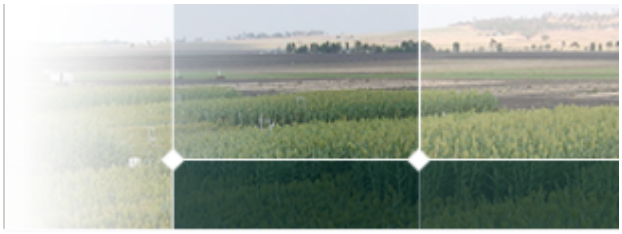


1 The APSIM Slurp Model

1.1 SLURP: the Sound of a crop using water

This model has been built using the Plant Modelling Framework (PMF) of Brown et al., 2014 to provide a simple representation of crops. It is usefull for water and nitrogen balance studies where the focus is on soil processes and a very simple crop is adequate. The model does not predict crop growth, development or yields. It simply takes up water and nitrogen. .

SLURP has no phenology so the behaviour of the SLURP is the same throughout its ontology (with the exception of root depth being able to increase). SLURP consists of 2 organs:

1. Leaf which Is represented with a Simpleleaf class and provides a Water uptake demand to the soil arbitrator.
2. Root which extracts water and nitrogen from the soil for plant growth

1.2 Inclusion in APSIM simulations

A Slurp crop is included in a simulation the same as any other APSIM crop

* The Slurp object needs to be dragged or copied from the Crop folder in the tool box into the field of your simulation.
* It is then planted with a sowing rule

```
Slurp.Sow(cultivar: StaticCrop, population: 1, depth: 10, rowSpacing: 150);
```

* Note that SLURP has no notion of population or rowSpacing but these parameters are required by the Sow method so filler values are provided
* `depth` in the Sow arguement is the depth that the crop is sown at. While this has no effect on emergence in SLURP it sets the depth that the root system is initialised at. For static crops this depth should be set to the rooting depth that is expected as roots will not grow during the simulation

1.3 Altering Slurp properties during runs

In some cases users will wish to change properties of Slurp while the simulation is running. This can be done using a the set method in a manager script.

```
object LAIResetValue = leaflai;
zone.Set("Slurp.Leaf.LAIFunction.Value()", LAIResetValue);
object HeightResetValue = CoverToday * MaximumHeight;
zone.Set("Slurp.Leaf.HeightFunction.Value()", HeightResetValue);
```

The model has been developed using the Plant Modelling Framework (PMF) of Brown et al., 2014. This new framework provides a library of plant organ and process submodels that can be coupled, at runtime, to construct a model in much the same way that models can be coupled to construct a simulation.This means that dynamic composition of lower level process and organ classes(e.g.photosynthesis, leaf) into larger constructions(e.g.maize, wheat, sorghum) can be achieved by the model developer without additional coding.

The model is constructed from the following list of software components. Details of the implementation and model parameterisation are provided in the following sections.

List of Plant Model Components.

Component Name	Component Type
Phenology	Models.PMF.Phen.Phenology

Component Name	Component Type
Arbitrator	Models.PMF.OrganArbitrator
Root	Models.PMF.Organs.Root
Leaf	Models.PMF.Organs.SimpleLeaf
MortalityRate	Models.Functions.Constant

1.4 Phenology

The phenological development is simulated as the progression through a series of developmental phases, each bound by distinct growth stage.

1.4.1 ThermalTime

ThermalTime = 1 (days)

List of stages and phases used in the simulation of crop phenological development

Phase Number	Phase Name	Initial Stage	Final Stage
1	Initial	Initialisation	Sowing
2	Ongoing	Sowing	Never

1.4.2 Initial

This phase goes from initialisation to sowing.

The *Target* for completion is calculated as:

Target = 0 (d)

Progression through phase is calculated daily and accumulated until the *Target* is reached.

Progression = [Phenology].ThermalTime

1.4.3 Ongoing

This phase goes from sowing to never.

The *Target* for completion is calculated as:

Target = 1000000000000000 (d)

Progression through phase is calculated daily and accumulated until the *Target* is reached.

Progression = [Phenology].ThermalTime

1.4.4 Constants

ThermalTime = 1 (days)

1.5 Arbitrator

1.5.1 Arbitrator

The Arbitrator class determines the allocation of dry matter (DM) and Nitrogen between each of the organs in the crop model. Each organ can have up to three different pools of biomass:

* **Structural biomass** which is essential for growth and remains within the organ once it is allocated there.
* **Metabolic biomass** which generally remains within an organ but is able to be re-allocated when the organ senesces and may be retranslocated when demand is high relative to supply.
* **Storage biomass** which is partitioned to organs when supply is high relative to demand and is available for retranslocation to other organs whenever supply from uptake, fixation, or re-allocation is lower than demand.

The process followed for biomass arbitration is shown in the figure below. Arbitration calculations are triggered by a series of events (shown below) that are raised every day. For these calculations, at each step the Arbitrator exchange

information with each organ, so the basic computations of demand and supply are done at the organ level, using their specific parameters.

1. **doPotentialPlantGrowth**. When this event occurs, each organ class executes code to determine their potential growth, biomass supplies and demands. In addition to demands for structural, non-structural and metabolic biomass (DM and N) each organ may have the following biomass supplies:

* **Fixation supply**. From photosynthesis (DM) or symbiotic fixation (N)
* **Uptake supply**. Typically uptake of N from the soil by the roots but could also be uptake by other organs (eg foliage application of N).
* **Retranslocation supply**. Storage biomass that may be moved from organs to meet demands of other organs.
* **Reallocation supply**. Biomass that can be moved from senescing organs to meet the demands of other organs.

1. **doPotentialPlantPartitioning.** On this event the Arbitrator first executes the DoDMSetup() method to gather the DM supplies and demands from each organ, these values are computed at the organ level. It then executes the DoPotentialDMAllocation() method which works out how much biomass each organ would be allocated assuming N supply is not limiting and sends these allocations to the organs. Each organ then uses their potential DM allocation to determine their N demand (how much N is needed to produce that much DM) and the arbitrator calls DoNSetup() to gather the N supplies and demands from each organ and begin N arbitration. Firstly DoNReallocation() is called to redistribute N that the plant has available from senescing organs. After this step any unmet N demand is considered as plant demand for N uptake from the soil (N Uptake Demand).
2. **doNutrientArbitration.** When this event occurs, the soil arbitrator gets the N uptake demands from each plant (where multiple plants are growing in competition) and their potential uptake from the soil and determines how much of their demand that the soil is able to provide. This value is then passed back to each plant instance as their Nuptake and doNUptakeAllocation() is called to distribute this N between organs.
3. **doActualPlantPartitioning.** On this event the arbitrator call DoNRetranslocation() and DoNFixation() to satisfy any unmet N demands from these sources. Finally, DoActualDMAllocation is called where DM allocations to each organ are reduced if the N allocation is insufficient to achieve the organs minimum N concentration and final allocations are sent to organs.

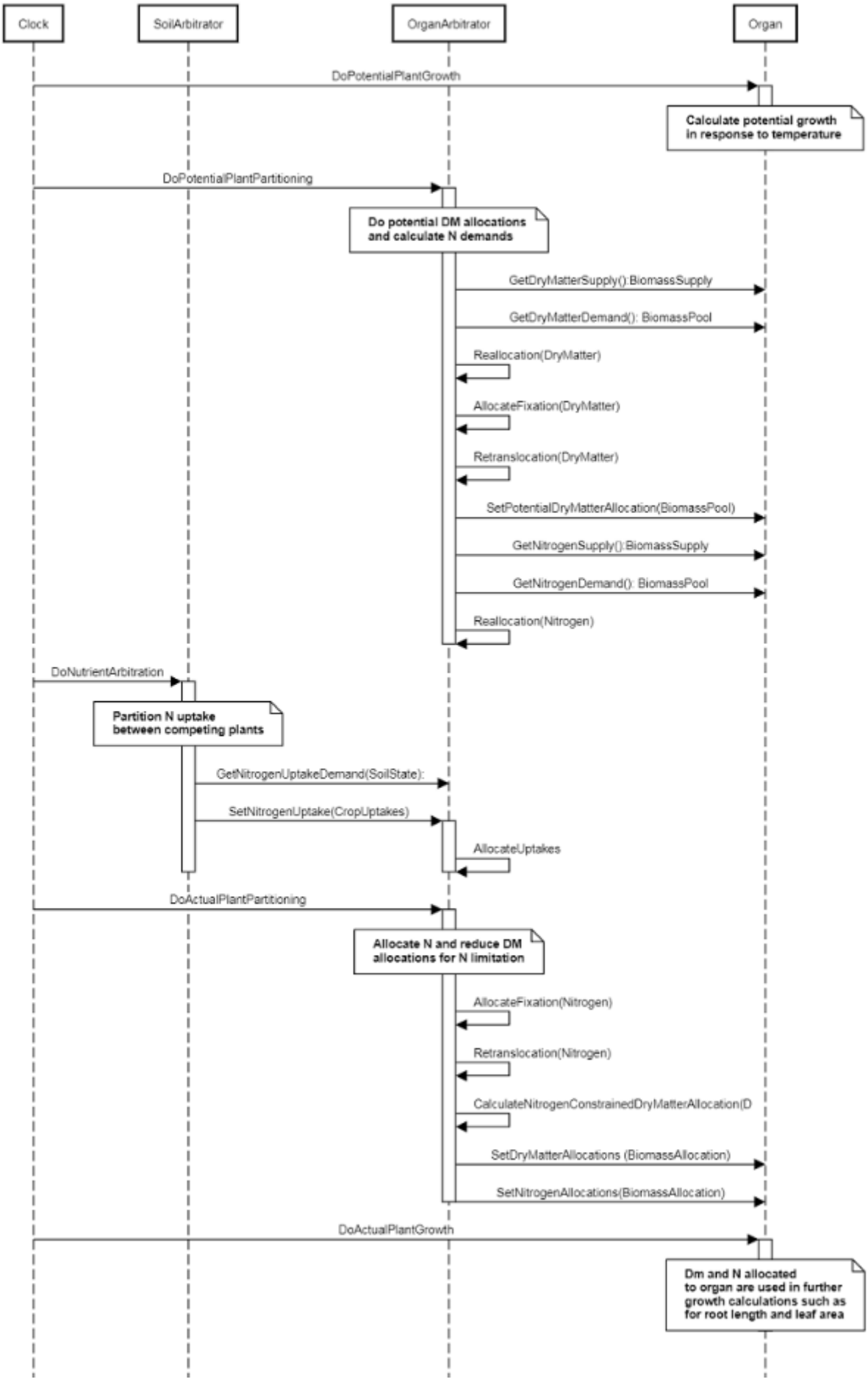


Figure 1: Schematic showing the procedure for arbitration of biomass partitioning. Pink boxes represent events that occur every day and their numbering shows the order of calculations. Blue boxes represent the methods that are called when these events occur. Orange boxes contain properties that make up the organ/arbitrator interface. Green boxes are organ specific properties.

1.6 Root

The root model calculates root growth in terms of rooting depth, biomass accumulation and subsequent root length density in each soil layer.

1.6.1 Growth

Roots grow downwards through the soil profile, with initial depth determined by sowing depth and the growth rate determined by RootFrontVelocity. The RootFrontVelocity is modified by multiplying it by the soil's XF value, which represents any resistance posed by the soil to root extension.

Root Depth Increase = RootFrontVelocity x XF_i x RootDepthStressFactor

where i is the index of the soil layer at the rooting front.

Root depth is also constrained by a maximum root depth.

Root length growth is calculated using the daily DM partitioned to roots and a specific root length. Root proliferation in layers is calculated using an approach similar to the generalised equimarginal criterion used in economics. The uptake of water and N per unit root length is used to partition new root material into layers of higher 'return on investment'. For example, the Root Activity for water is calculated as

RAw_i = -$WaterUptake_i$ / $LiveRootWt_i$ x $LayerThickness_i$ x ProportionThroughLayer

The amount of root mass partitioned to a layer is then proportional to root activity

$DMAllocated_i$ = TotalDMAllocated x RAw_i / TotalRAw

1.6.2 Dry Matter Demands

A daily DM demand is provided to the organ arbitrator and a DM supply returned. By default, 100% of the dry matter (DM) demanded from the root is structural. The daily loss of roots is calculated using a SenescenceRate function. All senesced material is automatically detached and added to the soil FOM.

1.6.3 Nitrogen Demands

The daily structural N demand from root is the product of total DM demand and the minimum N concentration. Any N above this is considered Storage and can be used for retranslocation and/or reallocation as the respective factors are set to values other then zero.

1.6.4 Nitrogen Uptake

Potential N uptake by the root system is calculated for each soil layer (i) that the roots have extended into. In each layer potential uptake is calculated as the product of the mineral nitrogen in the layer, a factor controlling the rate of extraction (kNO3 or kNH4), the concentration of N form (ppm), and a soil moisture factor (NUptakeSWFactor) which typically decreases as the soil dries. *$NO3\ uptake = NO3_i$ x kNO3 x $NO3_{ppm,\ i}$ x NUptakeSWFactor* _NH4 uptake = $NH4_i$ x kNH4 x $NH4_{ppm,\ i}$ x NUptakeSWFactor_As can be seen from the above equations, the values of kNO3 and kNH4 equate to the potential fraction of each mineral N pool which can be taken up per day for wet soil when that pool has a concentration of 1 ppm.Nitrogen uptake demand is limited to the maximum daily potential uptake (MaxDailyNUptake) and the plant's N demand. The former provides a means to constrain N uptake to a maximum value observed in the field for the crop as a whole.The demand for soil N is then passed to the soil arbitrator which determines how much of the N uptake demandeach plant instance will be allowed to take up.

1.6.5 Water Uptake

Potential water uptake by the root system is calculated for each soil layer that the roots have extended into. In each layer potential uptake is calculated as the product of the available water in the layer (water above LL limit) and a factor controlling the rate of extraction (KL). The values of both LL and KL are set in the soil interface and KL may be further modified by the crop via the KLModifier function. *$SW\ uptake = (SW_i - LL_i) \times KL_i \times KLModifier$*

1.6.6 Constants

MaxDailyNUptake = 1 (g/m2)

DMConversionEfficiency = 1 (g/m2)

RemobilisationCost = 0

MaximumRootDepth = 1000 (mm)

NitrogenDemandSwitch = 1 (0-1)

MaximumNConc = 0.01

MinimumNConc = 0.01

SenescenceRate = 0 (/d)

SpecificRootLength = 105 (m/g)

CarbonConcentration = 0.4

MaintenanceRespirationFunction = 0

RootDepthStressFactor = 1

1.6.7 RootShape

This model calculates the proportion of each soil layer occupied by roots.

1.6.8 RootFrontVelocity

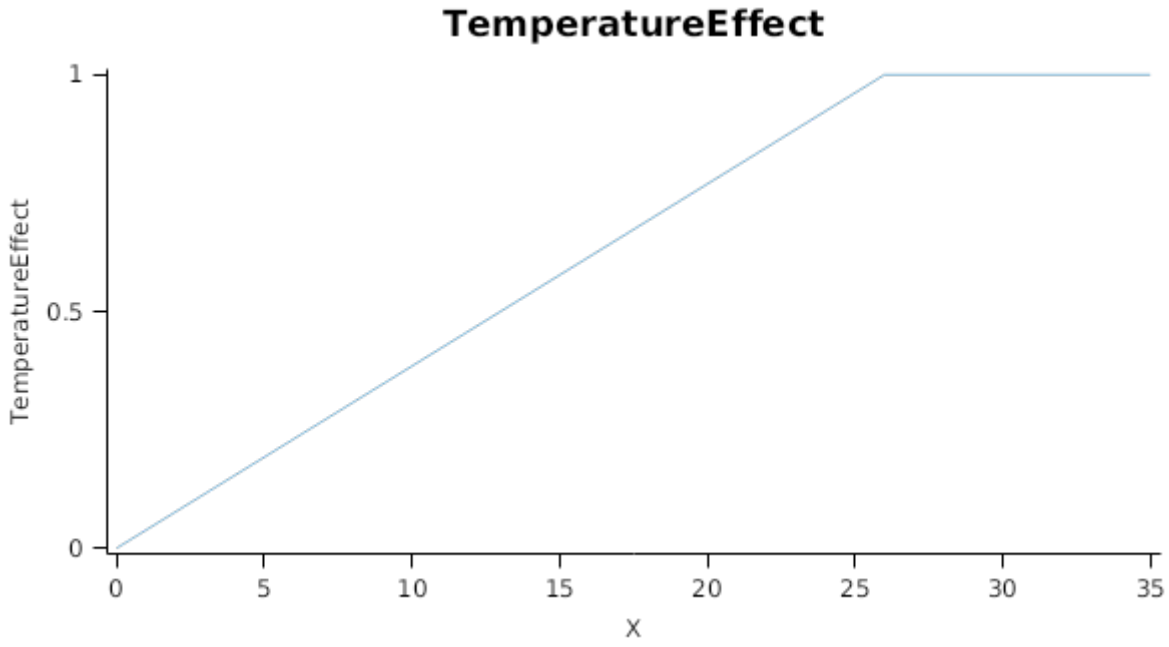
RootFrontVelocity = *TemperatureEffect* x *Potential* x *DMConversionEfficiency*

TemperatureEffect is the average of sub-daily values from a XYPairs.

Firstly 3-hourly estimates of air temperature (Ta) are interpolated using the method of Jones et al., 1986 which assumes a sinusoidal temperature. pattern between Tmax and Tmin.

Each of the interpolated air temperatures are then passed into the following Response and the Average taken to give daily TemperatureEffect

X	TemperatureEffect
0.0	0.0
26.0	1.0
35.0	1.0



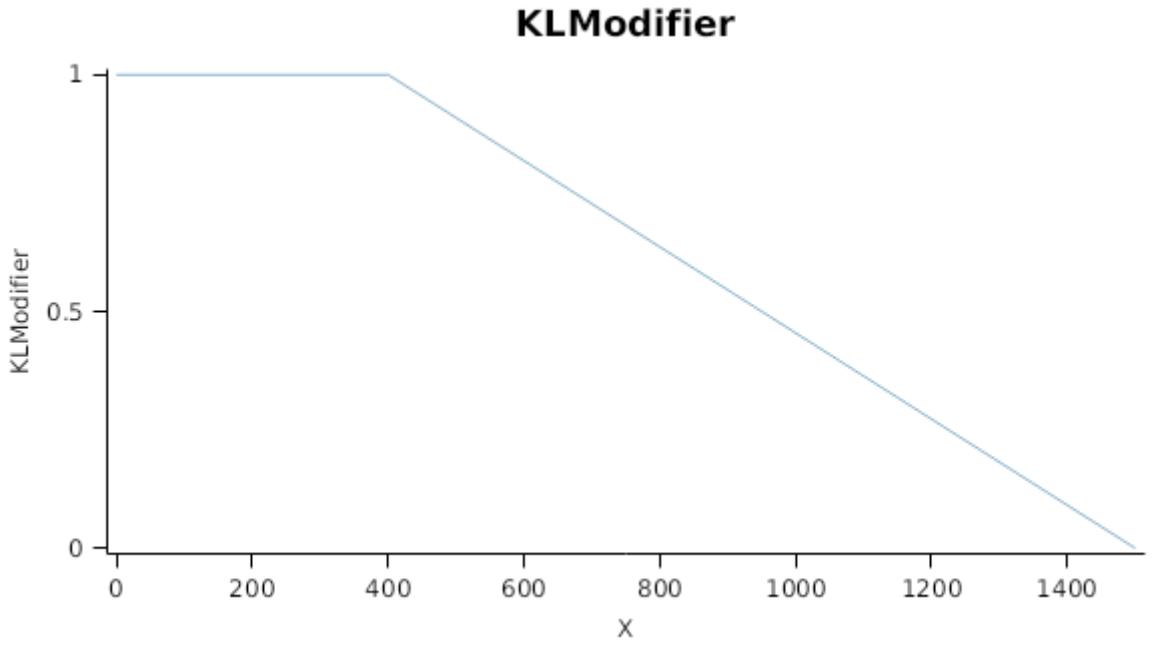
Potential = 0 (mm/d)

DMConversionEfficiency = 1

1.6.9 KLModifier

KLModifier is calculated using linear interpolation

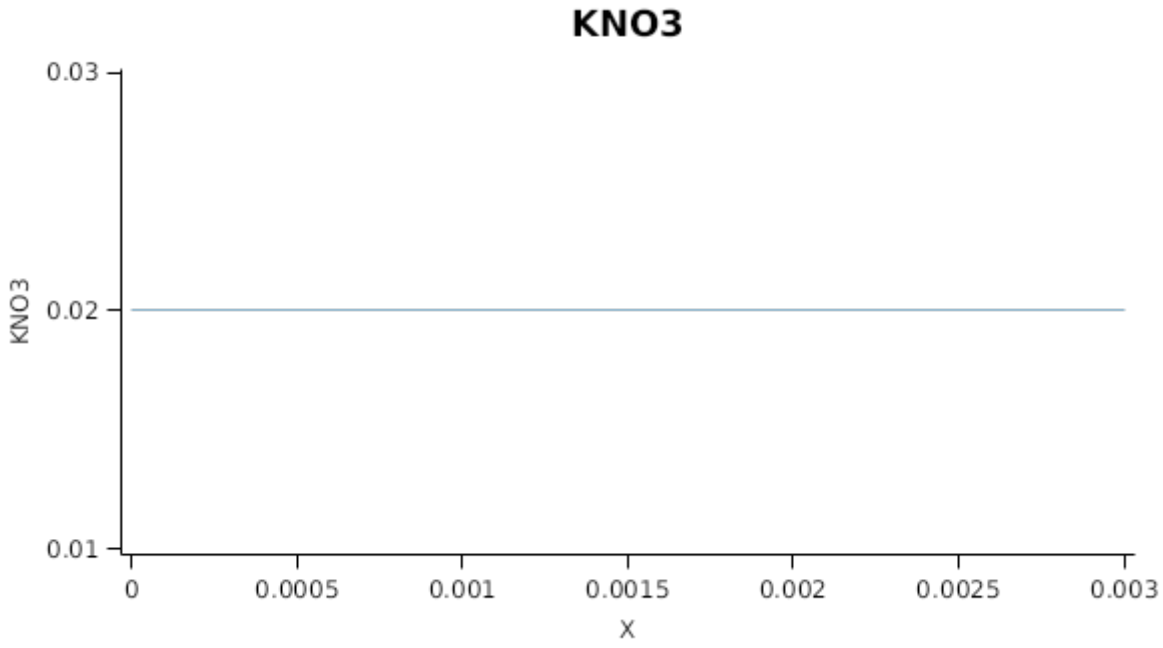
X	KLModifier
0.0	1.0
400.0	1.0
1500.0	0.0



1.6.10 KNO3

KNO3 is calculated using linear interpolation

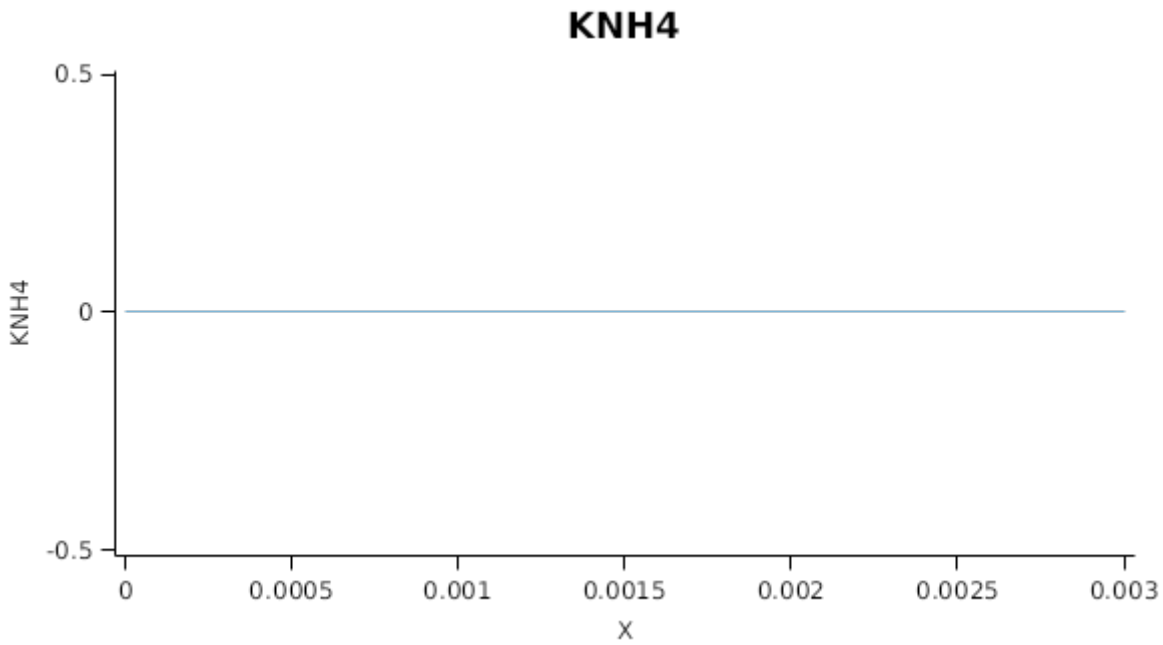
X	KNO3
0.0	0.0
0.0	0.0



1.6.11 KNH4

KNH4 is calculated using linear interpolation

X	KNH4
0.0	0.0
0.0	0.0



1.6.12 BiomassRemovalDefaults

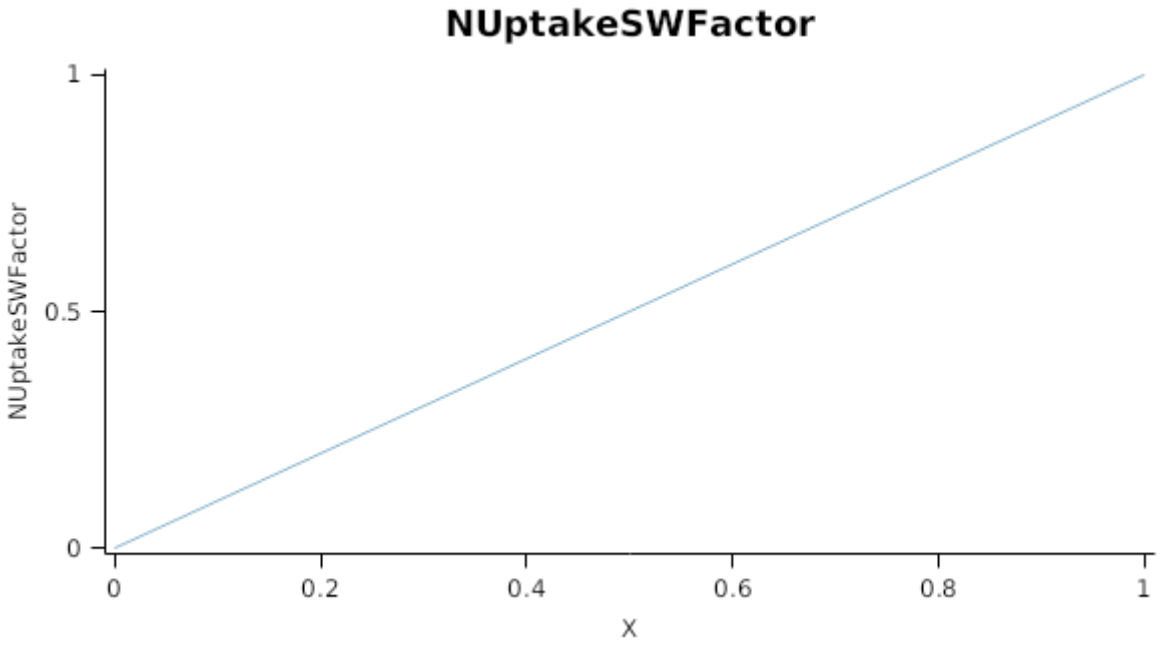
This organ will respond to certain management actions by either removing some of its biomass from the system or transferring some of its biomass to the soil surface residues. The following table describes the default proportions of live and dead biomass that are transferred out of the simulation using "Removed" or to soil surface residue using "To Residue" for a range of management actions. The total percentage removed for live or dead must not exceed 100%. The difference between the total and 100% gives the biomass remaining on the plant. These can be changed during a simulation using a manager script.

Method	% Live Removed	% Dead Removed	% Live To Residue	% Dead To Residue
Harvest	0	0	0	0
Cut	0	0	0	0
Prune	0	0	0	0
Graze	0	0	0	0

1.6.13 NUptakeSWFactor

NUptakeSWFactor is calculated using linear interpolation

X	NUptakeSWFactor
0.0	0.0
1.0	1.0



1.6.14 DMDemands

1.6.14.1 DMDemands

This class holds the functions for calculating the absolute demands and priorities for each biomass fraction.

Structural = *DMDemandFunction* x *StructuralFraction*

Returns the product of its PartitionFraction and the total DM supplied to the arbitrator by all organs.

DMDemandFunction = *PartitionFraction x [Arbitrator].DM.TotalFixationSupply*

PartitionFraction = 0 (0-1)

StructuralFraction = 1

Metabolic = 0

The partitioning of daily growth to storage biomass is based on a storage fraction.

StorageFraction = 1 - [Root].DMDemands.Structural.StructuralFraction

QStructuralPriority = 1

QMetabolicPriority = 1

QStoragePriority = 1

1.6.15 NDemands

1.6.15.1 NDemands

This class holds the functions for calculating the absolute demands and priorities for each biomass fraction.

Structural = [Root].minimumNconc x [Root].potentialDMAllocation.Structural

Metabolic = *MetabolicNconc* x [Root].potentialDMAllocation.Structural

MetabolicNconc = [Root].criticalNConc - [Root].minimumNconc

The partitioning of daily N supply to storage N attempts to bring the organ's N content to the maximum concentration.

Storage = [Root].maximumNconc × ([Root].Live.Wt + potentialAllocationWt) - [Root].Live.N

The demand for storage N is further reduced by a factor specified by the [Root].NitrogenDemandSwitch.

NitrogenDemandSwitch = [Root].nitrogenDemandSwitch

MaxNconc = [Root].maximumNconc

QStructuralPriority = 1

QMetabolicPriority = 1

QStoragePriority = 1

1.6.16 CriticalNConc

CriticalNConc = [Root].MinimumNConc

1.6.17 InitialWt

This class holds the functions for calculating the absolute demands for each biomass fraction.

Structural = 0.005 (g/plant)

Metabolic = 0

Storage = 0

1.7 Leaf

This organ is simulated using a SimpleLeaf organ type. It provides the core functions of intercepting radiation, producing biomass through photosynthesis, and determining the plant's transpiration demand. The model also calculates the growth, senescence, and detachment of leaves. SimpleLeaf does not distinguish leaf cohorts by age or position in the canopy.

Radiation interception and transpiration demand are computed by the MicroClimate model. This model takes into account competition between different plants when more than one is present in the simulation. The values of canopy Cover, LAI, and plant Height (as defined below) are passed daily by SimpleLeaf to the MicroClimate model. MicroClimate uses an implementation of the Beer-Lambert equation to compute light interception and the Penman-Monteith equation to calculate potential evapotranspiration.
These values are then given back to SimpleLeaf which uses them to calculate photosynthesis and soil water demand.

NOTE: the summary above is used in the Apsim's autodoc.

SimpleLeaf has two options to define the canopy: the user can either supply a function describing LAI or a function describing canopy cover directly. From either of these functions SimpleLeaf can obtain the other property using the Beer-Lambert equation with the specified value of extinction coefficient. The effect of growth rate on transpiration is captured by the Fractional Growth Rate (FRGR) function, which is passed to the MicroClimate model.

1.7.1 Initial Dry Matter

InitialWt = *InitialPlantWt* x [Plant].Population

InitialPlantWt = 0 (g/plant)

1.7.2 Dry Matter Demand

The dry matter demand for the organ is calculated as defined in DMDemands, based on the DMDemandFunction and partition fractions for each biomass pool.

1.7.2.1 DMDemands

This class holds the functions for calculating the absolute demands and priorities for each biomass fraction.

Structural = *DMDemandFunction* x *StructuralFraction*

Returns the product of its PartitionFraction and the total DM supplied to the arbitrator by all organs.

DMDemandFunction = PartitionFraction x [Arbitrator].DM.TotalFixationSupply

PartitionFraction = 1 (0-1)

StructuralFraction = 0.5 (0-1)

Metabolic = 0

The partitioning of daily growth to storage biomass is based on a storage fraction.

StorageFraction = 1 - [Leaf].DMDemands.Structural.StructuralFraction

QStructuralPriority = 1

QMetabolicPriority = 1

QStoragePriority = 1

1.7.3 Nitrogen Demand

The N demand is calculated as defined in NDemands, based on DM demand the N concentration of each biomass pool.

1.7.3.1 NDemands

This class holds the functions for calculating the absolute demands and priorities for each biomass fraction.

Structural = [Leaf].minimumNconc x [Leaf].potentialDMAllocation.Structural

Metabolic = *MetabolicNconc* x [Leaf].potentialDMAllocation.Structural

MetabolicNconc = [Leaf].criticalNConc - [Leaf].minimumNconc

The partitioning of daily N supply to storage N attempts to bring the organ's N content to the maximum concentration.

Storage = [Leaf].maximumNconc × ([Leaf].Live.Wt + potentialAllocationWt) - [Leaf].Live.N

The demand for storage N is further reduced by a factor specified by the [Leaf].NitrogenDemandSwitch.

NitrogenDemandSwitch = [Leaf].nitrogenDemandSwitch

MaxNconc = [Leaf].maximumNconc

QStructuralPriority = 1

QMetabolicPriority = 1

QStoragePriority = 1

1.7.4 Nitrogen Concentration Thresholds

The N demand is calculated as defined in NDemands, based on DM demand the N concentration of each biomass pool.

1.7.4.1 NDemands

This class holds the functions for calculating the absolute demands and priorities for each biomass fraction.

Structural = [Leaf].minimumNconc x [Leaf].potentialDMAllocation.Structural

Metabolic = *MetabolicNconc* x [Leaf].potentialDMAllocation.Structural

MetabolicNconc = [Leaf].criticalNConc - [Leaf].minimumNconc

The partitioning of daily N supply to storage N attempts to bring the organ's N content to the maximum concentration.

Storage = [Leaf].maximumNconc × ([Leaf].Live.Wt + potentialAllocationWt) - [Leaf].Live.N

The demand for storage N is further reduced by a factor specified by the [Leaf].NitrogenDemandSwitch.

NitrogenDemandSwitch = [Leaf].nitrogenDemandSwitch

MaxNconc = [Leaf].maximumNconc

QStructuralPriority = 1

QMetabolicPriority = 1

QStoragePriority = 1

1.7.5 Dry Matter Supply

DMReallocationFactor = 0

DMRetranslocationFactor = 0

1.7.6 Photosynthesis

Photosynthesis = 5 (g/m2/d)

1.7.7 Nitrogen Supply

NReallocationFactor = 0

NRetranslocationFactor = 0

1.7.8 Canopy Properties

Leaf has been defined with a LAIFunction, cover is calculated using the Beer-Lambert equation.

Area = 5 (m2/m2)

ExtinctionCoefficient = 0.7

Tallness = 400 (mm)

1.7.9 StomatalConductance

Stomatal Conductance (gs) is calculated for use within the micromet model by adjusting a value provided for an atmospheric CO2 concentration of 350 ppm. The impact of other stresses (e.g. Temperature, N) are captured through the modifier, Frgr.

gs = Gsmax350 x FRGR x stomatalConductanceCO2Modifier

StomatalConductanceCO2Modifier = 1

1.7.10 Senescence and Detachment

Leaf has senescence parameterised to zero so all biomass in this organ will remain alive.

Leaf has detachment parameterised to zero so all biomass in this organ will remain with the plant until a defoliation or harvest event occurs.

1.7.11 Biomass removal

This organ will respond to certain management actions by either removing some of its biomass from the system or transferring some of its biomass to the soil surface residues. The following table describes the default proportions of live and dead biomass that are transferred out of the simulation using "Removed" or to soil surface residue using "To Residue" for a range of management actions. The total percentage removed for live or dead must not exceed 100%. The difference between the total and 100% gives the biomass remaining on the plant. These can be changed during a simulation using a manager script.

Method	% Live Removed	% Dead Removed	% Live To Residue	% Dead To Residue
Harvest	0	0	0	0
Cut	0	0	0	0
Prune	0	0	0	0
Graze	0	0	0	0

1.8 StaticTree

StaticTree overrides the following properties:

[Root].MaximumRootDepth.FixedValue = 2600

[Leaf].ExtinctionCoefficient.FixedValue = 0.65

[Leaf].Tallness.FixedValue = 5000

[Root].KLModifier.XYPairs.X = 0, 400, 500, 1000, 1500, 2000

[Root].KLModifier.XYPairs.Y = 1, 1, 0.9, 0.6, 0.3, 0.1

1.9 Pasture

Pasture overrides the following properties:

[Root].MaximumRootDepth.FixedValue = 800

[Root].RootFrontVelocity.Potential.FixedValue = 10

[Leaf].ExtinctionCoefficient.FixedValue = 0.65

[Root].KLModifier.XYPairs.X = 0, 200, 300, 500, 1000

[Root].KLModifier.XYPairs.Y = 1, 1, 0.2, 0.03, 0.0

[Leaf].Gsmax350 = 0.004

[Leaf].R50 = 175

1.10 Lucerne

Lucerne overrides the following properties:

[Root].MaximumRootDepth.FixedValue = 2500

[Root].RootFrontVelocity.Potential.FixedValue = 11

[Leaf].ExtinctionCoefficient.FixedValue = 0.8

[Root].KLModifier.XYPairs.X = 0, 200, 500, 1000, 1500, 2000

[Root].KLModifier.XYPairs.Y = 1, 1, 0.5, 0.2, 0.08, 0.03

[Leaf].Gsmax350 = 0.006

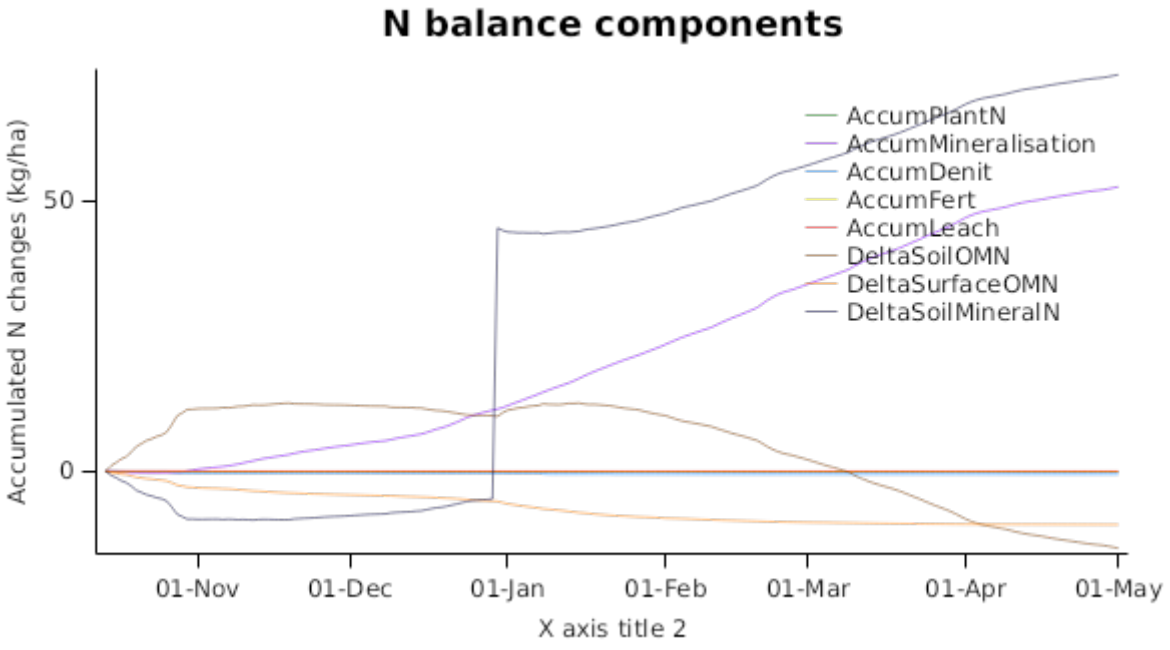
[Leaf].R50 = 150

1.11 MortalityRate

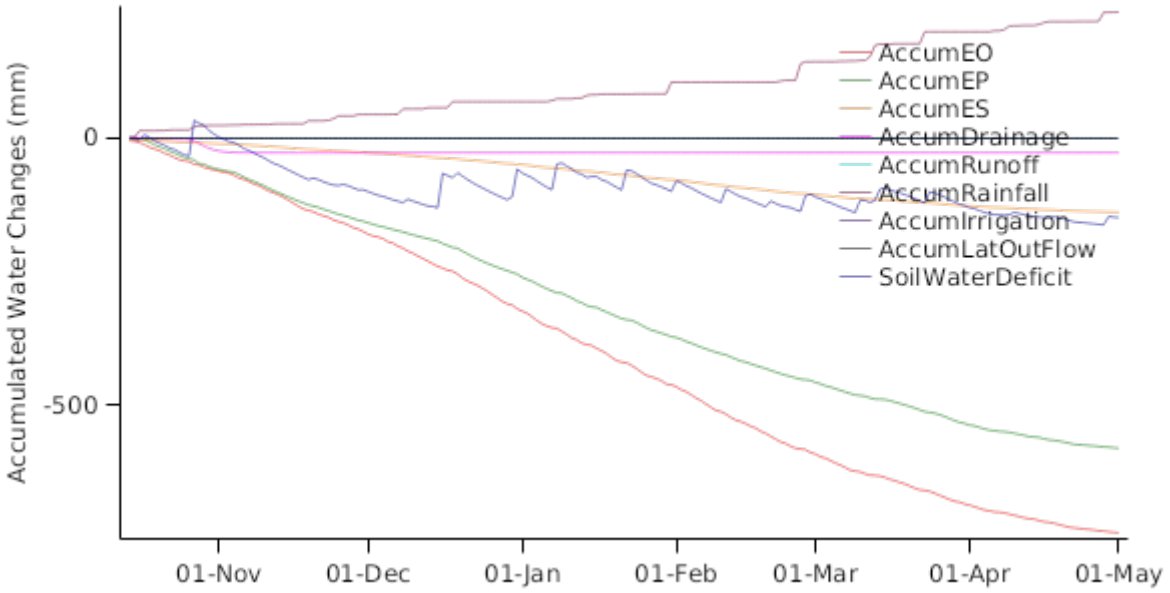
MortalityRate = 0

2 SlurpTestBase

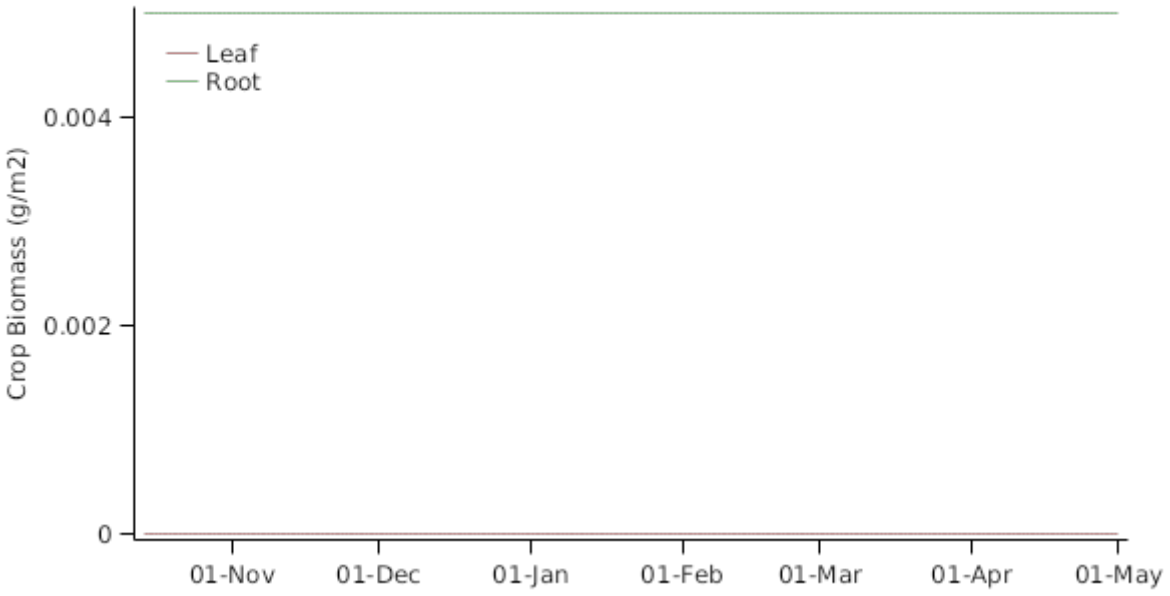
This is a simple test that plants SLURP, changes its leaf area index and potential biomass production on one occasion and presents some graphs of Plant, N and Water balance components to show things are working.



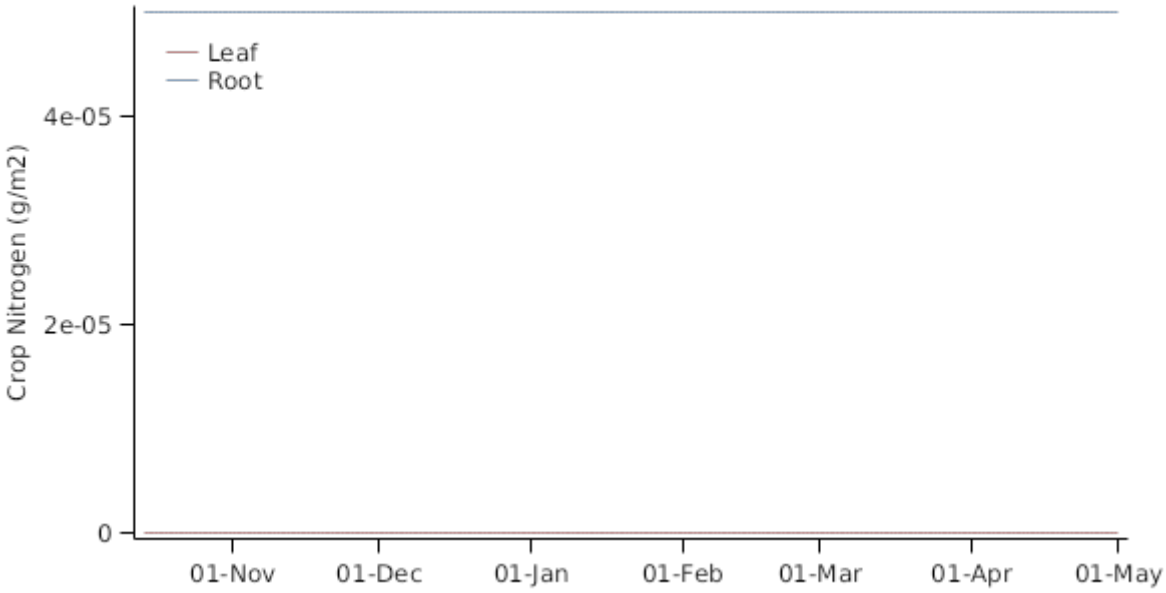
WaterBalanceComponents



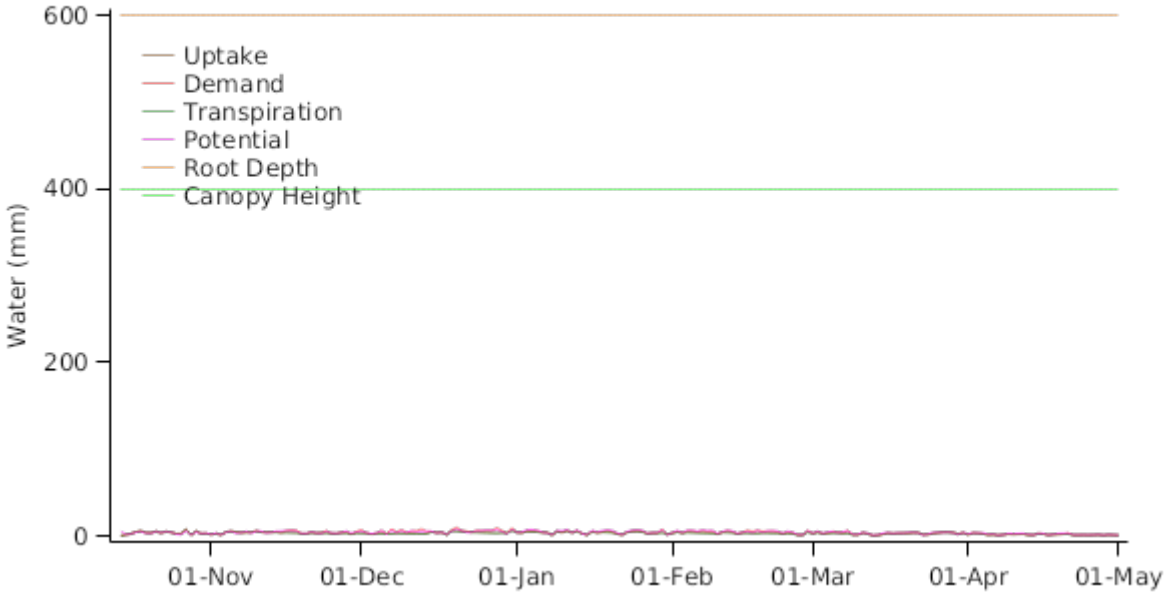
Plant Biomass components



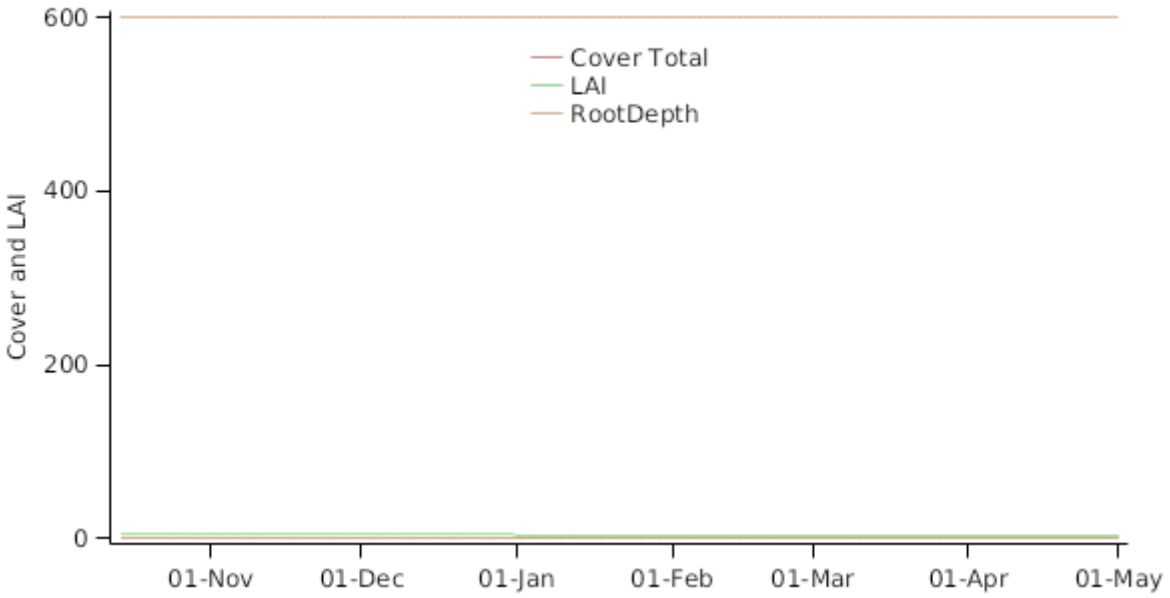
Plant Nitrogen components



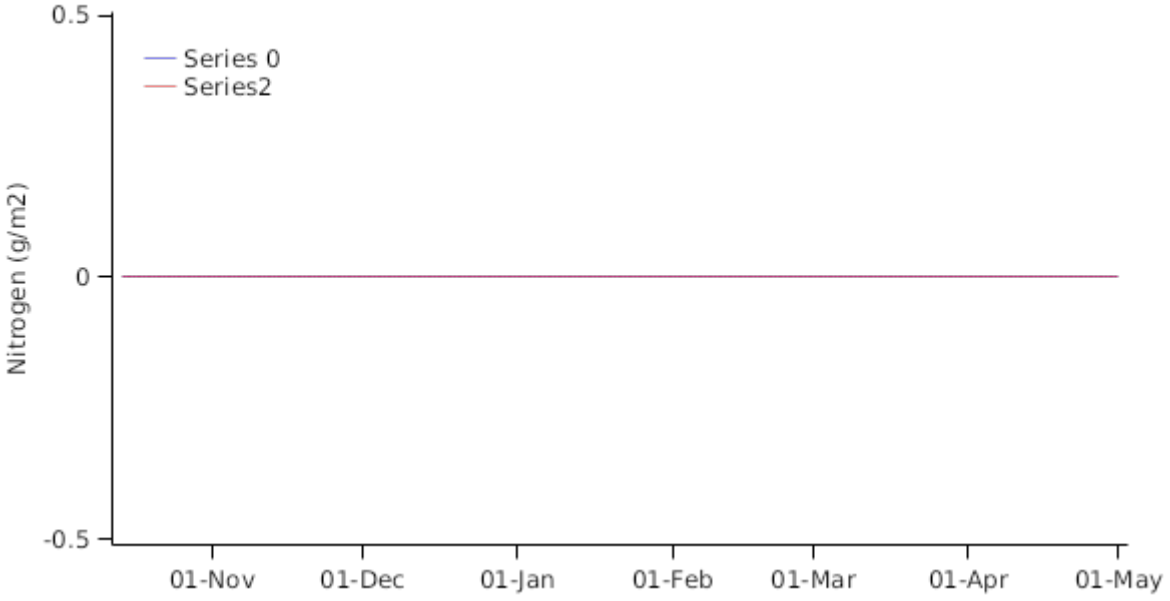
Water supply and demand

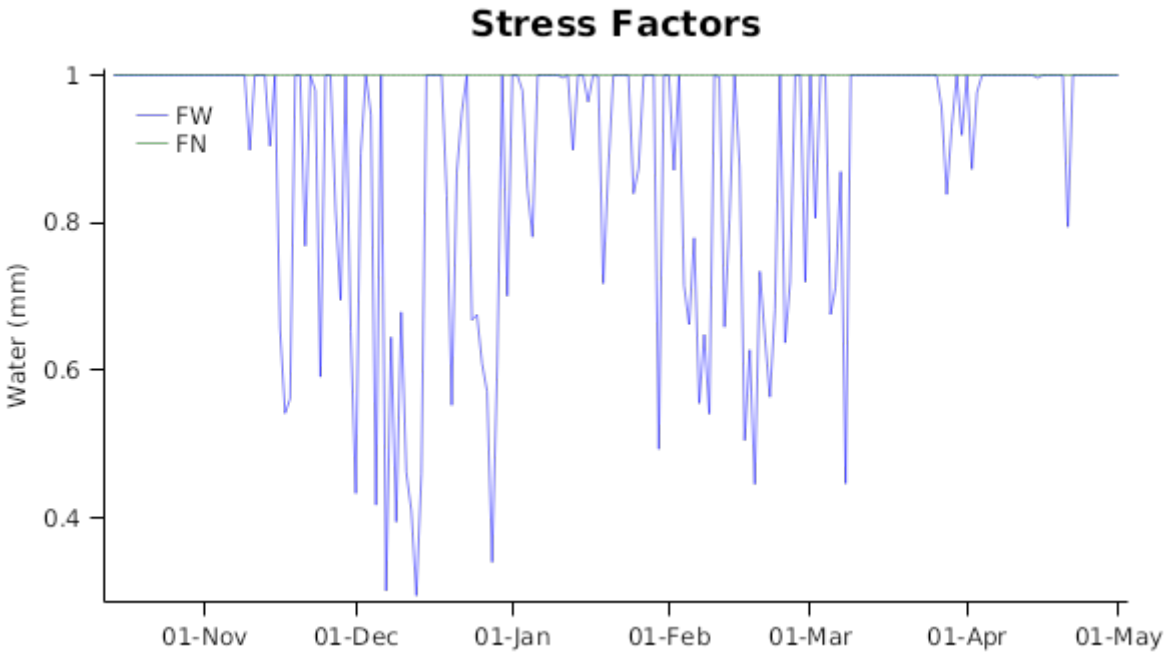


Crop structure



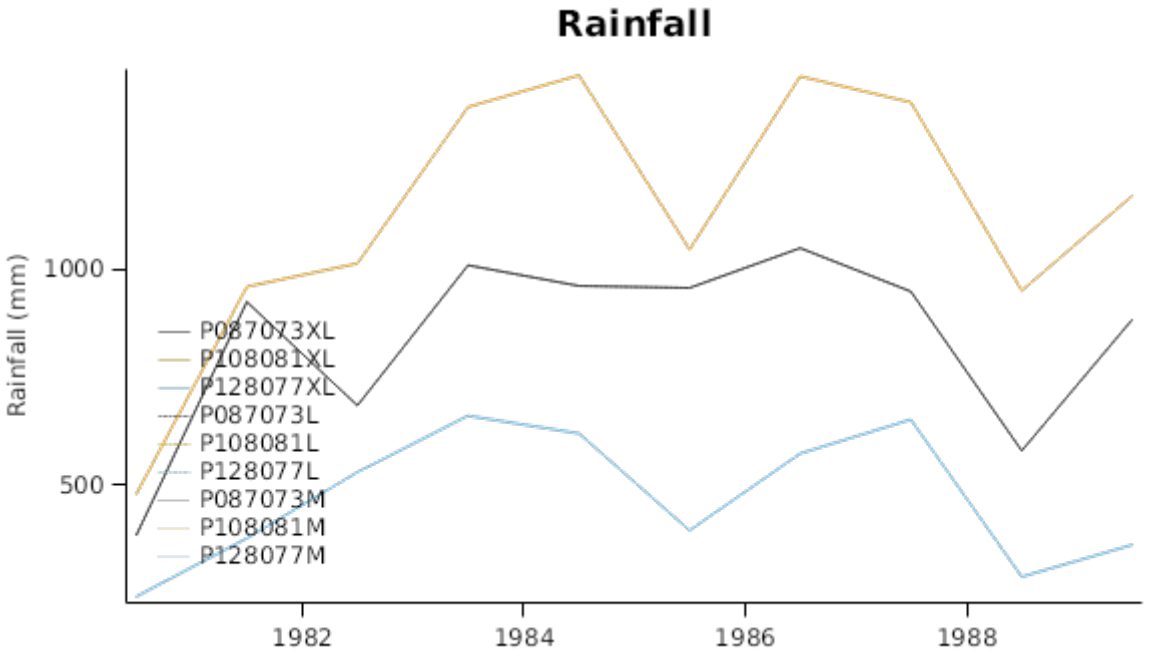
N supply and demand



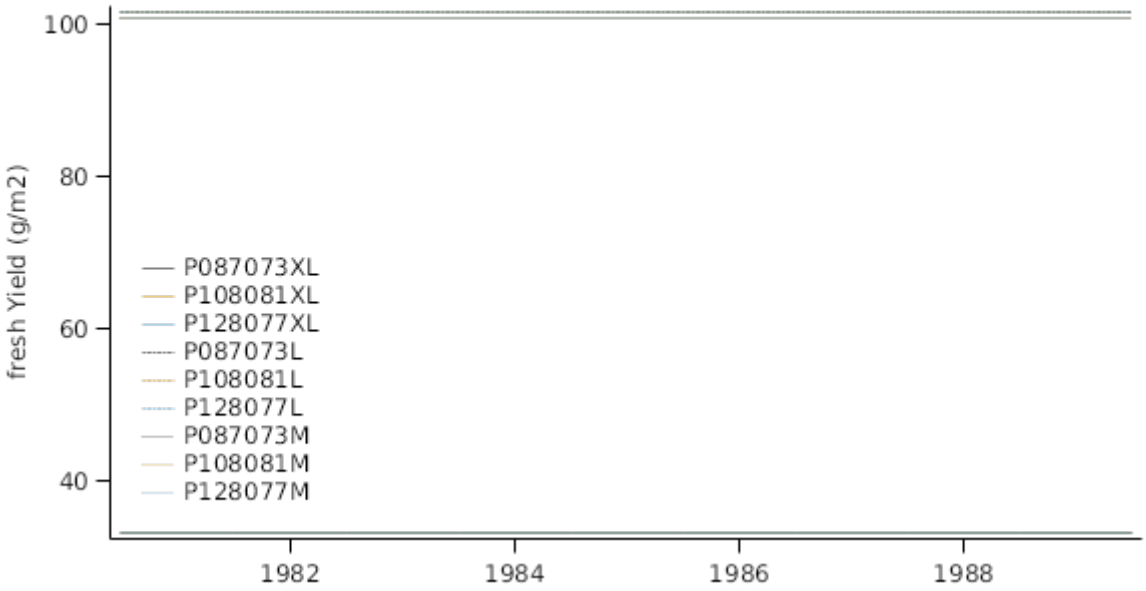


3 Tests

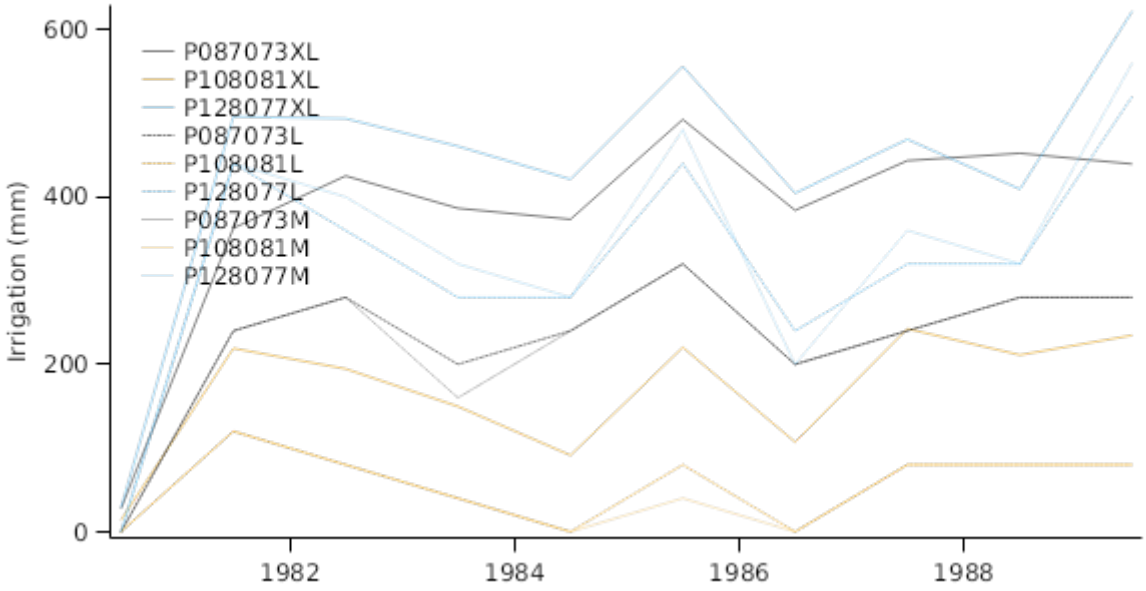
This test has combinations of different Soils and Climate with Slurp run over 10 years and an automatic irrigation rule included. This demonstrats that simulations including Slurp can generate sensible patterns in irrgation, drainage, runoff and potential water stress index



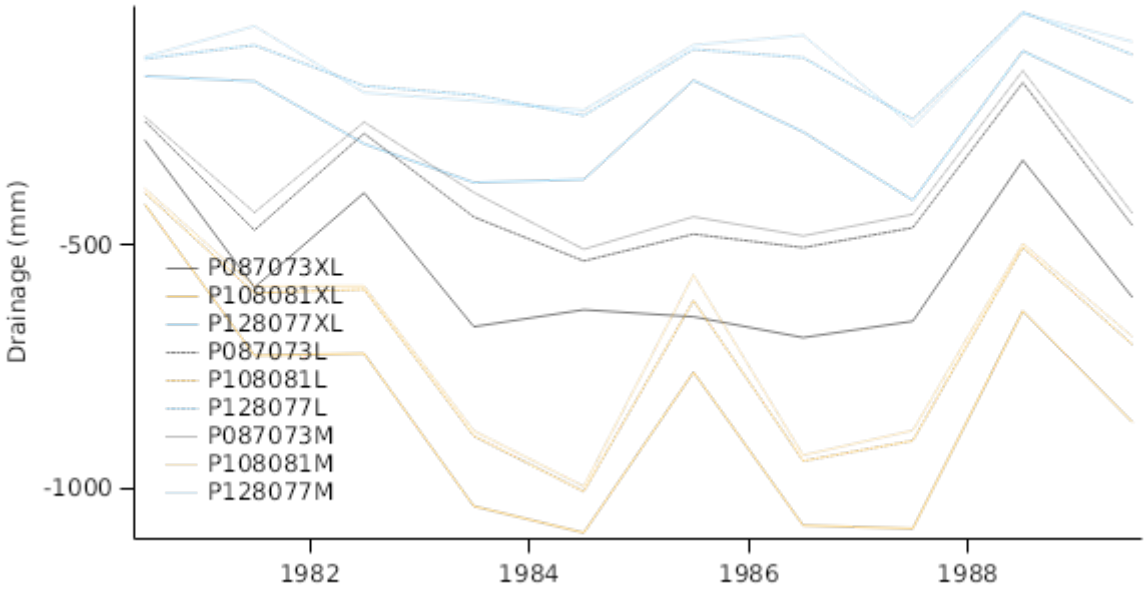
PAWC



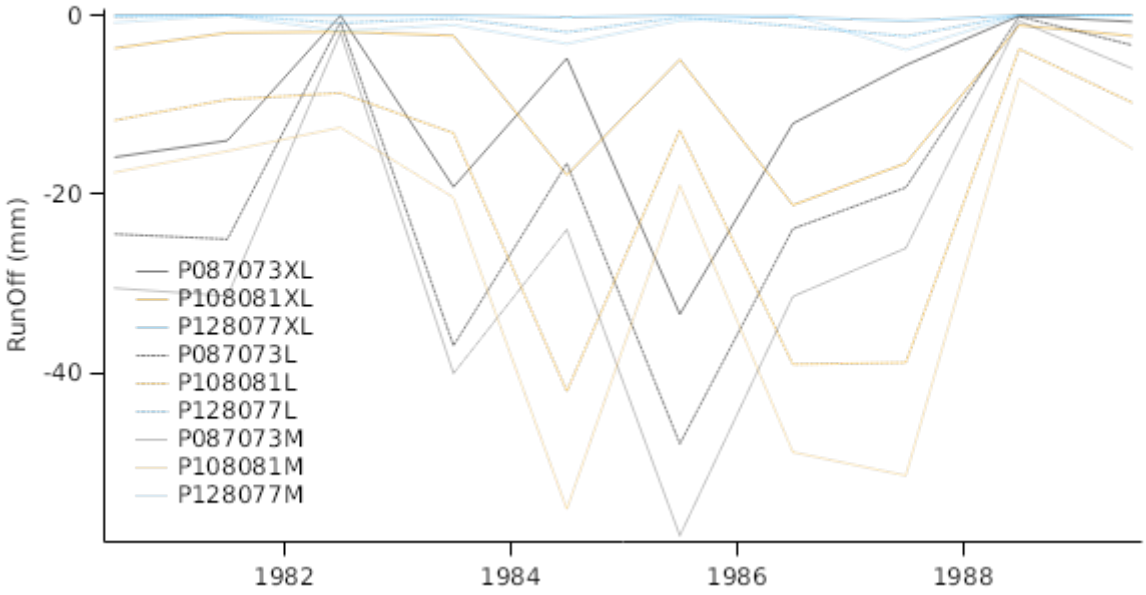
Irrigation

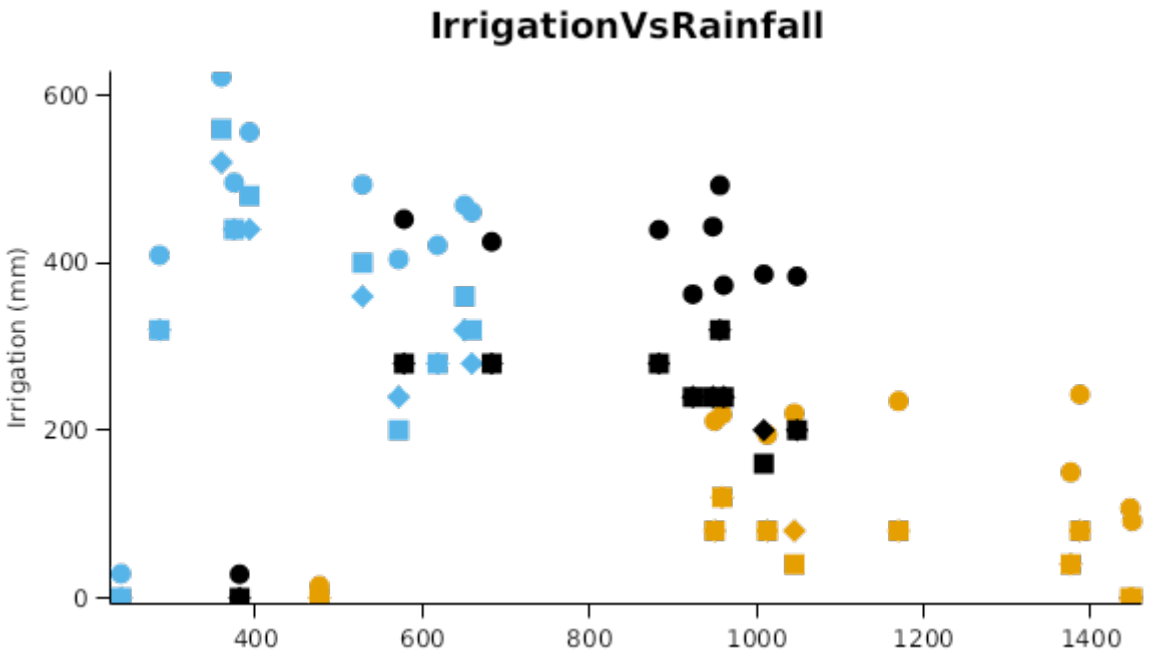


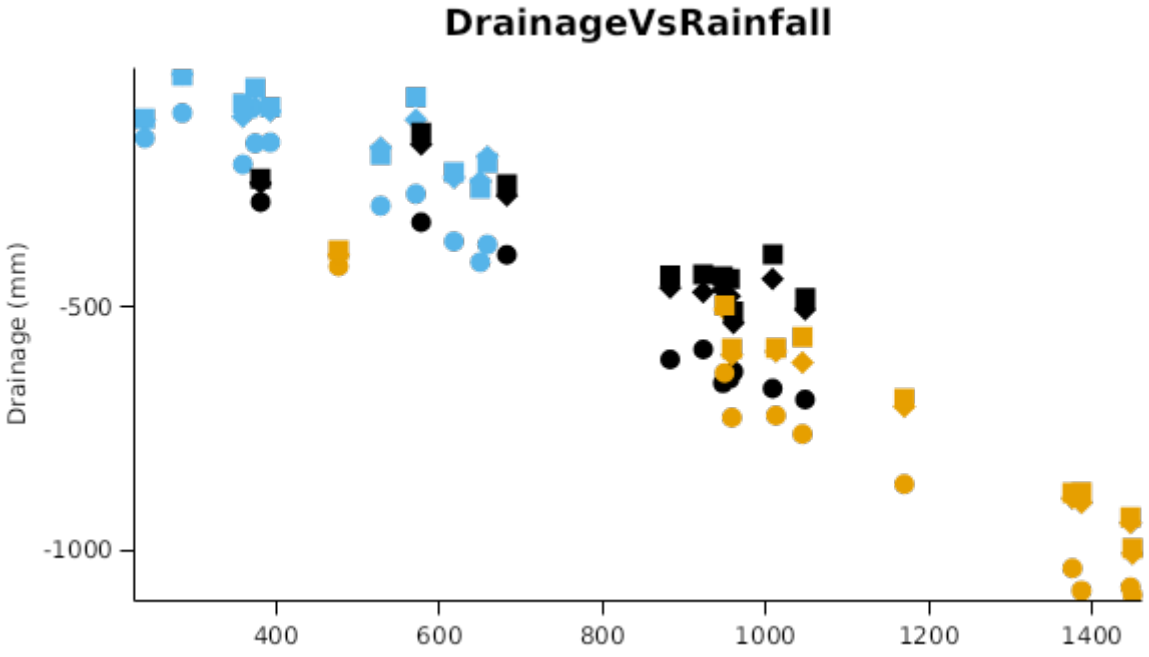
Drainage

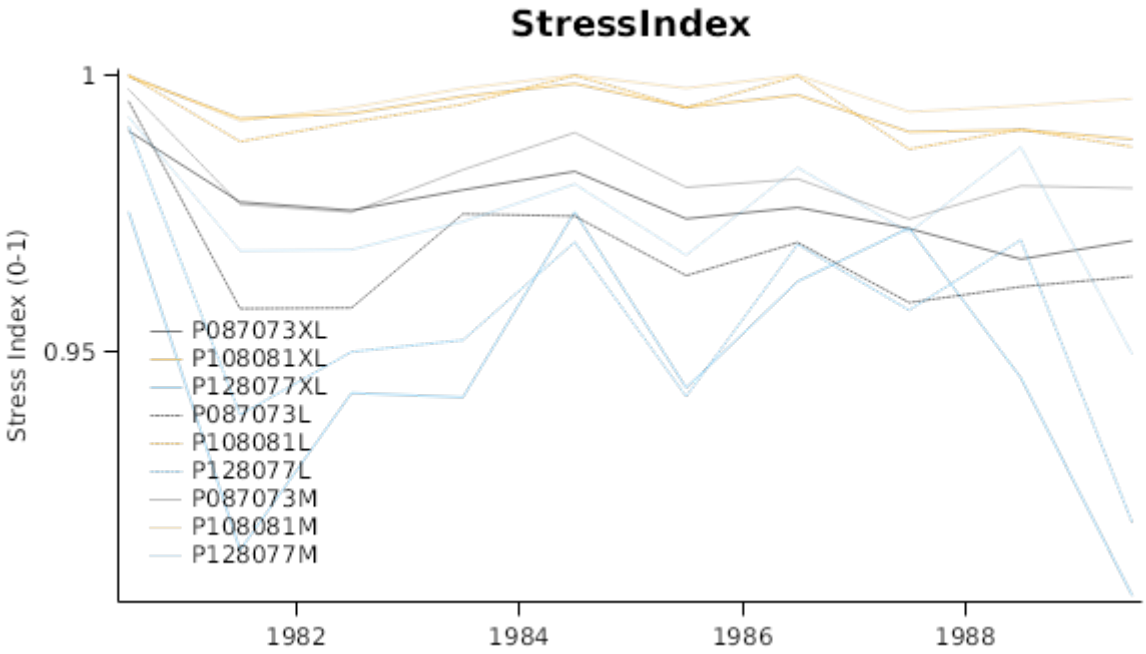


Runoff



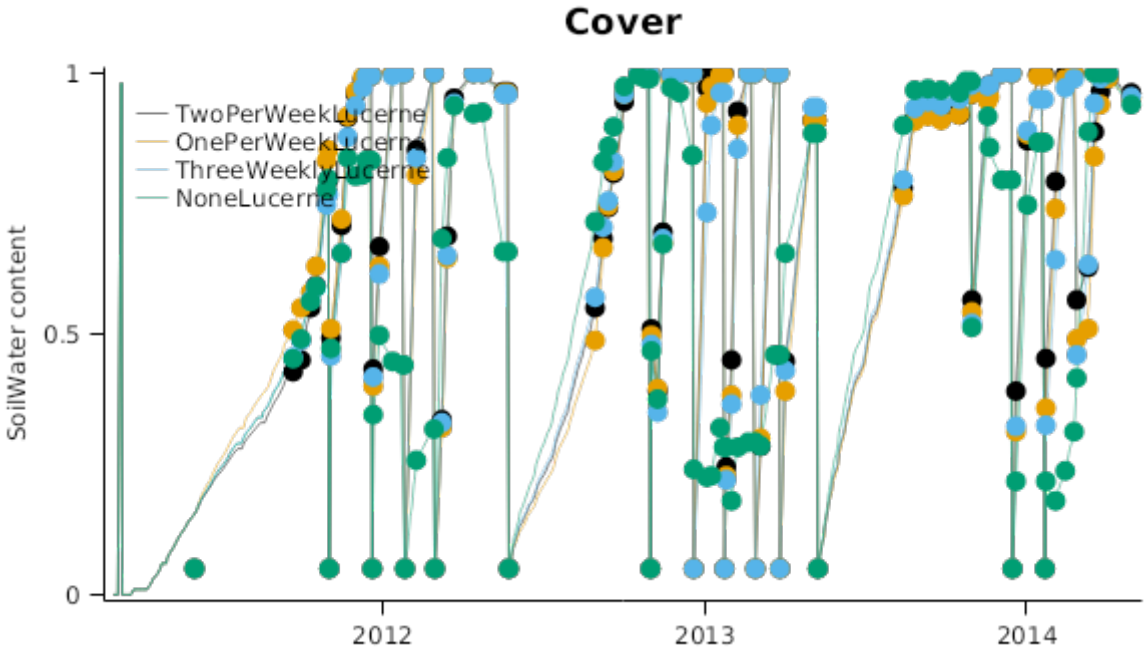

IrrigationVsRainfall
Irrigation (mm)
600
400
200
0
400
600
800
1000
1200
1400

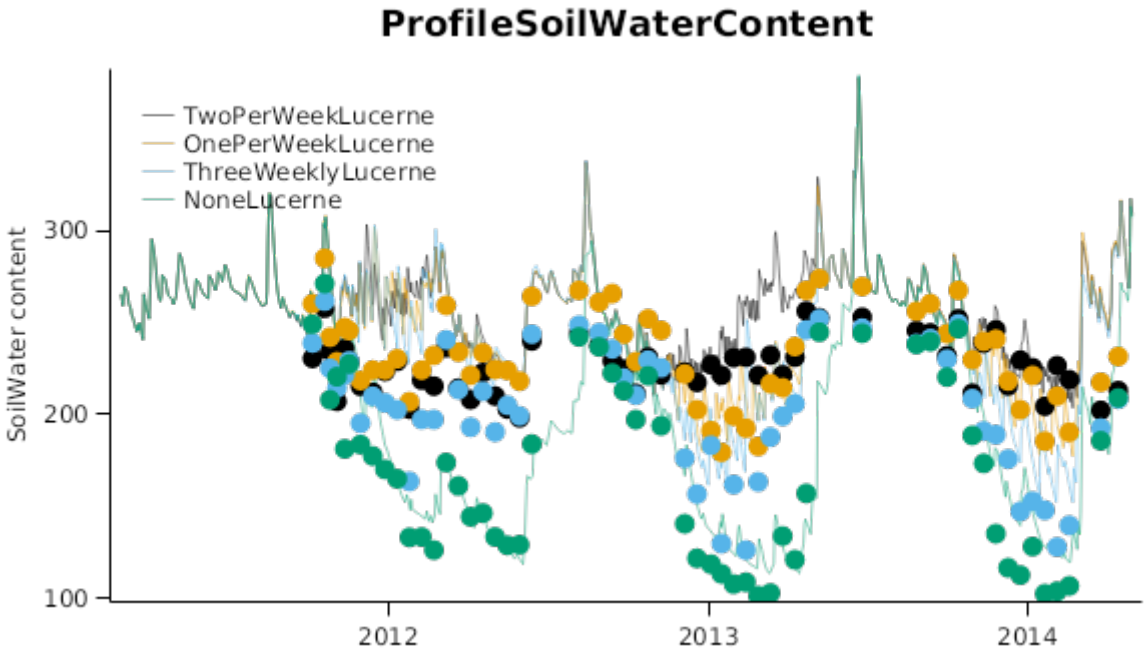

DrainageVsRainfall
Drainage (mm)
-500
-1000
400
600
800
1000
1200
1400

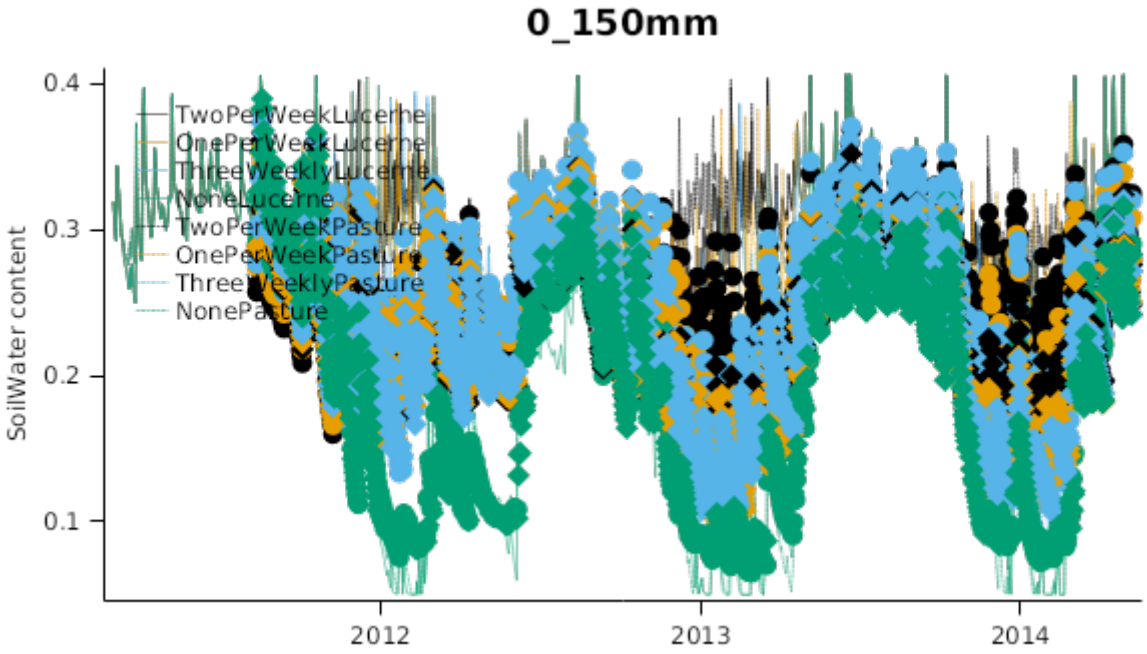


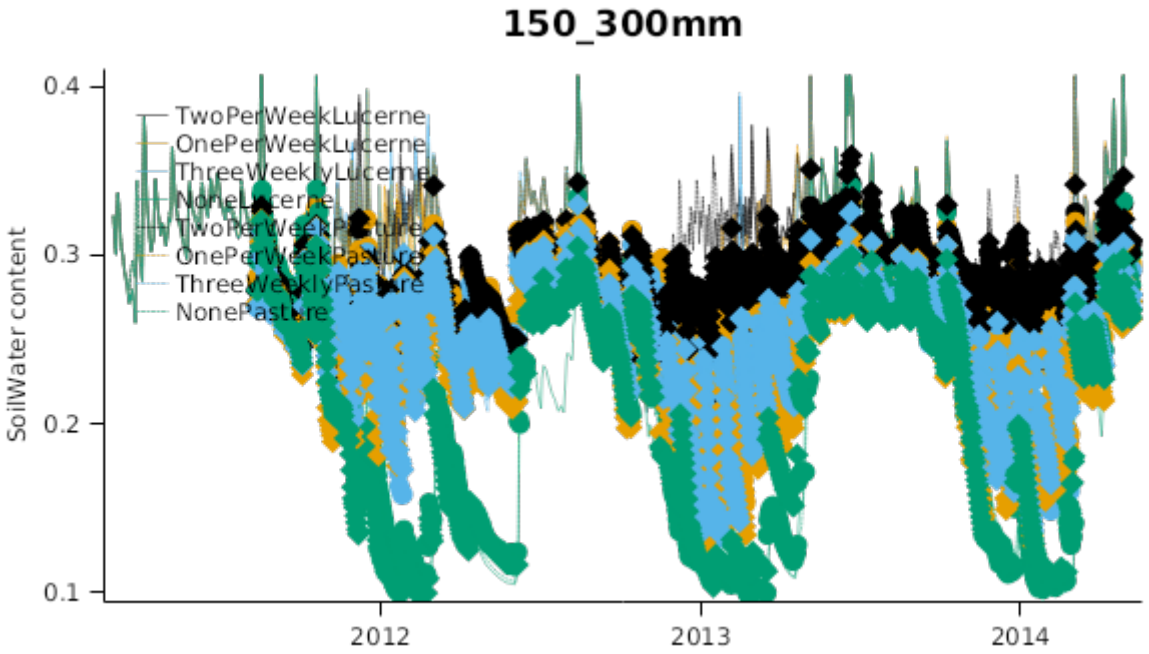
4 LandP

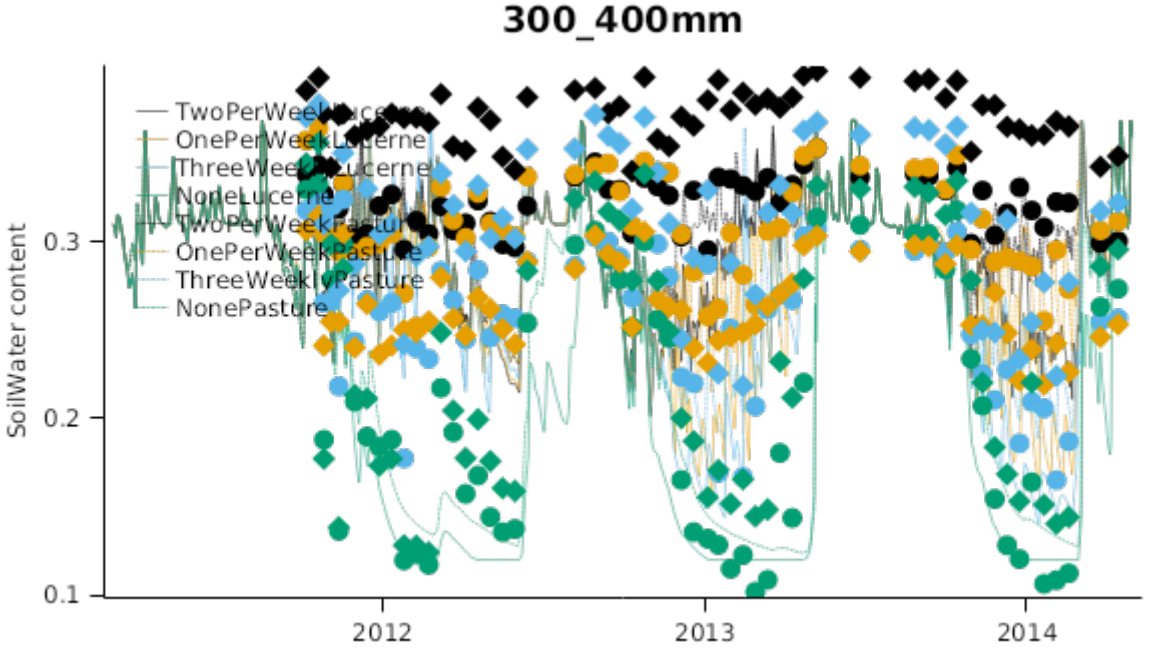
In this test we run Slurp against a dataset of Soil water content measured over 3 years under lucerne with different irrigation treatments applied. Slurp was able to reproduce the tempeoral patterns of the 4 different irrigation treatments with adequate accuracy.

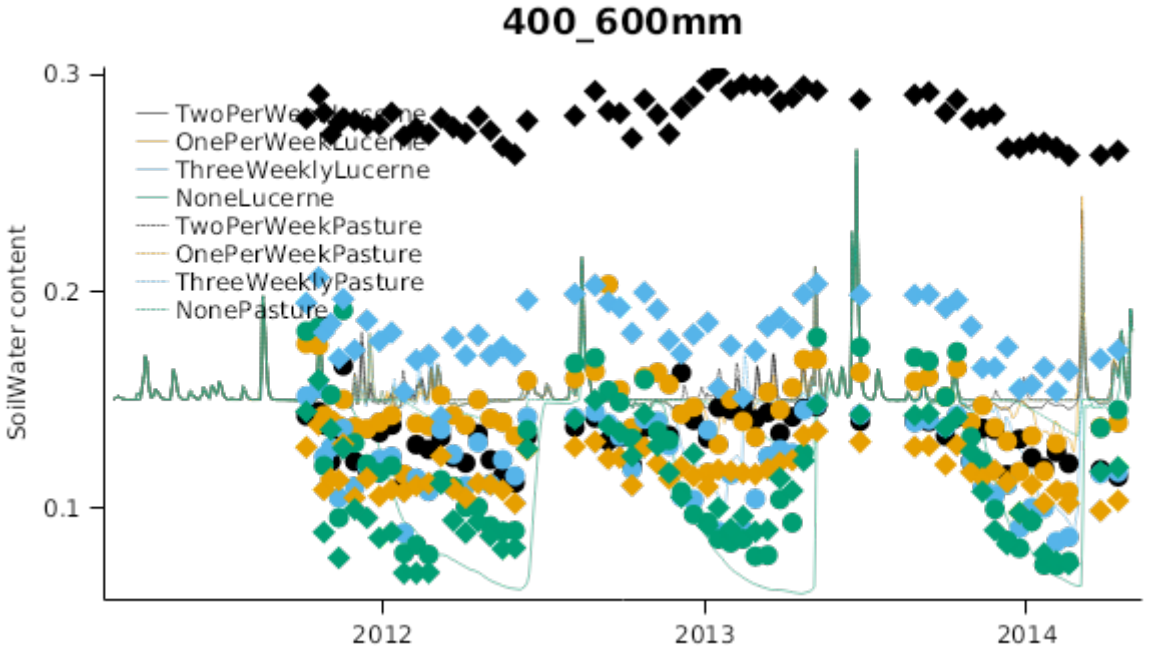


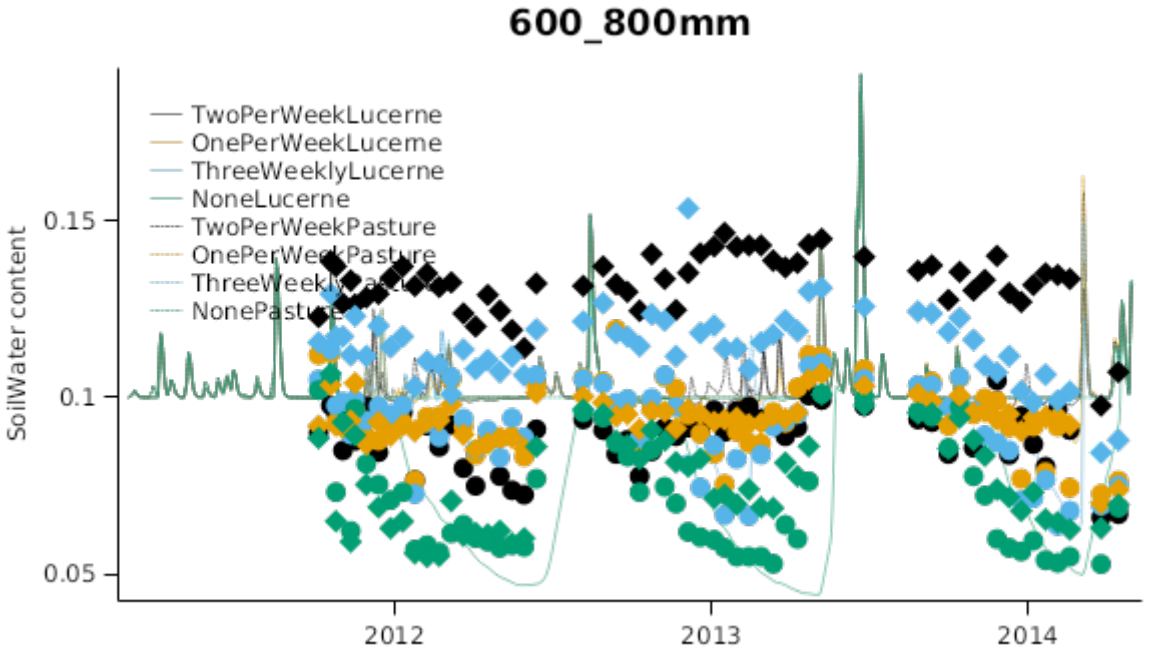

ProfileSoilWaterContent
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
SoilWater content
300
200
100
2012
2013
2014

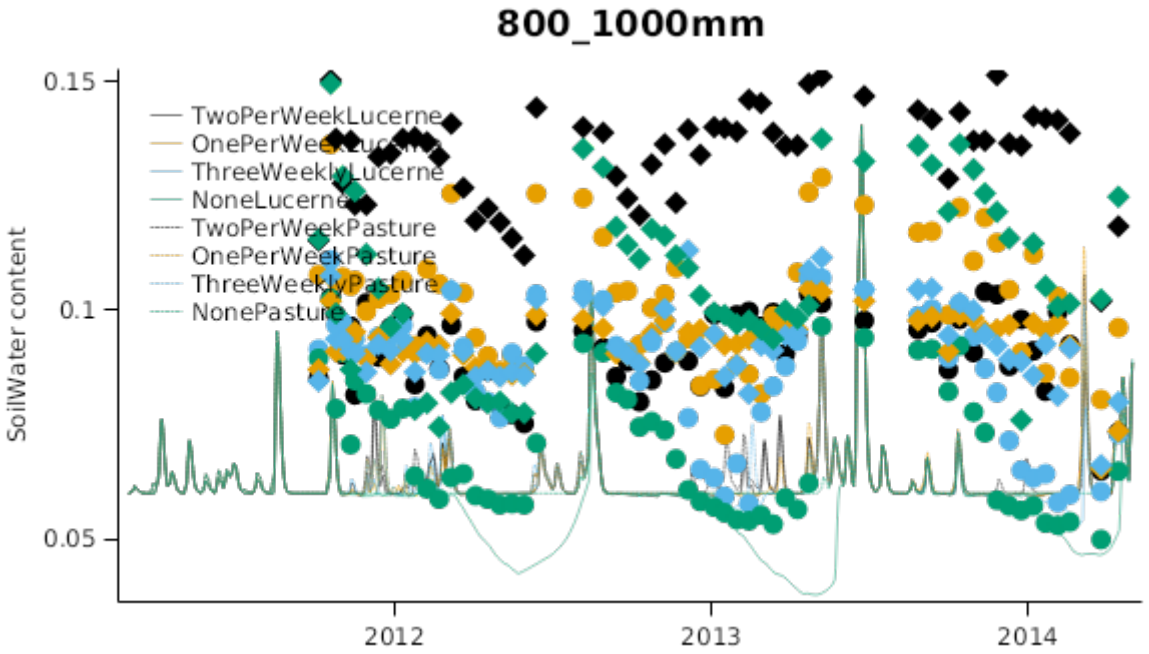

0_150mm
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture
SoilWater content
0.4
0.3
0.2
0.1
2012
2013
2014

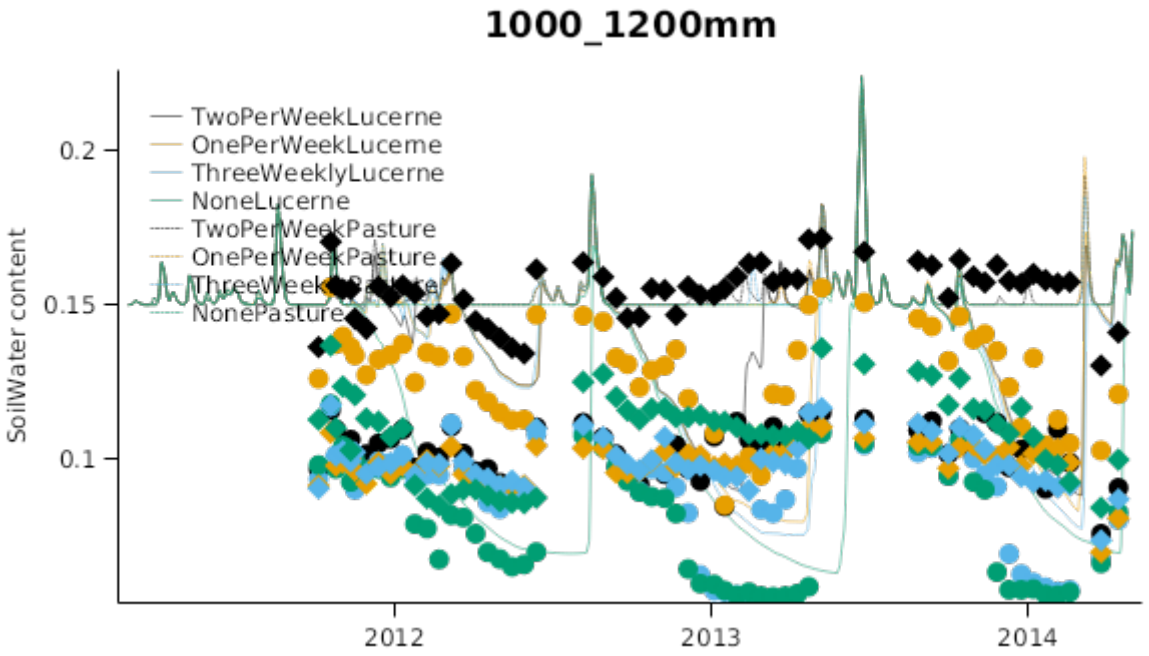

150_300mm
SoilWater content
0.4
0.3
0.2
0.1
2012
2013
2014
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture

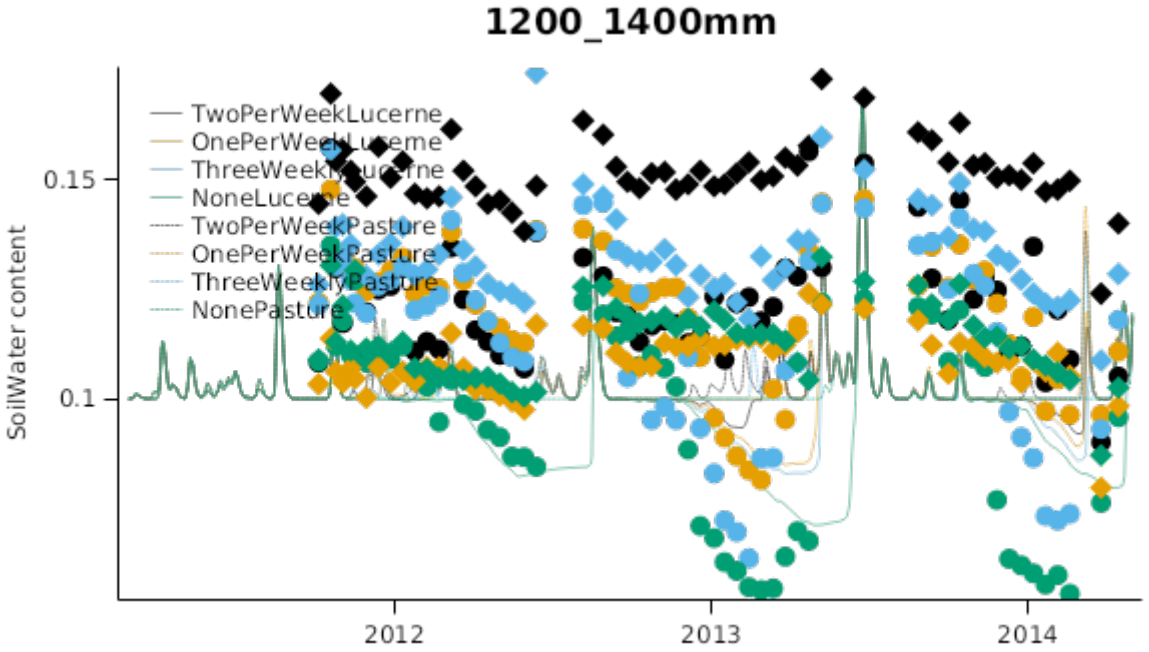

300_400mm
SoilWater content
0.3
0.2
0.1
2012
2013
2014
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture

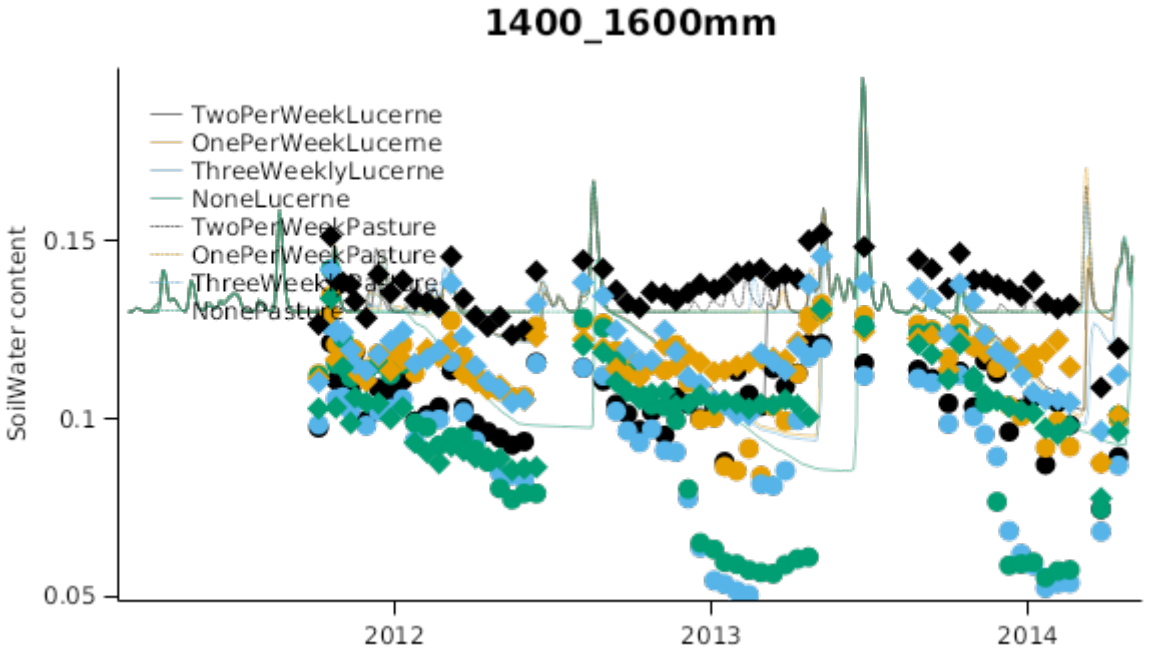

400_600mm
SoilWater content
0.3
0.2
0.1
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture
2012
2013
2014

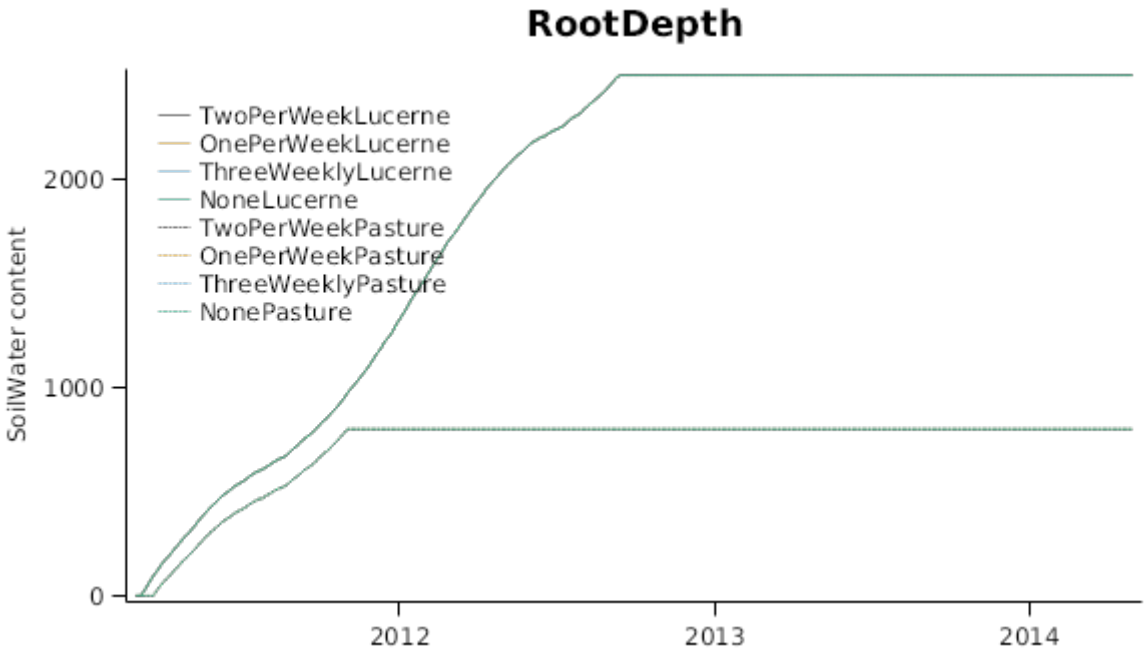

600_800mm
SoilWater content
0.15
0.1
0.05
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture
2012
2013
2014

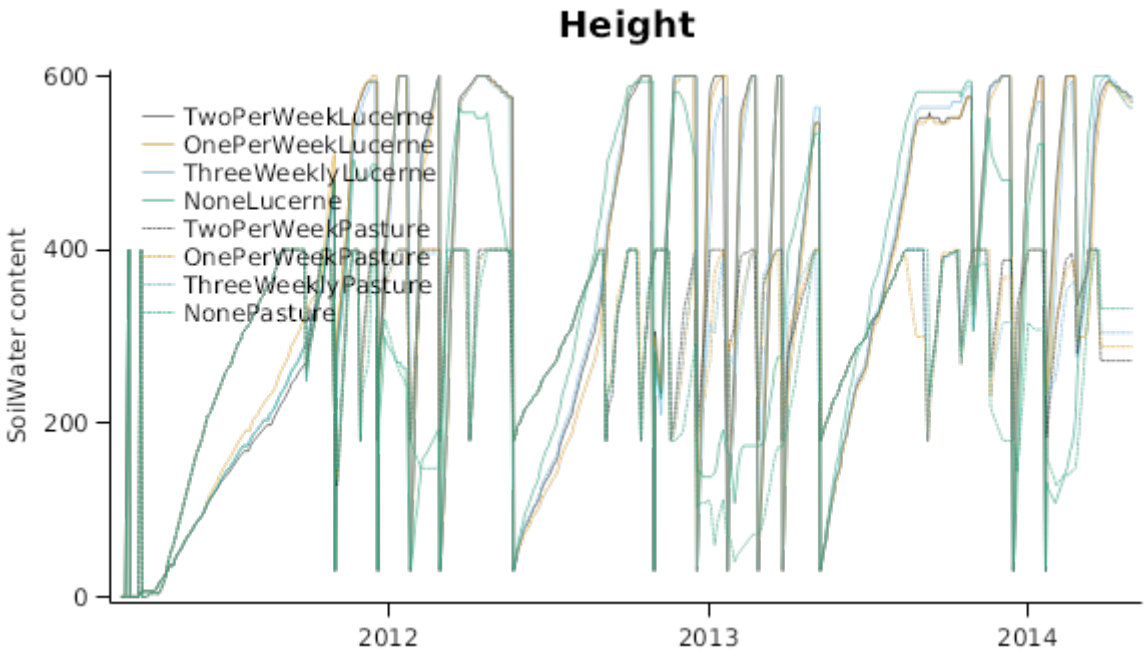

800_1000mm
SoilWater content
0.15
0.1
0.05
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture
2012
2013
2014


1000_1200mm
SoilWater content
0.2
0.15
0.1
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture
2012
2013
2014


1200_1400mm
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture
SoilWater content
0.15
0.1
2012
2013
2014


1400_1600mm
TwoPerWeekLucerne
OnePerWeekLucerne
ThreeWeeklyLucerne
NoneLucerne
TwoPerWeekPasture
OnePerWeekPasture
ThreeWeeklyPasture
NonePasture
SoilWater content
0.15
0.1
0.05
2012
2013
2014





5 References

Brown, Hamish E., Huth, Neil I., Holzworth, Dean P., Teixeira, Edmar I., Zyskowski, Rob F., Hargreaves, John N. G., Moot, Derrick J., 2014. Plant Modelling Framework: Software for building and running crop models on the APSIM platform. Environmental Modelling and Software 62, 385-398.

Jones, C.A., Kiniry, J.R., Dyke, P.T., 1986. CERES-Maize: a simulation model of maize growth and development..