method StripIntercrop.CalculateInterceptionFractionUpperCanopyTallerCrop(double fint_u_t_h, double fint_u_t_c, double weight) calculates the fraction of the global radiation that is intercepted by the upper canopy of the taller crop

$$f_{\mathrm{t,u}} = \big(1 - w(t)\big) \cdot f_{\mathrm{int,u,t,h}}(t) + w(t) \cdot f_{\mathrm{int,u,t,c}}(t)$$

Symbol	Description	Unit
$f_{\mathrm{int,u,t,c}}(t)$	Fraction of radiation that would have been intercepted by the top canopy of the taller crop in case of a compressed canopy	
$f_{\mathrm{int,u,t,h}}(t)$	Fraction of radiation that would have been intercepted by the top canopy of the taller crop in case of a homogeneous canopy	
w	Weight factor that describes to what extend the top canopy of the intercrop resembles a compressed canopy	

Input

Variable name	Type	Description
fint_u_t_c	double	Fraction of radiation that would have been intercepted by the top canopy of the taller crop in case of a compressed canopy
fint_u_t_h	double	Fraction of radiation that would have been intercepted by the top canopy of the taller crop in case of a homogeneous canopy
weight	double	Weight factor that describes to what extend the top canopy of the intercrop resembles a compressed canopy

Output

The method StripIntercrop.CalculateInterceptionFractionUpperCanopyTallerCrop(double fint_u_t_h, double fint_u_t_c, double weight) calculates the fraction of the global radiation that is intercepted by the upper canopy of the taller crop and returns it value.

6.18.8 M3_library.StripIntercrop.CalculateInterceptionFractionHomogeneousCanopy

The method StripIntercrop.CalculateInterceptionFractionHomogeneousCanopy(double k, double L) calculates the fraction of the global radiation that is intercepted by a homogeneous canopy:

$$f_{\mathrm{int,h,i}}(t) = 1 - \mathrm{e}^{-k \cdot L_{\mathrm{i}}(t)}$$

Symbol	Description	Unit
$f_{\mathrm{int,h,i}}(t)$	Fraction of light that is intercepted by a homogeneously mixed intercrop	
i	Crop species index	
k_{i}	Extinction coefficient	
$L_{\mathrm{i}}(t)$	Leaf area index	

Input

Variable name	Type	Description
k	double	Extinction coefficient
L	double	Leaf area index

Output

The method StripIntercrop.CalculateInterceptionFractionHomogeneousCanopy(double k, double L) calculates the fraction of the global radiation that is intercepted by a homogeneous canopy and returns the calculated value.

6.18.9 M3_library.StripIntercrop.CalculateInterceptionFractionByCompressedCanopy

StripIntercrop.CalculateInterceptionFractionByCompressedCanopy(double k, double L_c, double P, double SR_ni, double W) calculated the fraction of global radiation that is intercepted by a compressed canopy:

$$f_{\mathrm{int,c,i}}(t) = \frac{\ell_{\mathrm{i}}}{\ell_{\mathrm{i}} + \ell_{\mathrm{j}}} \cdot \left(1 - \mathrm{e}^{-k \cdot L_{\mathrm{c}}(t)}\right)$$

Symbol	Description	Unit
$f_{\mathrm{int,h}}(t)$	Fraction of light that is intercepted by a compressed canopy of species *i*	
k	Extinction coefficient	
ℓ_{i}	Strip width of the crop which is grown in an intercrop with the crop for which the fraction of light interception in case of a compressed canopy is calculated	m
ℓ_{j}	Strip width of the crop which is grown in an intercrop with the crop for which the fraction of light interception in case of a compressed canopy is calculated	m
$L_{\mathrm{c,i}}(t)$	Compressed area index	$m^2\ m^{-2}$

Input

Variable name	Type	Description
k	double	Extinction coefficient
L_c	double	Compressed leaf area index
P	double	Strip width of the crop which is grown in an intercrop with the crop for which the fraction of light interception in case of a compressed canopy is calculated
W	double	Strip width of the crop which is grown in an intercrop with the crop for which the fraction of light interception in case of a compressed canopy is calculated

6.18.10 M3_library.StripIntercrop.CalculateInterceptionFractionShorterCrop

StripIntercrop.CalculateInterceptionFractionShorterCrop(double k_s, double L_s_c, double P, double SP_ni, double W) calculates the fraction of the global radiation that is intercepted by the shorter crop:

$$f_{\mathrm{s}}(t) = f_{\mathrm{trans,s}} \cdot \left(1 - \mathrm{e}^{-k_{\mathrm{s}}(t) \cdot L_{\mathrm{s,c}}(t)}\right) \cdot \frac{\ell_{\mathrm{s}}(t)}{\ell_{\mathrm{s}}(t) + \ell_{\mathrm{t}}(t)}$$

Symbol	Description	Unit
$f_{\mathrm{int,s}}(t)$	Fraction of global radiation that is intercepted by the lower canopy of the taller crop.	
$f_{\mathrm{trans,s}}$	Fraction of global radiation that is transmitted to the top of the canopy of the shorter crop	
$k_{\mathrm{s}}(t)$	Extinction coefficient of the shorter crop	
$\ell_{\mathrm{s}}(t)$	Strip width of shorter crop	m
$\ell_{\mathrm{t}}(t)$	Strip width of taller crop	m
$L_{\mathrm{t,l,c}}(t)$	Cumulative leaf area index of the lower canopy of the taller crop	$m^2\ m^{-2}$

Input

Variable name	Type	Description
k_s	double	Extinction coefficient
L_s_c	double	Compressed leaf area index of the shorter crop
P	double	Strip width of the taller crop
SP_ni	double	Fraction of the global radiation that is intercepted by the shorter crop.
W	double	Strip width of the shorter crop

Output

StripIntercrop.CalculateInterceptionFractionShorterCrop(double k_s, double L_s_c, double P, double SP_ni, double W) calculates the fraction of the global radiation that is intercepted by the shorter crop: and returns the value.

6.18.11 M3_library.StripIntercrop.CalculateFractionOfRadiationAtTopLowerCanopyTallerCrop

The method StripIntercrop.CalculateFractionOfRadiationAtTopLowerCanopyTallerCrop(double k, double L, double L_c, double IR_b) calculates the radiation of the global radiation that enter the top of the lower canopy of the taller crop:

$$f_{\mathrm{trans,u,t}}(t) = V_{\mathrm{t}}(t) \cdot \mathrm{e}^{-k_{\mathrm{t}}(t) \cdot L_{\mathrm{c,u,t}}(t)} + \left(1 - V_{\mathrm{t}}(t)\right) \cdot \mathrm{e}^{-k_{\mathrm{t}}(t) \cdot L_{\mathrm{u,t}}(t)}$$

Symbol	Description	Unit
$f_{\mathrm{trans,u,t}}(t)$	Fraction of global radiation that enters the top of the lower canopy of the taller crop	
$k_{\mathrm{t}}(t)$	Extinction coefficient of taller crop	
$L_{\mathrm{u,t}}(t)$	Leaf area index of the upper canopy of the taller crop	$m^2\ m^{-2}$
$L_{\mathrm{c,u,t}}(t)$	Compressed leaf area index of the upper canopy of the taller crop	$m^2\ m^{-2}$
$V_{\mathrm{t}}(t)$	View factor of the sky of the strips of the taller crop	

Input

Variable name	Type	Description
k_s	double	Extinction coefficient
L_s_c	double	Compressed leaf area index of the shorter crop
P	double	Strip width of the taller crop
SP_ni	double	Fraction of the global radiation that is intercepted by the shorter crop.

W	double	Strip width of the shorter crop

Output

The method StripIntercrop.CalculateFractionOfRadiationAtTopLowerCanopyTallerCrop(double k, double L, double L_c, double IR_b) calculates the radiation of the global radiation that enter the top of the lower canopy of the taller crop and returns the calculated value.

6.18.12 M3_library.StripIntercrop. CalculateFractionOfRadiationAtTopCanopyShorterCrop

The method CalculateFractionOfRadiationAtTopCanopyShorterCrop(double k, double L, double IP_b) calculates the radiation of the global radiation that enters the top of the canopy of the shorter crop:

$$f_{\mathrm{trans,s}}(t) = V_{\mathrm{s}}(t) + \left(1 - V_{\mathrm{s}}(t)\right) \cdot \mathrm{e}^{-k_{\mathrm{t}}(t) \cdot L_{u,\mathrm{t}}}$$

Symbol	Description	Unit
$f_{\mathrm{trans,s}}(t)$	Fraction of global radiation that enters the top of the canopy of the shorter crop	
$k_{\mathrm{t}}(t)$	Extinction coefficient of taller crop	
$L_{\mathrm{u,t}}(t)$	Leaf area index of the upper canopy of the taller crop	$m^2\ m^{-2}$
$V_{\mathrm{s}}(t)$	View factor of the sky of spaces between the strips of the taller crop	

Input

Variable name	Type	Description
k	double	Extinction coefficient
L	double	Compressed leaf area index of the shorter crop
IP_b	double	Strip width of the taller crop

6.18.13 M3_library.StripIntercrop.CalculateWeightFunction

The method StripIntercrop.CalculateWeightFunction(double k, double L_c, double SP_ni, double SR_ni) calculates the weight factor that expresses to what extend the top canopy of the taller crop represents a compressed intercrop.

$$w = \begin{cases} 0 & | \quad L_{\mathrm{c,u,t}}(t) = 0 \\ \dfrac{f_{\mathrm{trans,s}}(t) - f_{\mathrm{trans,u,t}}(t)}{1 - \mathrm{e}^{-k_{\mathrm{t}}(t) \cdot L_{\mathrm{c,u,t}}}} & | \quad L_{\mathrm{c,u,t}}(t) > 0 \end{cases}$$

Symbol	Description	Unit
$f_{\mathrm{trans,s}}(t)$	Fraction of global radiation that enters the top of the canopy of the shorter crop	
$f_{\mathrm{trans,u,t}}(t)$	Fraction of global radiation that enters the top of the lower canopy of the taller crop	
$k_{\mathrm{t}}(t)$	Extinction coefficient of the taller crop	
w	Weight factor that describes to what extend the top canopy of the intercrop resembles a compressed canopy	

Input

Variable name	Type	Description
$f_{\mathrm{trans,s}}(t)$	double	Fraction of global radiation that enters the top of the canopy of the shorter crop
$f_{\mathrm{trans,u,t}}(t)$	double	Fraction of global radiation that enters the top of the lower canopy of the taller crop
k_t	double	Extinction coefficient
L_c	double	Compressed leaf area index

Output

The method StripIntercrop.CalculateWeightFunction(double k, double L_c, double SP_ni, double SR_ni) calculates the weight factor and returns the calculated value.

6.18.14 M3_library.StripIntercrop.CalculateViewFactorUpperCanopyTallerCrop

The method StripIntercrop.CalculateViewFactorUpperCanopyTallerCrop(double W, double H_diff) calculates the view factor of the upper canopy of the taller crop:

$$V_{\mathrm{t}}(t) = \frac{\sqrt{\left(\Delta H(t)\right)^2 + \ell_{\mathrm{t}}(t)^2} - \Delta H(t)}{\ell_{\mathrm{t}}(t)}$$

Symbol	Description	Unit
$\ell_{\mathrm{t}}(t)$	Strip width of the taller crop	m
$V_{\mathrm{t}}(t)$	View factor of the sky of the upper canopy of the taller crop	
$\Delta H(t)$	Height difference between taller and shorter crop	m

Input

Variable name	Type	Description
H_diff	double	Height difference between taller and shorter crop
W	double	Strip width of the taller crop

Output

The method StripIntercrop.CalculateViewFactorUpperCanopyTallerCrop(double W, double H_diff) calculates the view factor of the upper canopy of the taller crop and returns its value.

6.18.15 M3_library.StripIntercrop.CalculatViewFactorSpaceBetweenRowsUpperCanopy

The method StripIntercrop.CalculatViewFactorSpaceBetweenRowsUpperCanopy (double P, double H_diff) calculates the view factor of the sky between the upper canopies of the taller crop strips:

$$V_{\mathrm{s}}(t) = \frac{\sqrt{\left(\Delta H(t)\right)^2 + \ell_{\mathrm{s}}(t)^2} - \Delta H(t)}{\ell_{\mathrm{s}}(t)}$$

Symbol	Description	Unit
$\ell_s(t)$	Strip width of the taller crop	m
$V_s(t)$	View factor of the sky of the upper canopy of the taller crop	
$\Delta H(t)$	Height difference between taller and shorter crop	m

Input

Variable name	Type	Description
H_diff	double	Height difference between taller and shorter crop
P	double	Strip width of the shorter crop

Output

The method StripIntercrop. CalculatViewFactorSpaceBetweenRowsUpperCanopy (double W, double H_diff) calculates the view factor of the sky between the upper canopies of the taller crop strips and returns its value.

6.18.16 M3_library.StripIntercrop.FindIndexOfTallerCrop

The method FindIndexOfTallerCrop determines which of the two species in the Crop array crops is the taller species.

Input

None

Output

Array index of the taller species.

6.18.17 M3_library.StripIntercrop.FindIndexOfShorterCrop

The method FindIndexOfTallerCrop determines which of the two species in the Crop array crops is the shorter species.

Input

None

Output

Array index of the shorter species.

6.18.18 M3_library.StripIntercrop.FindTallerCrop

StripIntercrop.FindTallerCrop() checks for each Crop object in the Crop array crops what the height is. It compares the height of the first and the second element of the crop array. If the first crop in this array is taller than the second one, the Boolean class variable crop1IsTallerCrop is set to "true" and crop1IsShorterCrop is set to "false. If the first crop in this array is shorter than the second one, the Boolean class variable crop1IsTallerCrop is set to "false" and crop1IsShorterCrop is set to "true". If the first and the second crop have equal heights, both crop1IsTallerCrop and crop1IsShorterCrop are set to "false".

Input

None

Output

StripIntercrop.FindTallerCrop() determines the values of the StripIntercrop class variables crop1IsTallerCrop and crop1IsShorterCrop, updates their values, and returns the StripIntercrop object.

6.18.19 M3_library.StripIntercrop.GrowIntercrop

The method StripIntercrop.GrowIntercrop() calls the function crop.GrowCrop for each crop in the Crops array that are a part of the intercrop.

Input

None

Output

StripIntercrop.GrowIntercrop() calls the function crop.GrowCrop for each crop in the Crops array that are a part of the intercrop. The method crop.GrowCrop calculates the state of the crop objects in the Crops array crops in the next time steps. It returns the updated StripIntercropObject.

6.19 M3_library.StripIntercropReader

The file StripIntercropReader.cs contains the class StripIntercropReader. The purpose of StripIntercropReader is to read all the crop parameter files corresponding to the crops in the intercrops. The table below gives an overview of the class variables and their meanings:

Variable	**Type**	**Meaning**
cropFileReader	CropFileReader	CropFileReader object
filePaths	string array	String array that contains the file paths of crop files

6.19.1 M3_library.StripIntercropReader.ReadStripIntercropFromCSV

The method StripIntercropReader.ReadStripIntercropFromCSV reads the crop files corresponding to all crops in the intercrop. The entries of the class variable filePaths, which is a string array, contain file paths of the crop files corresponding to the crops that were grown in the intercrop. It initialize a number of Crop objects that is equal to the number of entries in filePaths loops through the file paths of the crop file, uses a CropFileReader object to initialize a Crop object for each file path, stores all Crop objects in a Crop array, and assigns this Crop array to a StripIntercrop object.

Input

None

Output

The method StripIntercropReader.ReadStripIntercropFromCSV reads crop files, initializes crop object based on the information in these crop files and stores them in a Crop array crops, assigns the Crop array to a StripIntercrop object, and returns this StripIntercropObject.

6.20 M3_library.Timer

The file Timer.cs contains the class Timer. This class holds the simulated date, the start date, end the end date of the simulations. It also includes a method to add a day. The table below shows an overview of the class variables, their types, and their meaning:

Variable name	**Type**	**Meaning**
date	DateTime	Date
startSimulationDate	DateTime	Date at which the simulation starts
endSimulationDate	DateTime	Date at which the simulation ends

6.20.1 M3_library.TimerAddDay

Timer.AddDay adds one day to the class variable date.

Input

None

Output

Timer.AddDay adds one day to the class variable date and returns the updated Timer object.

6.20.2 M3_library.TimerSubtractDay

Timer.AddDay subtracts one day to the class variable date.

Input

None

Output

Timer.AddDay subtracts one day to the class variable date and returns the updated Timer object.

6.21 M3_library.WeatherFileReader

The file WeatherFileReader.cs contains the class WeatherFileReader. The purpose of this class is to read daily weather data (date, daily global radiation, minimum daily temperature, and maximum daily temperature) from a weather file. It selects only the weather data for a particular weather station. Weather stations can be distinguished by their indices and the class variable weatherStation provides this index. The table below shows an overview of the class variables, their types, and their meaning:

Variable name	Type	Meaning
weatherStation	WeatherStation	WeatherStation object that contains information of the weather station, which is used for the simulation

6.21.1 M3_library.WeatherStation.ReadWeatherFileFromCSV

The method WeatherStation.ReadWeatherFileFromCSV(int weatherStationID, string weatherFilePath) reads through a weather input file. The weatherFilePath input variable of this method contains the path of the weather file to be read. The weatherStationID indicates for which weather station the weather data have to be stored. The method loops through the weatherFile and reads the values of the variables (date, minimum daily temperature, maximum daily temperature, global daily radiation) for each record. The method also checks for each record weather has the same value for the weather station identifier as the input variable weatherStationID. If this is the case, the weather data will be added to a list of dailyWeatherData objects. After the whole weather file has been read, the list with dailyWeatherData objects is converted in an array and returned.

Input

Symbol	Type	Description
weatherFilePath	string	Path of the weather file that has to be read
weatherStationID	int	The weather station identifier for which the weather data have to be read and stored.

Output

The method WeatherStation.ReadWeatherFileFromCSV(int weatherStationID, string weatherFilePath) reads all weather data from the weather file with file path weatherFilePath. It returns an array of DailyWeather

objects, which contains all weather data from the weather file with the same weather station identifier as the value of the input variable weatherStationID.

6.21 M3_library.WeatherStation

The file WeatherStation.cs contains the class WeatherStation. This purpose of the class WeatherStation is to store geographical information about the weather station (city/weather station name, country, longitude, latitude), its index (which distinguishes it from other weather stations) and an array of daily weather objects that were observed by this weather station. It also contains a function to collect the daily weather data for this weather station. The table below shows the class variables, their types, their meaning, and their units:

Variable name	Type	Description	Unit
city	string	City in which the weather station is located or name of the weather station	
country	string	Country in which the weather station is located	
latitude	double	Latitude	°
longitude	double	Longitude	°
weatherStationID	int	Weather station identifier	
weatherRecords	DailyWeather array	Weather data array that contains weather data that were observed by this weather station	

6.21.1 M3_library.WeatherStation.GetDailyWeatherData

The method GetDailyWeatherData(DateTime date) loops through the DailyWeather array weatherRecords and checks for each dailyWeather element whether its date equals the input variable date. If it is, the Boolean isDailyWeatherFound is set from false to true and the loop stops. The record for which its date equals the input variable date is returned.

Input

Symbol	Type	Description
date	DateTime	The date for which the daily weather record is read

Output

The method GetDailyWetherData returns an instance of the DailWeather object in the class variable weatherRecords, which has the same date as the input variable date.

6.22 M3_library.WeatherStationReader

The file WeatherStationReader.cs contains the class WeatherStationReader. The purpose of the class WeatherStationReader is to read parameters from a weather station file and use these parameters to create a WeatherStation object. The table below shows this class variable, its type, its meaning.

Variable name	Type	Meaning
weatherStation	WeatherStation	WeatherStation object that holds weather records and other attributes of the weather station that is read.

6.22.1 M3_library.WeatherStationReader.ReadWeatherStationFromFile

The method ReadWeatherStationFromFile(int weatherStationID, string filePath) reads a weather station file. The path of this file is specified in the input string variable filePath. It extracts the weather station ID (first column; index 0), the country (second column; index 1), the city/name of the station (third column, index 2), the latitude (fourth column, index 3), and the longitude (fifth column, index 4). For each record of weather stations in the weather station file, it compare the weather station ID in the file and the input variable weatherStationID. If they match, this record is used to declare and initialize a weatherStation objects and returns this object.

Input

Variable nam	Type	Meaning
weatherStationID	int	Identifier of the weather station that is extracted.
weatherFilePath	string	Path that contains the weather station file that is read.

Output

The method ReadWeatherStationFromFile(int weatherStationID, string filePath) returns a WeatherStation object with the weather station ID that matches the input variable integer weatherStationID.

7 Model input files

7.1 Crop files

A crop file is a *.CSV (comma separated values) file that contains the parameters of a crop species. It consists of 4 columns and 29 rows. The first row is the header. The subsequent 28 rows contain information about the parameters. Each row contains four values:

Column	Column title	Description
1	Symbol	Symbol for a parameter
2	Meaning	Meaning of the parameter
3	Value	Parameter value
4	Unit	Unit of the parameter

The method M3_library.CropFileReader.ReadCropFromCSV reads crop files. It uses row 1 to recognize which parameter it represents and row 3 to see what its value is. It then assigns this value to the correct variable in the code of the model. Although the model does not use the information in row 2 and 4 in the calculations, these columns must be present in the crop file and it is recommended that the values of these columns are not changed. The table below shows the symbol of each parameter that needs to be present in this field, the symbol that was used in the manuscript that introduces the M^3 model, a description, and a unit of the parameter.

Symbol in code	Symbol in manuscript (Berghuijs *et al.*, 2020)	Description	Unit
A_lv0	$A_{0,\mathrm{i}}$	Initial leaf area per plant	$m^2\ plant^{-1}$
coeffACrit	$\alpha_{\mathrm{crit,i}}$	Biomass at which the critical nitrogen content would be 1, in case the critical nitrogen was not kept at a constant value at low biomasses.	$kg\ m^{-2}$
coeffAMax	$\alpha_{\mathrm{max,i}}$	Biomass at which the maximum nitrogen content would be 1, in case the maximum nitrogen was not assumed at a constant value at low biomass	$kg\ m^{-2}$
coeffAMin	$\alpha_{\mathrm{min,i}}$	Biomass at which the minimum nitrogen content would be 1, in case the minimum nitrogen was not kept at a constant value at low biomass	$kg\ m^{-2}$
coeffBCrit	$\beta_{\mathrm{crit,i}}$	Shape parameter of critical nitrogen content	
coeffBMax	$\beta_{\mathrm{max,i}}$	Shape parameter of maximum nitrogen content	
coeffBMin	$\beta_{\mathrm{min,i}}$	Shape parameter of minimum nitrogen content	
dev_1	$d_{1,\mathrm{i}}$	Developmental stage below which the partitioning fraction to the leaves equals $f_{\mathrm{l},0}$	
dev_2	$d_{2,\mathrm{i}}$	Developmental stage above which no more biomass is partitioned to the leaves	

dev_y1	$d_{3,\mathrm{i}}$	Developmental stage above which biomass is partitioned to the storage organs	
dev_y2	$d_{4,\mathrm{i}}$	Developmental rate above which no more biomass is partitioned to other plant organs than the leaves and the storage organs	
f_lv_0	$f_{\mathrm{l},0}$	Partitioning fraction to the leaves if the developmental rate is below $\delta_{1,i}$	
f_NFix	$f_{\mathrm{fix,i}}$	Fraction of the nitrogen demand that can be obtained by N_2 fixation	
f_NCrit	$f_{\mathrm{crit,i}}$	Fraction of critical nitrogen content to maximum nitrogen content at low biomass	
f_NMin	$f_{\mathrm{min,i}}$	Fraction of minimum nitrogen content to maximum nitrogen content at low biomass	
h_0	$H_{0,\mathrm{i}}$	Initial plant height	m
h_max	$H_{\mathrm{m,i}}$	Maximum plant height	m
k	k_{i}	Extinction coefficient	
Nmax0	$\nu_{\mathrm{max},0,\mathrm{i}}$	Maximum crop nitrogen content at low biomass	
rgrh	$r_{\mathrm{h,i}}$	Temperature-sensitive relative growth height of plant height	$^{o}C^{-1}$ d^{-1}
rue	η_{i}	Radiation use efficiency	kg J^{-1}
s_la	$S_{\mathrm{la,i}}$	Specific leaf area	m^2 kg^{-1}
s_sen_t	$s_{\mathrm{sen,i}}$	Sensitivity of leaf senescence to temperature	$^{o}C^{-1}$ d^{-1}
t_b	$T_{\mathrm{b,i}}$	Base temperature	^{o}C
t_min_sen	$T_{\mathrm{sen,i}}$	Minimum temperature above which senescence takes place	^{o}C
t_sum_em_flo	$S_{\mathrm{a,i}}$	Temperature sum from emergence to anthesis	^{o}C d
t_sum_flo_mat	$S_{\mathrm{m,i}}$	Temperature sum from anthesis to maturity	^{o}C d
t_sum_so_em	$S_{\mathrm{e,i}}$	Temperature from sowing to emergence	^{o}C d

7.2 Farmer files

A farmer file is a *.CSV (comma separated value) file that contains management parameters. It is read by the method M3_library.FarmerReader.ReadFarmerFile. It consists of 17 columns. The first 4 columns contain data about the sowing dates of the species in the intercrop and of the fertilization dates. The first column contains either the word date_sow or date_nfert, which indicates whether the third column contain a sowing date or a date at which nitrogen is applied by fertilizer application. The number of rows containing a sowing date is always 2, one per species, even in pure cultures. If one wants to simulate a pure culture, a very low values (for instance 10^{-10}) for both the sowing density and the strip width have to be given for the second crop, such that its contribution to the light interception and the nitrogen uptake is negligible. The number of rows that contain a nitrogen fertilizer application event depends and can vary between 0 and infinity. Columns 5 until 8 contain information about the sowing density and the rate of nitrogen application. If the first four columns in a row contain information about the sowing date, column 7 contains a sowing density (seeds m^{-2}). If the first four columns in a row contain information about a nitrogen fertilization event, column 7 contains a fertilizer rate (kg N m^{-2}). Columns 9 until 12 contain information about the harvest dates. If a harvest date is in the same row as the sowing date, the crop species that is sown at that sowing date is

harvested at the harvest date in the same row. Column 13 until 16 contain information about the strip widths (m). If the first four columns in a row contain information about a sowing date, column 15 contains a strip width for the species that is sown at that date. If the first four columns in a row contain information about a sowing date, the final two columns (16 and 17) contain the directory and the file name of the crop parameter file of the crop species that is sown at that date. The column titles in the first row are:

Column	Column title	Description
1	Symbol	Symbol for a management parameter. It equals date_sow for a sowing date and date_nfrert for the date of a nitrogen fertilizer application event.
2	Meaning	Indicates whether the third column contains a fertilizer date or a sowing date
3	Value	Contains the date of either as sowing or a nitrogen fertilization event.
4	Unit	Date format
5	Symbol	
6	Meaning	
7	Value	
8	Unit	
9	Symbol	Contains date_har (harvest date)
10	Meaning	Date of a harvest event
11	Value	Harvest date
12	Unit	Date format
13	Symbol	Contains strip_width (strip width)
14	Meaning	Width of a strip of a species in an intercrop
15	Value	Strip width
16	Unit	Unit of the strip width (m)
17	CropParameterFileDirectory	File directory of a crop file
18	CropParameterFileName	File name of a crop file

7.3 Simulation input files

A simulation input file is a *.CSV (comma separated) file that contains the names and directories of various input files for each simulation that is run if the project RunSimulationsFromInputFile is executed. It contains 13 columns. The first row contains a header. The other rows contain the file names and directories. Each row has 13 values:

Column	Column title	Description
1	RunID	ID of the simulation
2	OutputFileDirectory	Directory to where the output file of the simulation is stored
3	OutputFileName	File name of the output file
4	Farmer file directory	Directory containing the farmer file that is used in the simulation.
5	Farmer file name	Name of the farmer file that is used in the simulation
6	Soil file directory	Directory which contains the soil file that is used in the simulation
7	Soil file name	Name of the soil file that is used in the simulation
8	Weather station ID	ID of the weather station that is used in the simulation
9	Weather station file directory	Directory which contains the weather station file that is used in the simulation
10	Weather station file name	Name of the weather station file that is used in the simulation

11	Weather file directory	Directory which contains the weather file that is used in the simulation
12	Weather file name	Name of the weather file that is used in the simulation

7.4 Soil files

A soil file is a *.CSV (comma separated value) file that contains soil specific parameters. It consists of 4 columns and 3 rows. The first row contains a header. The other 3 rows contain information about the parameters. Each row contains four values:

Column	Column title	Description
1	Symbol	Symbol for a parameter
2	Meaning	Meaning of the parameter
3	Value	Parameter value
4	Unit	Unit of the parameter

The method M3_library.CropFileReader.ReadSoilFile reads soil files. It uses row 1 to recognize which parameter it represents and row 3 to see what its value is. It then assigns this value to the correct variable in the code of the model. Although the model does not use the information in row 2 and 4 in the calculations, these columns must be present in the crop file and it is recommended that the values of these columns are not changed. The table below shows the symbol of each parameter that needs to be present in this field, the symbol that was used in the manuscript that introduces the M^3, a description, and a unit of the parameter.

Symbol in code	Symbol in manuscript	Description	Unit
f_recov	f_{recov}	Recovery fraction of nitrogen in fertilizer	
initial_n	$N_{\mathrm{s},0}$	Initial amount of soil mineral nitrogen	kg m^{-2}
n_min_rate	M	Net mineralization rate	kg m^{-2} d^{-1}

7.5 Weather files

A weather file is a *.CSV (comma separated values) file that contains daily weather observations. It consists of 11 columns. The first row is a header. Each other row represent a daily weather observation for a particular weather station. The method M3_library.WeatherFileReader.ReadWeatherFromCSV reads weather files. Each weather observation and contains 11 values. The weather station index is used to link the weather observations to weather stations specified in the weather station file.

Column	Column title	Description	Units
1	stn	Weather station index	
2	date	Date	yyyy-mm-dd
3	doy	Day of year	
4	day	Day of month	
5	month	Month number	
6	year	Year	
7	tmin	Minimum daily temperature	^{o}C
8	tmax	Maximum daily temperature	^{o}C
9	precipitation	Daily precipitation	mm d^{-1}
10	irradiation	Daily global radiation	MJ m^{-2} d^{-1}

11	remark		

7.6 Weather station files

A weather station file is a *.CSV (comma separated values) file that contains the parameters that describe a weather station. It consists of five columns and a number of rows that equals the number of available weather stations. Each row contains five values:

Column	Column title	Description
1	WeatherStationID	Index of the weather station
2	Country	Country where the weather station is located.
3	City	City where the weather station is located
4	Latitude	Geographical latitude (decimals) of the weather station
5	Longitude	Geographical longitude (decimals) of the weather station

The method M3_library.WeatherStationReader.ReadWeatherStationFromFile reads weather station files. It uses the first row to recognize what the identifier of the weather station is. The values in the other rows are not used, but it is recommended that the values are not changed.

8. Output files

Output files are *.CSV files (comma separated values) files that contain the output of a single simulation. It contains 10 columns that are not crop specific and contain values of soil related variables and time related variables. The remaining 50 columns contain output variables that are specific for one of the two crop species in the intercrop (25 columns per crop species). The meaning of the variables in these columns are:

Non-crop specific variables

Column	**Column title**	**Meaning**	**Unit**
1	Date	Date	yyyy - mm - dd
2	Year	Year	
3	Day of year	Day of the year	
4	Month	Month number	
5	Day	Day of the month	
6	Soil mineral nitrogen amount (kg m-2)	Amount of soil mineral nitrogen	kg m^{-2}
7	Net mineralization rate (kg N m^{-2} d^{-1})	Net mineralization rate	kg m^{-2} d^{-1}
8	Fertilization rate (kg N m-2 d-1)	Amount of applied fertilizer	kg m^{-2} d^{-1}
9	Nitrogen uptake rate (kg m-2 d-1)	Rate at which soil mineral nitrogen is removed by water uptake by the crop species in the intercrop	kg m^{-2} d^{-1}
10	Soil mineral nitrogen net growth rate (kg N m-2 d-1)	Daily growth rate of the amount of soil mineral nitrogen	kg m^{-2} d^{-1}

Crop specific variables

Column	Column title	Meaning	Unit
1	Development rate of species 1	Developmental stage of species 1	
2	Emergence	Indicate whether or not the species has emerged	
3	Aboveground dry matter weight of species 1 (kg m-2)	Aboveground dry matter weight of species 1	kg m^{-2}
4	Aboveground dry matter production of species 1 (kg m-2 d-1)	Rate of aboveground dry matter production of species 1	kg m^{-2} d^{-1}
5	Aboveground dry mortality of species 1 (kg m-2 d-1)	Rate of aboveground dry matter removal of species 1	kg m^{-2} d^{-1}
6	Leaf area index of species 1 (-)	Leaf area index of species 1	m^2 m^{-2}
7	Leaf area index of species 1 production (d-1)	Rate of leaf area index production of species 1	m^2 m^{-2} d^{-1}
8	Leaf area index of species 1 mortality (d-1)	Rate of leaf area index removal of species 1	m^2 m^{-2} d^{-1}
9	Fraction of light intercepted	Fraction of global radiation that is intercepted by species 1	
10	Shoot height of species 1 (m)	Shoot height of species 1	m
11	Shoot height growth of species 1 (m d-1)	Rate of shoot height growth of species 1	kg m^{-2}
12	Storage organ dry matter weight 1 (kg m-2 d-1)	Dry weight of storage organs of species 1	kg m^{-2}
13	Storage organ dry matter production 1 (kg m-2 d-1)	Rate of storage organ production of species 1	kg m^{-2} d^{-1}
14	Crop nitrogen amount of species 1 (kg N m-2)	Amount of nitrogen that is stored in species 1	kg m^{-2} d^{-1}
15	Crop nitrogen total demand 1 (kg N m-2)	Nitrogen demand of species 1	kg m^{-2} d^{-1}
16	Crop nitrogen demand from soil 1 (kg N m-2)	Nitrogen demand of species 1 that can be fulfilled by nitrogen uptake from the soil.	kg m^{-2} d^{-1}
17	Crop nitrogen uptake rate of species 1 (kg N m-2 d-1)	Rate of nitrogen uptake by species 1	kg m^{-2} d^{-1}
18	Crop nitrogen fixation rate of species 1 (kg N m-2 d-1)	Rate of N_2 fixation by species 1	kg m^{-2} d^{-1}
19	Crop nitrogen loss by senescence of species 1 (kg N m-2 d-1)	Rate of crop nitrogen loss	kg m^{-2} d^{-1}
20	Crop nitrogen net growth of species 1 (kg N m-2 d-1)	Growth rate of the amount of nitrogen in the crop	kg m^{-2} d^{-1}
21	Crop nitrogen content of species 1 (kg N kg-1)	Crop nitrogen content of species 1	kg m^{-2} d^{-1}
22	Maximum nitrogen content of species 1 (kg kg-1)	Maximum nitrogen content of species 1	kg kg^{-1}
23	Critical nitrogen content of species 1 (kg kg-1)	Critical nitrogen content of species 1	kg kg^{-1}
24	Minimum nitrogen content of species 1 (kg kg-1)	Minimum nitrogen content of species 1	kg kg^{-1}
25	Crop growth reduction factor of species 1 (-)	Crop growth reduction factor due to nitrogen deficiency in species 1	kg kg^{-1}
26	Development rate of species 2	Developmental stage of species 2	
27	Emergence	Indicate whether or not the species has emerged	
28	Aboveground dry matter weight of species 2 (kg m-2)	Aboveground dry matter weight of species 2	kg m^{-2}
29	Aboveground dry matter production of species 2 (kg m-2 d-1)	Rate of aboveground dry matter production of species 2	kg m^{-2} d^{-1}
30	Aboveground dry mortality of species 2 (kg m-2 d-1)	Rate of aboveground dry matter removal of species 2	kg m^{-2} d^{-1}
31	Leaf area index of species 2 (-)	Leaf area index of species 2	m^2 m^{-2}

32	Leaf area index of species 2 production (d-1)	Rate of leaf area index production of species 2	$m^2\ m^{-2}\ d^{-1}$
33	Leaf area index of species 2 mortality (d-1)	Rate of leaf area index removal of species 2	$m^2\ m^{-2}\ d^{-1}$
34	Fraction of light intercepted	Fraction of global radiation that is intercepted by species 2	
35	Shoot height of species 2 (m)	Shoot height of species 2	m
36	Shoot height growth of species 2 (m d-1)	Rate of shoot height growth of species 2	$kg\ m^{-2}$
37	Storage organ dry matter weight 2 (kg m-2 d-1)	Dry weight of storage organs of species 2	$kg\ m^{-2}$
38	Storage organ dry matter production 2 (kg m-2 d-1)	Rate of storage organ production of species 2	$kg\ m^{-2}\ d^{-1}$
39	Crop nitrogen amount of species 2 (kg N m-2)	Amount of nitrogen that is stored in species 2	$kg\ m^{-2}\ d^{-1}$
40	Crop nitrogen total demand 2 (kg N m-2)	Nitrogen demand of species 2	$kg\ m^{-2}\ d^{-1}$
41	Crop nitrogen demand from soil 2 (kg N m-2)	Nitrogen demand of species 2 that can be fulfilled by nitrogen uptake from the soil.	$kg\ m^{-2}\ d^{-1}$
42	Crop nitrogen uptake rate of species 2 (kg N m-2 d-1)	Rate of nitrogen uptake by species 2	$kg\ m^{-2}\ d^{-1}$
43	Crop nitrogen fixation rate of species 2 (kg N m-2 d-1)	Rate of N_2 fixation by species 2	$kg\ m^{-2}\ d^{-1}$
44	Crop nitrogen loss by senescence of species 2 (kg N m-2 d-1)	Rate of crop nitrogen loss	$kg\ m^{-2}\ d^{-1}$
45	Crop nitrogen net growth of species 2 (kg N m-2 d-1)	Growth rate of the amount of nitrogen in the crop	$kg\ m^{-2}\ d^{-1}$
46	Crop nitrogen content of species 2 (kg N kg-1)	Crop nitrogen content of species 2	$kg\ m^{-2}\ d^{-1}$
47	Maximum nitrogen content of species 2 (kg kg-1)	Maximum nitrogen content of species 2	$kg\ kg^{-1}$
48	Critical nitrogen content of species 2 (kg kg-1)	Critical nitrogen content of species 2	$kg\ kg^{-1}$
49	Minimum nitrogen content of species 2 (kg kg-1)	Minimum nitrogen content of species 2	$kg\ kg^{-1}$
50	Crop growth reduction factor of species 2 (-)	Crop growth reduction factor due to nitrogen deficiency in species 2	$kg\ kg^{-1}$

9. References

Allen, R. G., Pereira, L. S., Raes, D. & Smith, M. (1998). *Crop evapotranspiration - Guidelines for computing crop water requirements - FAO Irrigation and drainage paper 56.* Rome: FAO - Food and Agricultural Organization of the United Nations.

Berghuijs, H. N. C., Wang, Z., Stomph, T. J., Weih, M., Van der Werf, W. & Vico, G. (2020). Identification of species traits enhancing yield in wheat-faba bean intercropping: development and sensitivity analysis of a minimalist mixture model. *Plant and Soil* 455(1-2): 203-226.

Gou, F., van Ittersum, M. K., Simon, E., Leffelaar, P. A., van der Putten, P. E. L., Zhang, L. Z. & van der Werf, W. (2017). Intercropping wheat and maize increases total radiation interception and wheat RUE but lowers maize RUE. *European Journal of Agronomy* 84: 125-139.

Goudriaan, J. (1977).Crop micrometeorology : a simulation study. Wageningen: Pudoc.

Hejlsberg, A., Torgersen, M., Wiltamuth, S. & Golde, P. (2008). *The C# Programming Language.* Addison-Wesley Professional.

Pronk, A. A., Goudriaan, J., Stilma, E. & Challa, H. (2003). A simple method to estimate radiation interception by nursery stock conifers: a case study of eastern white cedar. *Njas-Wageningen Journal of Life Sciences* 51(3): 279-295.