A Programmer's Guide for BioSym - the Biophysical Modelling Toolbox of EnSym

Editor: Joseph Ha
Contributors: Mark Eigenraam, Joselito Chua, Wayne Lewis

Version 1.0
6 July 2010

Preface

The primary purpose of this document is to provide a detailed guide to the computer codes underpinning the BioSym toolbox contained within EnSym - Environmental Systems Modelling Platform. BioSym contains a number of biophysical process-based models that are used to model crop growth, soil erosion, crop residue, crop cover and the water balance components such as surface runoff, soil evaporation, transpiration, soil water storage, redistribution, lateral flows and deep drainage.

This document describes the computer codes in detail, and identifies and locates relevant publications associated with segments of the code. This document will complement the theory manuals for the various computational models contained in BioSym.

Apart from providing documentation on the computer codes, this document highlights concepts and models from the literature that underpin the computer codes as well as their design and the input needed for their application.

Another purpose of this document is to assist with the further development of BioSym, future maintenance and to support collaboration with other research and development institutions.

Acknowledgements

The origin of BioSym is CAT1D (Beverley, 2007) and EPIC (Williams et al., 1989), PERFECT (Littleboy et al., 1989), SWAT 2000 (Neitsch et al., 2002). From December 2008 onward, the Department of Sustainability and Environment (DSE) has been further developing the computer program. To distinguish the divergence in the development of the code, BioSym is adopted as the name for the computer program used in DSE.

Environmental Systems Modelling Platform (EnSym)

EnSym is a modular and user-friendly software platform to facilitate the use of environmental modelling tools. It enables easy and rapid evaluation of environmental outcomes due to changes in land management and climatic conditions. It contains a number of toolboxes that deal with different aspects of the environment including land based biophysical process, groundwater dynamics, spatial and contextual connectivity and finally a set of tools for systematic spatial and temporal reporting.

The software provides a stand-alone package that allows user to operate in a "black box" mode, which hides implementation details and usages of the modelling tools. The overlying user interfaces are written in Matlab programming language using a modern design with graphical user interfaces. The environmental modelling tools can be written in any computer programming language. This may, in the long run, contribute to new ways of sharing scientific research. By sharing both data and modelling tools in a consistent framework, the integration and application of new modelling tools into environmental and natural resource management will be straight forward.

The input interface of EnSym will automatically subdivide a catchment and then extract model input data from map layers and the associated relational data bases for each catchment. Soils, land use, weather, management, model and topographic data are collected and transferred to appropriate model input variables. These data sets for modelling the Victorian environment had been collected over a number of years by the Victorian Government. The output interface allows the user to display output maps and numerical and graphical output data by selecting a point from the map. Users can thus visualise, interpret and test outputs such as sensitive changes in climate, land use and land management practices through a single interface,

EnSym is developed by the Victorian State Government using a version control system to assist in collaborative development, documentation, and feature tracking. While users do not need to study EnSym's source code, collaborators are welcome to become involved and add new modelling modules, tools and functionalities. Matlab provides gateway wrappers to provide easy access to external modelling programs. One particular design aspect of EnSym is that it can handle dynamic model loading and can easily switch between different tools.

Two of the key toolboxes of EnSym are the biophysical (BioSym) and surface flow (D-Flow) toolboxes. The BioSym toolbox simulates daily soil/water/plant interactions, overland water flow processes, soil loss, carbon sequestration and water contribution to stream flow from both lateral flow and groundwater recharge. The agronomic models can be applied to any combination of soil type, climate, topography and land practice. BioSym can thus be used to evaluate the impacts of climate change, vegetation types (e.g. cropping, grazing, forestry and native vegetation) and land management (e.g. forest thinning and stocking rates) in different parts of the landscape. D-Flow predicts surface water flow directions from digital elevation model (DEM). Flow directions are needed in hydrology to determine the flow paths of water and the movement of sediments, nutrients and contaminants.

Biophysical Modelling (BioSym)

BioSym is a continuous time model that operates on a daily time step. The objective in model development was to predict the impact of management on water, sediment, and agricultural chemical yields in the catchment. To satisfy the objective, the model (a) is physically based (calibration is not possible on catchment scale); (b) uses readily available inputs; (c) is computationally efficient to operate on catchment scale in a reasonable time, and (d) is continuous time and capable of simulating long periods for computing the effects of management changes.

The modules in BioSym come from publicly available models. They include CAT1D (Beverley, 2007), EPIC (Williams et al., 1989), PERFECT (Littleboy et al., 1989) and SWAT 2000 (Neitsch et al., 2002). Recently, we upgraded our 3PG+ forest model to its latest version (Feikema et al., 2010). These models are widely used by the environmental modelling community. The readers are referred to the open literature for references of their developments and model validations.

The physically based models in BioSym provide detailed representations of fundamental processes such as plant growth, infiltration, evapotranspiration, runoff, erosion and sediment transport, nutrient and pollutant transport, stream transport and management practices. By modelling each process separately, the simulation is sensitive to climatic change, land use activities and management changes.

BioSym solves for physical processes conceptually by using simplified analytical solutions and empirical equations. The code for BioSym was written with the objective of simulating all major hydrologic components as simply and realistically as possible, and to use inputs readily available over large spatial scales to enhance the likelihood that the model would become routinely used in planning and water resource decision making.

D-Flow

D-Flow uses the principles of single and multiple flow algorithms, such as Deterministic 8 (D8) and D∞, to direct the flow from each cell to one or more of its 8 neighbouring cells based on the steepest downslope drop. It borrows ideas from image processing to correct the shortcoming of the mentioned flow algorithms in their inability to route flow over flats and sinks as well as to take into account the retention capability of depression drained areas. D-Flow follows the flow of water in the catchment, from land areas to streams and rivers, through lakes, to estuaries and ultimately to the ocean. The use of D-Flow is to move the runoff from one part of the landscape to the next. Water movement is related to erosion, to sediment, nutrient and pollutant transport.

Document Presentation

In the document of the various modules or subroutines of the computer program BioSym, the following presentation styles are used:

- Reference material

  Concepts, mathematical equations, models and reference have yellow background.

- Primary subroutine of a model such as dynamic wheat model.

  Primary module name

- Secondary subroutine called by the primary subroutine.

  ■ **Secondary module name**

- Heading / description of a block of codes within a module.

  Calculate layer transpiration

Contents

1 Soil Parameters

The following parameters are read from variable `matsoil`(1-157).

Table 1: Soil parameters read in from `matsoil`.

Line	Variable
1	Soil name
2	ndeps
3 - 82	tmpsoi(1 - 10, 1 - 8)
83 - 92	stones(1 - 10)
93 - 102	segreg(1 - 10)
103 - 142	tmpsoi(1 - 10, 9 - 12)
143	cona
144	uritch
145	cn2b
146	cnred
147	cnrough
148	cnrain
149	kusle $(0.1 \leq 0.6)$
150	pusle
151	slope
152	deflng
153	beta
154	bd
155	cracklimit
156	rootmx
157	y/n (cracking = .true. / .false.)

`tmpsoi` are 12 parameters for each of the 10 soil layers. The actual number of soil layers for a soil is specified by `ndeps` which can be 5 or 6 (see `soils.csv` file). The 12 parameters are passed to `soiladjust` one layer at a time. For top layer, layer 1, `Ksat` is modified by a factor `surfcompft`. The values for layer `ndeps-1` (second last layer) is saved to `plxxx` for use when inserting new layers and the values for the last layer is saved to `smxxx` for use when extending soil depth based on roots. `soiladjust` calls `soilchara`.

The parameters are passed to `soilchara` and to `soilcharb`. In `soilchara`, the soil profile is initialised - void/pore space, fraction of impermeable material, initial soil characteristics, soil moisture, soil salt, soil osmotic effect, soil boron and soil aluminium. In `soilcharb`, calculate depth retention weight factor `wf` (CREAMS equation i-6), drainage factor, field capacity and saturation, root factor for transpiration model.

Table 2: Parameters in `tmpsoi`.

n	Variable
1	`depth 1 - 10`
2	`Airdry 1 - 10`
3	`LowLmt 1 - 10`
4	`UpLmt 1 - 10`
5	`Sat 1 - 10`
6	`Ksat 1 - 10`
7	`SoiCon (soil water constant) 1 - 10`
8	`SoiPow (soil water power) 1 - 10`
9	`Init (moisture) 1 - 10`
10	`Na (sodium) 1 - 10`
11	`Boron 1 - 10`
12	`Aluminium 1 - 10`

2 Irrigation

Table 3: Irrigation parameters in **matmang** and their typical values when `irrigrot = 1`.

n	Variable	Value
34	irrigrot - irrigation trigger (0 = off, 1 = on)	1
35	irrigcyc - irrigation cycle (a = all year, c = cropphase only)	a
36	irrigdst - irrigation season start day	1
37	irrigmst - irrigation season start month	1
38	irrigdfn - irrigation season stop day	31
39	irrigmfn - irrigation season stop month	12
40	irrigamt - fixed irrigation amount (mm) per day	0
	- overwrites swd and %sat approach	
41	irrigwet - irrigation scheme (0 = based on swd, 1 = based on %sat	1
42	irrigswd - irrigation trigger based on irrigdep depth	50
	- swd in mm if irrigwet = 0 or %sat if irrigwet = 1	
43	irrigdep - irrigation depth (mm) to calculate swd or %sat	600
44	irrigmax - maximum irrigation delivery rate (mm/day)	10
45	irrigsep - minimum days between successive irrigation events	0
46	irrigppt - rainfall threshold above which no irrigation occurs	9999

manager

Consider irrigation:
`if(irrigrot(idcrop) > 0 .and. irrigwindow(dayno) > 0) irigok=1`

`if(irigok = 1)`, define irrigation number (`irrigidd = irrigwindow(dayno)`) and the following actions:

Activate irrigation event (`irrigonn`) based on within-crop cycle or all year / continuous.

`if(irrigcyc = 'c')`, `irrigonn = 1 if(cropphase)`.
`if(irrigcyc = 'a')`, `irrigonn = 1`.

Consider days since sowing (`if(irrigsow(idcrop, irrigidd) > 0)`,

```
irrigonn = 0
if(cropphase) then
  if(dayssinceplant ≥ irrigsow(idcrop,irrigidd)) irrigonn = 1
endif
```

That is, only consider irrigation if after sowing.

Do not irrigate (`if(irrigcnt > 0) irrigonn = 0`) if last irrigation day < minimum days between irrigation.

Once irrigation event is satisfied (`irrigonn = 1`), consider whether to actually apply irrigation by checking the following:

- No irrigation on rainy days

  `if(rain > irrigppt(idcrop, irrigidd) .and. ifloods > 0) irigok = 0`

- No irrigation if preceeding rainfall < threshold (`irrigdyp`). This is only carried out if (`irrigdys(idcrop,irrigidd) > 0`) and `irrigdys` specifies number of previous days to sum rain.

  call `sumrain`, `if(accrain ≥ irrigdyp(idcrop,irrigidd)) irigok=0`

NB: The last case never occurs in the DLL as `irrigdys` and `irrigdyp` are not passed in from Matlab.

If irrigation is to be applied (`irigok > 0`), assign irrigation via one of the followings depending on the value given to `irrigamt`:

- `if(irrigamt(idcrop,irrigidd) > 0.0)`
  `airr = airr + irrigamt(idcrop, irrigidd)` and
  `irrigcnt = irrigsep(idcrop,irrigidd)`.

  That is, apply irrigation by a specified amount (`irrigamt`).

- `if(irrigamt(idcrop,irrigidd) ≤ 0.0)`,
  calculates the followings quantities for soil layers ≤ `irrigdep (idcrop, irrigidd)`:
  `swdirg` = $\sum$ `(fc(i) - sw(i))`,
  `swxirg` = $\sum$ `swmax(i)` and
  `swmirg` = $\sum$ `sw(i)`

  If at least one of the followings holds:
  `irrigwet(idcrop,irrigidd) = 0 .and. swdirg ≥ irrigswd(idcrop,irrigidd)`
  `irrigwet(idcrop,irrigidd) = 1 .and. pstirg ≤ irrigswd(idcrop,irrigidd)`
  where `pstirg = swmirg*100/smxirg`
  then apply irrigation
  `airr = airr + min(swdirg, irrigmax(idcrop, irrigidd))`.
  `irrigcnt = irrigsep(idcrop,irrigidd)`.

  Here, apply irrigation based on either soil water deficit (if `irrigwet = 0`) or percent of saturation (if `irrigwet = 1`). The value (`irrigswd`) in either mm or % sets the threshold to trigger irrigation. The amount of irrigation to apply is constrainted by field capacity (`fc(i)`) and `irrigmax` whichever is the smaller.

Field capacity is the amount of soil moisture or water content held in soil after excess water has drained away and the rate of downward movement has materially decreased, which usually takes place within 2 - 3 days after a rain or irrigation in previous soils of uniform structure and texture. The physical definition of field capacity (expressed symbolically as θ_{fc}) is the bulk water content retained in soil at -33 J/kg (or -0.33 bar) of hydraulic head or suction pressure. The term originated from Israelson and West (1922) and Frank Veihmeyer and Arthur Hendrickson (1931).

Israelson, O.W. and West, F.L., 1922. Water holding capacity of irrigated soils, *Utah State Agricultural Experiment Station Bull*, **183**, 1-24.

Veihmeyer, F.J. and Hendrickson, A.H., 1931. The moisture equivalent as a measure of the field capacity of soils, *Soil Science*, **32**, 181-193.

3 Water Balance

The soil water balance includes rainfall infiltration, overland flow, soil and plant water extraction, moisture redistribution, drainage (recharge), and water table interactions. Soil water movement in both the unsaturated and saturated zones is described by a mixed form of the Richard's equation. Overland flow can be generated when the rainfall rate exceeds the infiltration rate of the soil, and when rain falls on a saturated surface. A watertable may develop anywhere within the soil profile. If non-zero slope is specified, then lateral subsurface flow occurs via any saturated water table at a soil layer boundary and is described by Darcy's law. Evaporation and transpiration draw water out of the soil.

Water entering the soil profile is initially determined by subtracting the calculated surface runoff from the total daily precipitation and irrigation. Once in the soil profile, water can be removed by evapotranspiration, lateral flow, downward movement if soil capacity is exceeded. Water fills up lower soil layers until it exits the soil profile and becomes ground water recharge.

The following is done during daily simulation:

In BioSyM, precipitation:

`rain` read in from `matclim` for the day and

```
rain = rain * rainfct

dprain = rain if(cropflag ==  9 and day == 1)

erain = rain
```

- **manager**

  `airr` is used to store water from flood if there is any.

  Add one of the followings from irrigation according to `irrigamt`:

  ```
  airr = airr + irrigamt(idcrop,irrigidd)
  airr = airr + min(swdirg, irrigmax(idcrop, irrigidd))
  ```

  Add flood, irrigation, runon and sub-surface lateral inflow to `rain`.

  ```
  rain=rain+airr+runi+subsi

  erain=rain
  ```

- **intercept**

  ```
  rain = rain - xint    ! rain is throughfall

  erain = rain
  ```

- **soilcracking**

  Soil crack allows water to go to lower layers of soil profile. For cracks to occur, the top and second layers must be less than 30% and 50% of field capacity. This is allowed only if rain > 10.

Amount of water available to fill crack = `min(cracklimit, rain)`. Amount of water in crack in each layer (fill lowest layer first) is a maximum of 50% of field capacity - `(fc(i)/2 - sw(i))`. `tred` is total amount of water in cracks and `red(i)` is amount of water in crack in layer i. Note water is not distributed to the crack in layer 1.

```
erain = rain + red(1) - amin1(cracklimit, rain)
red(1) = 0.0
```

- **runoff**

```
runf=((erain-0.2*s)**2.0) / (erain+0.8*s)

infil=erain-runf

roff=runf
```

- **soilevaporation:**

```
se = se1 + se2
```

- **plant**

There are 9 options depending on plant type. `cropflag = 1, 2, 4, 5, 6, 7, 8, 9 and 10`. Subroutines 1 lists calls to read in the appropriate data from `matcrop`. Set appropriate `plantxxx` to true.

Table 4: Logical variable to set .true.

cropflag	logical variable	others
1	plantwheat	ivar = 2, popn = 100
2	plantsunflower	ivar = 2, popn = 5
4	plantetpan	
5	plantlai	
6	plantgrass	
7	plantgrasg	
8	plantgrasn	
9	planttrees	iplnt3pg = 1
10	plantnativ	

```
ivar = cropvariety(icrop)
popn = croppopn(icrop)
```

NB `cropvariety` and `croppopn(1)` are read from `sequencefile, Manager.seq` which never occurs when DLL is run.

Set `cropplant, dotillage, cropphase` to `.true.`.

Set `firsttillage, rainok` to `.false.` and `tillagecode='p'`.

Table 5: Plant types

n	Subroutines 1	Subroutine 2
1	wheatMat	wheat
2	sunflowerMat	
4	etpanMat	etpan
5	laiMat	laimodel
6	grsparams, grstempfn	
7	grsparams, grstempfn, grginital	
8	nswparams, nswinital	
9	init3pg, mat3pg	
10	natmat, nattempfn, natinital	

4 Potential Evaporation

convertpana

Computes potential evaporation according to the Priestley Taylor equation or a simple potential.

s slope of the saturation vapour pressure-temperature relationship (kPa/C)

$$s = 4098 \left(0.6108 \exp\left(\frac{17.27T}{T + 237.3}\right)\right) / (T + 237.3)^2$$

Tetens (1930); Murray (1967). An alternative is the Staghellini equation (ASA Standards, 1998):

$$s = 0.04145 \exp(0.06088T)$$

Calculate saturation vapour pressure

```
satvpd = 0.611 * exp( (17.27 * temp) / (temp + 237.3) )
```

Calculate slope of vapour pressure curve

```
slopevpd = 4098.0 * satvpd / ( (temp+237.3)**2.0 )
```

Atmospheric pressure $P = P_0 \left(\frac{T_{K0} - \eta z}{T_{K0}}\right)^{g/\eta R}$ where $T_{K0} = 293.16$ is reference air temperature at sea level, $\eta = 0.0065$ is lapse rate, $g = 9.807$ is gravity, $R = 287.06$ is specific gas constant (J/kg/K), $P_0 = 101.3$ is atmospheric pressure at sea level (kPa) and z is elevation.

Calculate atmospheric pressure

```
xelev = 93.7
if(paddelevm > 0.0) xelev = paddelevm
atmpkpa = 101.3 * ( ((293.0 - (0.0065 * xelev)) / 293.0)**5.26 )
```

Latent heat of vapourisation (MJ/kg) $\lambda = 2.501 - 0.002361T$

Calculate latent heat of vaporisation

```
xlatent = 2.501 - ( temp * 2.361 * 1.0e-3 )
```

Psychrometric constant (kPa/C) $\gamma = C_p P/\epsilon\lambda$, where $\epsilon = 0.622$ (ratio of molcular weight to dry air) and $C_p = 0.001013$ (MJ/kg/C).

Calculate psychrometric constant

```
psyconst = 0.00163 * atmpkpa / xlatent
```

Net radiation (MJ/m^2/d) is the difference between incoming and outgoing radiation $r = R_s - R_l$, where $R_s = (1 - a)R_i$ the net short wave radiation, a canopy reflection coefficient (which is 0.23 for hypothetical reference crop) and R_l is net isothermal long wave radiation.

Calculate net radiation

```
xnetrad = (rad * 0.62) - 1.47
```

Soil heat flux parameter (MJ/m^2/d) G is the energy that is utilized in heating the soil. G is positive when the soil is warming and negative when the soil is cooling. The effect of soil heat flux is ignored for daily calculation.

Calculate soil heat flux parameter

```
soiflux = 0.0
```

Let α P-T coefficient (= 1.26), s slope of the saturation vapour pressure-temperature relationship (kPa/C), γ psychrometric constant (kPa/C), λ latent heat of vapourisation (MJ/kg), r net radiation (MJ/m^2/d) and G soil heat flux (MJ/m^2/d).

$$E = \alpha \frac{s}{s + \gamma} \frac{r - G}{\lambda}$$

Calculate potential evaporation

```
albedo = 1.60
fct1 = slopevpd / (slopevpd + psyconst)
fct2 = (xnetrad - soiflux) / xlatent
ptet = albedo * fct1 * fct2

pan = ptet
```

Simple potential evaporation

```
pan = 0.8 * pan
```

Values for Priestley-Taylor Coefficient

From the literature:

Values 1.26 for grass surface surrounded by grass.

Values 2.04 for grass surface surrounded by forest.

Values 1.65 for forest surface surrounded by grass.

Values 3.62 for forest surface surrounded by forest.

For well-water crop, 1.42.

References

ASAE Standards. 1998. EP406.2: heating, cooling, and ventilating greenhouses. St. Joseph, MI, USA.

ASCE. 1990. Evapotranspiration and irrigation water requirements. ASCE. Manuals and reports on engineering practice. No. 70. New York, NY, USA.

Holtslag, A.A.M., Van Ulden, A.P., 1983. A simple scheme for daytime estimates of the surface fluxes from routine weather data. *J. Climate Applied Meteorology*, **22**, 517-529.

Lhomme, J.-P., 1996. A theoretical basis for the Priestley-Taylor coefficient. *Boundary-Layer Meteorol.*, **82**, 179-191.

Murray, F.W., 1967. On the computation of saturation vapor pressure. *J. Appl. Meteor.*, **6**, 203-204.

Steiner, J.L., Howell, T.A. and Schneider, A.D., 1991. Lysimetric evaluation of daily potential evapotranspiration models for grain sorghum. *Agron. J.*, **83**, 240-247.

Tetens, O., 1930. Uber einige meteorologische Begriffe. *Z. Geophys.*, **6**, 297-309.

Villalobos, F.J., Mateos, L., Orgaz, F. and Fereres, E., 2002. Fitotecnia. Bases y tecnologia de la produccion agricola. Mundi-Prensa, Madrid, Spain.

5 Type 4 / ET:PAN

This is the crop factor model of PERFECT (Littleboy el at., 1999). Transpiration is calculated from the user defined annual distribution of green cover and a crop factor (`cf`). Phenology of this model is specified by the user-defined input of the number of days from planting to harvest. Inputs can be derived from field data or relevant literature (Doorenbos and Pruitt, 1977).

etpanMat

It reads crop parameters for model ET:PAN from `matcrop`.

Line	Variable	Description
1	Parameter	Description
2	Cropflag	Internal flag (do not change)
3	cf	Crop Factor
4	EtPanLAI	LAI/cover Factor
5	EtPanWUE	Water Use Efficiency (g/m2 per mm of transpiration)
6	EtPanHI	Harvest Index
7	EtPanDays	No. of days from Planting to Harvest
8	drymini	Initial dry matter at planting (kg/ha) - (only relevant for pastures)
9	cresmin	Minimum residual (kg/ha) - (only relevant for pastures)
10	MaxResidCover	Maximum Residue Cover (0-1)
11	ddwe	Daily root growth (mm)
12	dwemax	Maximum root depth
13	irootfn	Root function 0=default 1=Ritchie 2=power law 3=proportional depth
14	idoroots	Soil root depth option 0=root depth limited by max soil depth; 1=extend soil depth to root depth; 2=insert soil layers
15	dayswake	Days for plant to reach full activity
16	dayssleep	Days for plant to reach full activity
17	Crplng	Erosion flow length (m)
18	Day1	Day number
19	Day15	Day number
20	Day46	Day number
21	Day74	Day number
22	Day105	Day number
23	Day135	Day number
24	Day166	Day number

25	Day196	Day number
26	Day227	Day number
27	Day258	Day number
28	Day288	Day number
29	Day319	Day number
30	Day349	Day number
31	Day366	Day number
32	Day1	Cover
33	Day15	Cover
34	Day46	Cover
35	Day74	Cover
36	Day105	Cover
37	Day135	Cover
38	Day166	Cover
39	Day196	Cover
40	Day227	Cover
41	Day258	Cover
42	Day288	Cover
43	Day319	Cover
44	Day349	Cover
45	Day366	Cover
46	resrat	Residue decay rate (%)
47	restrg	Day number below which resrd1 applies else resrd2 applies
48	resrd1	Residue decay rate1 (kg/day)
49	resrd2	Residue decay rate2 (kg/day)
50	idc	crop/landcover category(1=warm season annual legume 2=cold season annual legume 3=perennial legume 4=warm season annual 5=cold season annual 6=perennial 7=trees)
51	wsyfc	Value of harvest index between 0 and HVSTI which represents the lowest value expected due to water stress ((kg/ha)/(kg/ha))
52	biomjs	Biomass-energy ratio
53	cnyldc	Fraction of nitrogen in yield (kg N/kg yield)
54	cpyldc	Fraction of phosphorus in yield (kg P/kg yield)
55	rsdco	Decay rate constant
56	bn1	Nitrogen uptake parameter #1 - normal fraction of N in crop biomass at emergence (kg N/kg biomass)

57	bn2	Nitrogen uptake parameter #2 - normal fraction of N in crop biomass at 0.5 maturity (kg N/kg biomass)
58	bn3	Nitrogen uptake parameter #3 - normal fraction of N in crop biomass at maturity (kg N/kg biomass)
59	bp1c	Phosphorus uptake parameter #1 - normal fraction of P in crop biomass at emergence (kg P/kg biomass)
60	bp2c	Phosphorus uptake parameter #2 - normal fraction of P in crop biomass at 0.5 maturity (kg P/kg biomass)
61	bp3c	Phosphorus uptake parameter #3 - normal fraction of P in crop biomass at maturity (kg P/kg biomass)

Table 6: Crop parameters for ET:PAN read in from `matcrop`.

NB `wsyfc` is in csv file and passed in from matlab but is not read by subroutine.

Call `rootl, etpanrecd, rootd`.

- **rootl** set maximum root length `dwemax`
- **etpanrecd** transfers `etpan` parameters to array `reletpan`.
- **rootd** calculates maximum root depth using `smxxx` of last soil layer. Calls `soilchara`, `soilextn`, `soilcharb`, `initialsw`.

etpan

Call `etpanload, etpanphenology, roots, etpantranspiration, etpanbiomass`.
Call `etpanyield` if `cropharvest = .true.`.

- **etpanload** loads `etpan` parameters from array `reletpan`.

```
drym=drymini/10.0
call rootd
call erosnls
```

- **etpanphenology** computes plant stage (`pstage`).

```
cropharvest=.true. if pstage ≥ 3
pstage = 3.0 * dayssinceplant / etpandays
```

- **roots** calculates linear root growth.

```
dwe = dwe + ddwe
```

`0 ≤ dwe ≤ dwemax`

- **etpantranspiration** calculates transpiration for ET:PAN model.

  Assign crop cover (`ccov`) from `etfac(dayno)`. The cover curve (`etfac`) is generated from `ity4day` and `ity4cov`.

```
gcov = ccov
lai = etpanlai * ccov
```

Compute potential transpiration:

```
pott = amin1(ccov * pan, pan - se)
if(pott = 0) return
```

Calculate root penetration per layer from root distribution function (`dfac(i)`):

```
call rootf
```

There are 3 schemes for computing `dfac` depending on the value of `irootfn = 1, 2, 3`, where 0 for original scheme, 1 for Ritchie exponential decay function, 2 for distribution based on depth power law decay function and 3 for distribution based on proportional depth. If rooting depth `dwe < 1.0e-8`, `dfac(i) = 0`.

Calculate layer transpiration (`trans(i)`):

```
trans(i) = dfac(i) * pott * cf
```

which is limited by `sw(i)` for each soil layer i: `0 ≤ trans(i) ≤ sw(i)`

The transpiration is reduced if the total is greater than `pott`.
If (`psup > pott`), `trans(i) = trans(i) * pott / psup`, where `psup =` $\sum$ `trans(i)`.

Calculate water stress index:

```
wsi = psup / pott if psup ≤ pott
```

- **etpanbiomass** calculates biomass for ET:PAN model which is proportional to transpiration.

```
drymadd = etpanwue * ttrans
drym = drym + drymadd
```

Calculate minimum residual cover:

```
if(cres < cresmin) then
  drymlos = (cresmin - cres) / 10.0
  if(drymlos > drym) drymlos = drym - 0.1
  drym = drym - drymlos
  cres = cres + (drymlos * 10.0)
  cresadd = cresadd + (drymlos * 10.0)
  covm = maxresidcover * (1. - exp(-1. * cres/1000.))
  if(covm < 0.0) covm = 0.0
  if(covm > 1.0) covm = 1.0
endif
```

- **etpanyield** calculates yield for ET:PAN model which is proportional to dry matter.

```
yield = etpanhi * drym
```

6 Type 1 / Dynamic Wheat Model

The dynamic wheat model of BioSym is the Woodruff-Hammer wheat model in PERFECT. The model consists of sub-models for phenology, crop growth, growth of leaf area, soil water and crop yield.

Hammer, G.L., Woodruff, D.R. and Robinson, J.B., 1987. Effects of climatic variability and possible climatic change on reliability of wheat cropping - a modelling approach, *Agriculture, Forestry & Meteor.*, **41**, 123-142.

wheatMat

Reads crop parameters from `matcrop`.

Line	Variable	Description
2	cropflag	Internal flag (do not change)
3	cropvariety	Varieties of the crop
	fixed phenology inputs	
4	fixedphenology	Fixed phenology to anthesis (Y/N)
5	egdd	Degree days for emergence to occur
6	days2anth	Number of days planting to anthesis
7	pgdd	Degree days from anthesis to harvest
	degree days phenology inputs	
8	degreedays	Phenology from degree days (Y/N)
9	egdd	Degree days for emergence to occur
10	pgdd	Degree days from anthesis to harvest
11	whdayno(1)	Day number point #1
12	whdayno(2)	Day number point #2
13	whdayno(3)	Day number point #3
14	whdayno(4)	Day number point #4
15	whdayno(5)	Day number point #5
16	whdayno(6)	Day number point #6
17	whdayno(7)	Day number point #7
18	whdayno(8)	Day number point #8
19	whdayno(9)	Day number point #9
20	whdayno(10)	Day number point #10
21	emergdd(1)	Degree days emergence to anthesis point #1
22	emergdd(2)	Degree days emergence to anthesis point #2
23	emergdd(3)	Degree days emergence to anthesis point #3
24	emergdd(4)	Degree days emergence to anthesis point #4

25	emergdd(5)	Degree days emergence to anthesis point #5
26	emergdd(6)	Degree days emergence to anthesis point #6
27	emergdd(7)	Degree days emergence to anthesis point #7
28	emergdd(8)	Degree days emergence to anthesis point #8
29	emergdd(9)	Degree days emergence to anthesis point #9
30	emergdd(10)	Degree days emergence to anthesis point #10
	full phenology equation	
31	fullphenology	Phenology from full phenology equation (Y/N)
32	egdd	Degree days for emergence to occur
33	pgdd	Degree days from anthesis to harvest -> 505
34	b01	Base temperature (planting to emergence)
35	b12	Base temperature (emergence to anthesis)
36	b23	Base temperature (anthesis to harvest)
	shoot growth factors	
37	whpstag(1)	Phenological stage point #1
38	whpstag(2)	Phenological stage point #2
39	whpstag(3)	Phenological stage point #3
40	whpstag(4)	Phenological stage point #4
41	whpstag(5)	Phenological stage point #5
42	whpstag(6)	Phenological stage point #6
43	whpstag(7)	Phenological stage point #7
44	whpstag(8)	Phenological stage point #8
45	whpstag(9)	Phenological stage point #9
46	whpstag(10)	Phenological stage point #10
47	shratio(1)	Shoot ratios point #1
48	shratio(2)	Shoot ratios point #2
49	shratio(3)	Shoot ratios point #3
50	shratio(4)	Shoot ratios point #4
51	shratio(5)	Shoot ratios point #5
52	shratio(6)	Shoot ratios point #6
53	shratio(7)	Shoot ratios point #7
54	shratio(8)	Shoot ratios point #8
55	shratio(9)	Shoot ratios point #9
56	shratio(10)	Shoot ratios point #10
	temperature indices	
57	whtempp(1)	Temperature for temperature index #1

58	whtempp(2)	Temperature for temperature index #2
59	whtempp(3)	Temperature for temperature index #3
60	whtempp(4)	Temperature for temperature index #4
61	whtempp(5)	Temperature for temperature index #5
62	whtempp(6)	Temperature for temperature index #6
63	whtempp(7)	Temperature for temperature index #7
64	whtempp(8)	Temperature for temperature index #8
65	whtempp(9)	Temperature for temperature index #9
66	whtempp(10)	Temperature for temperature index #10
67	whtempi(1)	Temperature index on growth point #1
68	whtempi(2)	Temperature index on growth point #2
69	whtempi(3)	Temperature index on growth point #3
70	whtempi(4)	Temperature index on growth point #4
71	whtempi(5)	Temperature index on growth point #5
72	whtempi(6)	Temperature index on growth point #6
73	whtempi(7)	Temperature index on growth point #7
74	whtempi(8)	Temperature index on growth point #8
75	whtempi(9)	Temperature index on growth point #9
76	whtempi(10)	Temperature index on growth point #10
		starting values
77	slai	Starting LAI (cm2̂/cm2̂)
78	stdw	Starting dry matter (g/m2̂)
79	sdwe	Starting root depth (mm)
80	ddwe	Root growth per day (mm) -> 13.5
81	dwemax	Maximum root depth (mm)
82	irootfn	Root function 0=default 1=Ritchie 2=power law 3=proportional depth
83	exc	Extinction Coefficient #1
84	exc1	Extinction Coefficient #2
		crop depth parameters
85	killwheat(1)	Water stress limit for extreme water stress
86	killwheat(2)	Number of consecutive days of extreme water stress for kill
		maximum residue cover
87	maxresidcover	Maximum Residue Cover (0-1) -> 0.98
88	crplng	Erosion flow length (m)
		crop performance data
89	iwhscm	Solution scheme (0=Qld 1=Vic)
90	whruef	Radiation use efficiency (Qld=1.84 Vic=1.60)
91	whlais	Leaf area conversion (Qld=0.0115 Vic=0.02)
92	whyldb	Yield coefficient
		residue decay data
93	resrat	Residue decay rate (%)

94	restrg	Day number below which resrd1 applies else resrd2 applies
95	resrd1	Residue decay rate1 (kg/day)
96	resrd2	Residue decay rate2 (kg/day)
		nutrient data
97	idc	crop/landcover category(1=warm season annual legume 2=cold season annual legume 3=perennial legume 4=warm season annual 5=cold season annual 6=perennial 7=trees)
98	wsyfc	Value of harvest index between 0 and HVSTI which represents the lowest value expected due to water stress ((kg/ha)/(kg/ha))
99	biomjs	Biomass-energy ratio
100	cnyldc	Fraction of nitrogen in yield (kg N/kg yield)
101	cpyldc	Fraction of phosphorus in yield (kg P/kg yield)
102	rsdco	Decay rate constant
103	bn1	Nitrogen uptake parameter #1 - normal fraction of N in crop biomass at emergence (kg N/kg biomass)
104	bn2	Nitrogen uptake parameter #2 - normal fraction of N in crop biomass at 0.5 maturity (kg N/kg biomass)
105	bn3	Nitrogen uptake parameter #3 - normal fraction of N in crop biomass at maturity (kg N/kg biomass)
106	bp1c	Phosphorus uptake parameter #1 - normal fraction of P in crop biomass at emergence (kg P/kg biomass)
107	bp2c	Phosphorus uptake parameter #2 - normal fraction of P in crop biomass at 0.5 maturity (kg P/kg biomass)
108	bp3c	Phosphorus uptake parameter #3 - normal fraction of P in crop biomass at maturity (kg P/kg biomass)

Table 7: Crop parameters for wheat read in from `matcrop`.

Call `rootl, wheatrecd, rootd`.

- **rootl** sets maximum root length `dwemax`
- **wheatrecd** transfers `wheat` parameters to arrays `intwheat`, `relwheat` and `logwheat`.
- **rootd** calculates maximum root depth using `smxxx` of last soil layer. `idoroots` sets options whether to extend and add soil layer with respect to root depth. Calls `soilchara`, `soilextn`, `soilcharb`, `initialsw`.

wheat

```
if(pstage > 1.0) call wheattranspiration
if(cropplant) call wheatload
if(cropkill) call waterstressdeath(killwheat)
call wheatphenology
if(pstage > 1.0) call wheatgrow
if(pstage > 1.0) call wheatleaf
if(pstage > 1.0) call wheatyield
```

Call `wheattranspiration`, `wheatload`, `waterstressdeath`, `wheatphenology`, `wheatgrow`, `wheatleaf`, `wheatyield`.

- **wheattranspiration** computes transpiration for dynamic wheat model:

  A key stage in crop development following emergence is of the terminal spikelet initiation (TSI) as it marks the end of the initiation of spikelet primordia and thus potential grain sites and is known to be highly sensitive to temperature (Porter and Gawith, 1999).

  Initialise transpiration array (`trans(i) = 0`) and computes `mcfc(i) = sw(i) / fc(i)` and $0 \leq$ `mcfc(i)` ≤ 1.

  Compute potential plant transpiration:

  Daily potential transpiration rate (`pott`) is the maximum amount of water that can be transpired by the plant. It is the maximum of a proportion of pan evaporation (`gcov * pan`) and a function of temperature (`0.006 * popn * temp / 100`) where `gcov` is effective crop cover on transpiration and `popn` is the plant density at establishment. It is limited by pan evaporation minus soil evaporation. Littleboy et al. (1999)

  ```
  pott = amin1(gcov * pan, pan - se)
  pott = amax1(pott, 0.006 * popn * temp / 20)
  pott = amin1(pott, pan - se)
  ```

  If `pott=0`, no plant transpiration and `return`.

  NB `popn = 100` if `croppopn(icrop) = 0` in ■ **plant** .

  Calculate factor for how far roots are into soil water profile:

  ```
  dfac(1)=1.0
  dfac(i)=amin1(1.0,(amax1(dwe-depth(i),0.))/(depth(i+1)-depth(i))), i=2,ndeps
  ```

  Calculate transpiration (soil water uptake) from each layer:

  `trans(i)=dfac(i)*amax1(0.5,tsi*factor(i)*mcfc(i)**1.67)` if `i` $\neq$ `ndeps`
  `trans(i)=dfac(i)*amax1(1.0,tsi*factor(i)*mcfc(i)**1.67)` if `i = ndeps`
  and check that
  $0 \leq$ `trans(i)` $\leq$ `sw(i)`

`trans(i)` is further constrainted by `pott`:
If (`psup` > `pott`), `trans(i) = trans(i) * pott / psup`, where `psup =` $\sum$ `trans(i)`.

`factor(i)` describes the distribution of root density throughout the soil profile. It represents the potential water uptake per unit depth of a given soil layer. They are calculated in ■ **soilcharb** .

```
do i = 1, ndeps
   diff = (depth(i+1) - depth(i))
   depav = (depth(i) + depth(i+1)) / 2.
   factor(i) = (0.01498-(0.00424*depav/1000.)) * diff
   factor(i) = amin1(0.014667*diff, factor(i))
   factor(i) = amax1(0.011167*diff, factor(i))
enddo
```

That is, it is limited by `0.011167 * diff` < `factor(i)` < `0.014667 * diff`.

In the above, `trans(i)` is transpiration from layer i, `factor(i)` is the root density factor for layer i and `mcfc(i)` is the ratio of soil water (`sw`) to field capacity (`fc`) for layer i.

NB Temperature index (`tsi`) used to modify transpiration (`trans(i)`) is not set until call to `wheatphenology`. `tsi` is obtained by interpolating `tindex-temps` curve at `temp`.

Calculate water stress index:
If `psup` ≤ `pott`, `wsi = psup / pott` and `wsi1 = 1.0 - (psup/pott)`.
If `psup > pott`, `wsi = 1`.

■ **wheatload** loads parameters for dynamic wheat model from array `intwheat`, `relwheat` and `logwheat`. Call `rootd` and `erosnls`. The parameter `phenologytest` is incremented by 1 for true value for each of `fixedphenology`, `degreedays` and `fullphenology`.

■ **waterstressdeath** kills a crop due to water stress.

```
if(pstage ≤ 2.0) then
  if(wsi ≤ death(1)) then
    sumkill = sumkill+1
  else
    sumkill = 0
  endif
  if(sumkill  ≥  death(2)) cropharvest = .true.
endif
```

For crops Wheat1, Wheat2, Wheat3, Triti1, Triti2 and Dummy, `death(1) = 0.2` and `death(2) = 21`. In other words, kill the crop if water stress persists for 21 days or more.

■ **wheatphenology** computes phenology for dynamic wheat model. In other words, computes the stage of the plant it is at.

The simulation of crop development is based on thermal time, which is the required daily accumulation of average air temperature above a base temperature and below a cutoff temperature to reach given growth stages (Stockle et al., 2003).

There are 3 options available for estimating phenology.

The first option is for the user to specify the number of days from planting to emergence, emergence to anthesis and anthesis to harvest.

The second option calculates phenology using the concept of cumulative degree days (`temp - base`) where `temp` is average daily temperature and `base` is user defined base temperature. Base temperature is function of phenological stage of crop - 0 from planting to anthesis and 6 from anthesis to physiological maturity. Total degree days is dependent on variety and time of planting.

The third option is a fully dynamic phenology equation: `anth = 0.014 * (1.0 - exp(-0.4623 * (temp-4.54))) * (1.0 - exp(-pp * (dlen - pp*17.44)))`. For a long season variety and daylength < 11.95, `anth = 0.6 * anth` where `dlen` is daylength and `pp` is photoperiod constant. `pp = 0.24` for quick early maturing genotypes, `pp = 0.507` for late maturing genotypes and `pp = 0.6851` for long season genotypes.
Littleboy et al. (1999)

Set parameterisation (`pfactor` array) based on either Condamine data (`iwhqld = 1`) or Birchopp data (`iwhvic = 1`). Assume Condamine data (`pfactor/0.35, 0.74, 1.00, 0.00/`) is used if none is chosen. Birchopp data are (`pfactor/0.35, 0.74, 0.96, 1.00/`).

Compute degree days to anthesis (`agdd`) for `dayno` if using degree days model by interpolating `plday-gddv` curve at dayno.

Test for emergence (`emerge`):
`if(emerge` $<$ `1.0)` compute

```
gdd1 = gdd1 + temp - b01  and
emerge = 1.0 - dim(egdd, gdd1) / egdd  and set
pstage = amin1(1.0, emerge)
```

If no emergence (`if(emerge` $\neq$ `1)`, `return`.

If emergence has occurred and has not occurred before, i.e.
(`if(emerge = 1.0 .and. dayemerge = 0)`),

set initial values for the following parameters and `return`:

```
dayemerge=dayno
danth = days2anth + dayplant
ccov = 1.0 - exp(- exc * lai)
gcov = amax1( amin1(0.03, 2.0*lai), 1.0-exp(-exc1*lai))
tsi= yintrp( tindex, temps, nptix, temp)
drym = stdw = 0.035
rdrym = drym/yintrp(srr, pstg, npsrr, pstage) - drym
lai = slai = 0.0005
dwe = sdwe = 100      (O'Leary et al. 1985)
emerge = 1.0
pstage = 1.0
```

Do the followings only in subsequent calls after emergence:

Check if anthesis has occurred:

Sum degree days from emergence to anthesis:

`if(anth < 1.0) gdd2 = gdd2 + temp - b12`

If `anth < 1`, compute `anth` by one of the followings:

- full phenology equation (`fullphenology = .true.`),

```
dlen = daylen()
pp = 0.6851 * pfactor(ivar)
if(ivar = 3 .and. dlen < 11.95) then
   anth = anth + 0.6 * 0.014 * (1.0 - exp(-0.4623 * (temp-4.54)))
else
   anth = anth + 0.014 * (1.0 - exp(-0.4623 * (temp-4.54))) * &
                 (1.0 - exp(-pp * (dlen - pp*17.44)))
endif
```

For `ivar = 1` (`cropvariety`), `pp = 0.6851 * 0.35` ≈ `0.24`.

- input days to anthesis (`fixedphenology = .true.`) or

```
dypl = dayplant
dyem = dayemerge
if(danth ≤ 365) then
   anth = amin1((float(dayno-dyem)/float(danth-dyem)), 1.0)
endif
if(danth ≥ 365 .and. dayno > dypl) then
   anth = amin1((float(dayno-dyem)/float(danth-dyem)), 1.0)
endif
if(danth ≥ 365 .and. dayno ≤ dypl) then
   anth = amin1((float(dayno-dyem+365)/float(danth-dyem)), 1.0)
endif
if(anth = 1.0) anth = 1.01
```

- degree days (`degreedays = .true.`).

```
anth = 1.0 - dim(agdd, gdd2) / agdd
if(anth = 1.0) anth = 1.01
```

If anthesis has occured (`anth > 1`):
set `anth = 1`, `cropanthesis = .true.` and `return`.

If anthesis has not occured (`anth < 1`):
set `pstage = emerge + anth` and `return`.

Do the followings only in subsequent calls after anthesis has occurred:

Sum degree days from anthesis to maturity

```
gdd3 = gdd3 + temp - b23
```

Check plant maturity based on input days to maturity or degree days to maturity:

```
pmat = 1.0 - dim(pgdd,gdd3) / pgdd
if(pmat .eq. 1.0) cropharvest = .true.
pstage = emerge + anth + pmat
```

- **wheatgrow** computes crop growth (`pgro`), root (`rdrym`) and dry matter (`drym`) for the Woodruff-Hammer dynamic wheat model.

On each day of the simulation, biomass accumulation is calculated from the minimum of radiation limited growth and water limited growth. Crop growth rate is partitioned and accumulated into dry matter and leaf area using a root:shoot ratio and leaf area ratio, both dependent on stage of development. LAI is further modified by water stress index. CERES (Jones and Kiniry, 1986) partitions biomass into individual leaves - number and area are calculated.
In the original version of PERFECT, biomass was calculated from the concept of water use efficiency (`wue`), the amount of biomass produced per millimetre of transpiration. Biomass accumulation is calculated as the product of `wue` and transpiration `ttrans`. This approach is most appropriate for environments where water is limiting. Under severe water-limiting conditions, biomass accumulation in a day may be negative indicating death of some plant material. An empirically based relationship described by Fischer (1979) is used to estimate crop growth rate.
Radiation limited yield is calculated from product of radiation use efficiency (`whruef`), (`1.0 - exp(-exc1*lai)`) and solar radiation (`rad`). where `exc1` is the extinct coefficient 1.
Littleboy et al. (1999)

Computes root growth:
`dwe = amin1(dwe + ddwe, dwemax)` where `ddwe` is the specified root growth constant.

Computes water and radiation limited growth (`pgro`):

```
wue = 10.2 - 1.3 * pan + 0.05 * pan * pan
pgro1 = wue * amin1(ttrans, gcov*pan)
pgro2 = whruef * gcov * rad
pgro = amin1(pgro1, pgro2)
```

Calculate temperature index (`tsi`) to modify transpiration from interpolating `tindex - temps` curve at `temp`.

Calculate water and nitrogen stress effects on growth (`pgro`):

```
wsix = amin1( (1.-(wsi1*.25)), 1.0)
nsix = amin1( (1.-(nsi2*.25)), 1.0)
pgro=pgro*amin1(wsix, nsix)
```

NB `nsi2` is not set anywhere.

Calculate root (`rgro`) and shoot growth (`sgro`):
Compute `srrat` from interpolating `srr - pstg` curve at `pstage`.

```
if(wsi1 > 0.5) srrat = srrat - 0.3 * wsi1
sgro = pgro * srrat
```

Temperature/popn responses are the "ramp" for early growth:

```
if(temp ≥ 18.0) sgro = amax1(sgro, 0.0125*popn*temp/25.0)
if(temp < 18.0) sgro = amax1(sgro, 0.007*popn*temp/18.0)
rgro = pgro - sgro
rgro = amax1(rgro, 0.012*popn*temp/20.0)
```

Calculate dry matter loss (`drymlos`) due to aging:

```
if(pstage < 1.5) drymlos = 0
if(pstage ≤ 2) drymlos = 0.002 * drym
if(pstage > 2) drymlos = 0.004 * drym
```

Accumulate dry matter (`drym`):

```
dryadd = sgro
drym = drym + drymadd - drymlos
rdrym = rdrym + rgro
```

- **wheatleaf** calculates leaf area index for the dynamic wheat model.

Calculate proportion of shoot growth to leaf area (`laippn`):

```
laippn = 0.015
if(pstage > 1.5) laippn = 0.05 - 0.0231 * pstage
laippn = tsi * laippn
```

Calculate water stress effect on leaf expansion:

```
selai = amax1(0.0, 1.0 - 3.0 * wsi1)
```

Calculate leaf loss (`lailos`) due to water or nitrogen stress:

```
lails1 = 0.0
lails2 = 0.0
if(pstage > 1.4 .and. wsi1 ≥ 0.10) lails1 = 0.1 * wsi1
if(nsi2 > 0.4) lails2 = lai * 0.0016 * nsi2
lailos = amax1(lails1, lails2)
```

NB `nsi2` is not set anywhere.

Calculate change in leaf area index (`dlai`):

```
if(pstage < 2.0) then
  dlai = (sgro * laippn * selai * (1.0 - nsi1)) - lailos
  dlai = amin1(dlai, 0.113)
else          Senescence of leaf area after anthesis
  if(tmm .eq. 0.0) tmm = pgdd + 10.0
  dlai = -1.0 * lai * ((temp-b23) / tmm) * (wsi1 + 1.0)
  tmm = tmm - (temp - b23)
endif
lai = amax1(0.0, lai + dlai)

if(pstage < 1.2) lai = drym * whlais
```

where `b23` is user defined base temperature (anthesis to maturity), `tmm` is degree days required until maturity and `whlais` is the leaf area coefficient.

The second of the above reduces LAI as a function of temperature and water stress. High temperature and high water stress will accelerate leaf senescence.

Calculate crop cover from `lai`:

PERFECT applies two extinction coefficients for crop cover for transpiration and potential soil evaporation. Crop cover (`ccov = 100 * (1 - exp(- exc * lai))`) is used in calculating potential soil evaporation and represents the proportion of incident energy intercepted by the crop. The proportion of crop cover effective for transpiration (`gcov = 100 * (1 - exp(- exc1 * lai))`) is greater than `ccov` to take account of sensible heat transfer from the soil back through the canopy. This increases potential transpiration when the soil surface is dry and actual soil evaporation is less than potential soil evaporation. Littleboy et al. (1999).

```
ccov = amax1((0.7*(1.0-exp(-drym/100.0))), (1.-exp(-exc*lai)))
gcov = amax1(amin1(0.03,2.*lai), 1.0-exp(-exc1*lai))
```

- **wheatyield** calculates yield for the dynamic wheat model.

Yield of wheat has been closely linked to crop growth potential over a short period around anthesis by Woodruff and Tonks (1983). The yield index is estimated from parameters summed over a 20-day period centred on the day of anthesis.
Hammer et al. (1987)

If anthesis (the time and process of budding and unfolding of blossoms) has not occurred (`anth < 1`), store last 10 days of `rain`, `temp`, `pan`, `ttrans` and `drym`.

If anthesis has occurred (`anth = 1.0`) - `cropanthesis = .true.`, sum last 10 days data of `temp`, `pan` and `ttrans`.

Add todays data to the summation if(icnt $<$ 20). Total for $\pm$ 10 days around anthesis.

Calculate wheat yield by 2 ways and take the smaller of the two:

```
if(cropharvest) then
  yp = strans / (span * amax1(11.5, stemp/19.0) )
  yielda = amax1(25.0, amin1(120.0, 0.2 * 0.1 * drymanthesis) ) + &
    (2507.0 * yp) + (121719.0 * yp * yp)
```

```
      yieldb = tdw10(10) * whyldb
      yield = amin1(yielda, yieldb)
    endif
```

where `span` = $\sum$ `pan`, `stemp` = $\sum$ `temp` and `strans` = $\sum$ `ttrans`. `tdw10 = drym`

`yp` is yield index, `drymanthesis` is total above-ground dry matter at anthesis.

NB When calculating `stemp`, 9.9 is added if `temp` is less than that.

The term incorporating `drymanthesis` was included to improve low-yielding crops (Hammer et al., 1987). The constants are those used in PERFECT Version 3.0 (Littleboy et al. 1999).
In PERFECT, `whyldb` = 0.87 and `drymanthesis` is used instead of `tdw10(10)`.

7 Type 5 / LAI Model

The LAI model uses a user defined LAI pattern over time with an optional water stress component. Planting, emergence, anthesis and harvest dates are all input by the user to define phenology.

THE LAI driven model uses functions described in Hammer et al. (1987) to estimate water stress and transpiration. Yield predictions are derived from Woodruff and Tonks (1983) for wheat, Hammer and Goyne (1982) for sunflower and Nix and Fitzpatrick (1969) for sorghum.
Littleboy et al. (1989).

laiMat

Reads crop parameters from `matcrop`.

Line	Variable	Description
2	iflag	Internal Flag (Do not change)
3	numlai	Number of crops (overstories+understories)
4	laimax	Potential maximum leaf area index (cm2/cm2)
5	phu	Total degree days oC from planting to harvest
6	dddylai	Day to reset hui
7	ddmnlai	Month to reset hui
8	plai	Proportion of growing season for maximum LAI
9	llap(1)	Proportion of Max LAI 1st point
10	dlap(1)	Proportion of growing season
11	llap(2)	Proportion of Max LAI 2nd point
12	dlap(2)	Proportion of growing season
13	ad	Senesence coefficient (cf EPIC)
14	rue	Radiation use efficiency (g/m2 per MJ of intercepted radiation)
15	EtPanHI	Harvest index (range between 0.0-1.0)
16	base	Base temperature (oC)
17	topt	Optimal temperature for plant growth (oC)
18	ddwe	Daily root growth (mm)
19	dwemax	Maximum root depth (mm)
20	irootfn	Root function 0=default 1=Ritchie 2=power law 3=proportional depth
21	KillLai(1)	Water stress threshold for crop death
22	KillLai(2)	Number of consecutive water stress days for crop kill

23	FrostLai(1)	temperature stress threshold for crop death
24	FrostLai(2)	Number of consecutive temperature stress days for LAI rate reduction
25	FrostLai(3)	Temperature stress lai reduction rate (=lai*frostlai(3) or =(#days-frostlai(2))**frostlai(3))
26	SoiwatLai(1)	Soil water deficit threshold to advance senescence
27	SoiwatLai(2)	Depth to calculate soil water deficit (mm)
28	SoiwatLai(3)	Number of consecutive soil water deficit stress days for LAI rate reduction
29	SoiwatLai(4)	Soil water deficit stress lai reduction rate (=lai*soiwatlai(4) or =(#days-soiwatlai(3))**soiwatlai(4))
30	MaxResidCover	Maximum Residue Cover (0-1)
31	ResRat	Use either -> Constant residual decay rate (if > 0 this option always used) (revised code)
32	ResTrg	or -> days since fallow to adopt decay resrd2 below
33	ResRd1	residual decay rate for days since fallow < restrg & when not in fallow (original code)
34	ResRd2	residual decay rate for days since fallow >= restrg (original code)
35	Ratoon	Ratoon crop (Y/N)
36	NumberRatoons	Number of ratoons
37	RatoonFactor	Growth scaling factor for ratoons
38	Crplng	Erosion flow length (m) (optional - else defined by soil descriptors)

nutrient data

39	idc	crop/landcover category(1=warm season annual legume 2=cold season annual legume 3=perennial legume 4=warm season annual 5=cold season annual 6=perennial 7=trees)
40	wsyfc	Value of harvest index between 0 and HVSTI which represents the lowest value expected due to water stress ((kg/ha)/(kg/ha))
41	biomjs	Biomass-energy ratio
42	cnyldc	Fraction of nitrogen in yield (kg N/kg yield)

43	cpyldc	Fraction of phosphorus in yield (kg P/kg yield)
44	rsdco	Decay rate constant
45	bn1	Nitrogen uptake parameter #1 - normal fraction of N in crop biomass at emergence (kg N/kg biomass)
46	bn2	Nitrogen uptake parameter #2 - normal fraction of N in crop biomass at 0.5 maturity (kg N/kg biomass)
47	bn3	Nitrogen uptake parameter #3 - normal fraction of N in crop biomass at maturity (kg N/kg biomass)
48	bp1c	Phosphorus uptake parameter #1 - normal fraction of P in crop biomass at emergence (kg P/kg biomass)
49	bp2c	Phosphorus uptake parameter #2 - normal fraction of P in crop biomass at 0.5 maturity (kg P/kg biomass)
50	bp3c	Phosphorus uptake parameter #3 - normal fraction of P in crop biomass at maturity (kg P/kg biomass)

Table 8: Crop parameters for LAI read in from `matcrop`.

```
llap(1) = llap(1)/100.0
llap(2) = llap(2)/100.0
dlap(1) = dlap(1)/100.0
dlap(2) = dlap(2)/100.0

rdwelai(1) = ddwe
rmaxlai(1) = dwemax
irotlai(1) = irootfn
covrlai(1) = maxresidcover

deathlai(1-6) = (killlai(1-2), frostlai(1-3), 0)
swatslai(1-5) = (soiwatlai(1-4), 0)
llapslai(1-2) = llap(1-2)
dlapslai(1-2) = dlap(1-2)
resudlai(1-4) = (resrat, restrg, resrd1, resrd2)
```

- **rootl** sets maximum rooting depth

  `rmaxlai = dwemax`

- **lairecd** transfer LAI model parameters to arrays `intcrops, relcrops, logcrops`.

- **rootd** as above

laimodel

Grow crop using the LAI model.

Call `laitransp, laiload, laistress, laileaf, laibiomass, roots`

- **laitransp** computes transpiration for LAI model:

  Initialise transpiration array (`trans(i) = 0`) and computes `mcfc(i) = sw(i) / fc(i)` and 0 ≤ `mcfc(i)` ≤ 1. `tranlai(1) = 0` and `wsi = 1`.

  Computes average `lai`:
  `lai =` ∑ `laia`

  Computes potential transpiration `pott`:

```
ccov = amin1(lai/3.0, 1.0)
ccov = amax1(0.0,ccov)
pott = amin1(pan*ccov, pan-se)
if(pott < 0.0) pott = 0.0
if(pott > pan) pott = pan
```

  Computes root penetration for each soil layer `dfac`:
  `call rootf` using `dwe, dwemax, irootfn` stored in `rootlai, rmaxlai, irotlai` respectively. Store `dfac(i)` in `rden`.

  Ensure ∑ `rden = 1`. If it is not, normalise it.

  Compute layer transpiration (`trans(i)`):
  `trans(i) = dfac(i) * pott`
  `tranlai(1) =` ∑ `dfac(i) * pott`
  which is limited by `sw(i)`: 0 ≤ `trans(i)` ≤ `sw(i)`.

  The transpiration is reduced if the total is greater than `pott`.
  If (`psup` > `pott`), `trans(i) = trans(i) * pott / psup` and
  `tranlai(1) = tranlai(1) * pott / psup` where `psup =` ∑ `trans(i)`

  Compute water stress index:
  `wsi = psup / pott` if `psup` ≤ `pott`.

  Reduce transpiration if total evapotranspiration > pan:
  That is (`ttrans + se`) > `pan` where `ttrans =` ∑ `trans(i)`.

```
trans(i) = trans(i) * tratio
tranlai(1) = tranlai(1) * tratio
```

  where `tratio = pan / (ttrans + se)`.
  `se = se * tratio`

- **laiload** loads LAI model parameters from arrays `intcrops, relcrops, logcrops`. Call `rootd, erosnls`.

Fit S-curve parameters:
Set `llap(1-2)` and `dlap(1-2)` from `llapslai(1-2)` and `dlapslai(1-2)`.
Call `scurve` - fits an S curve to two points. Derived from EPIC3270
Set `ah(1-2) = llap(1-2)`.

- **laistress** computes growth stress factors for temperature and water.

Compute temperature stress index:
Equation 2.235 of EPIC
`ratio = (temp - base)/(topt - base)`
`tsi = sin(0.5 * pi * ratio)`
and `0.0` $\leq$ `tsi` $\leq$ `1.0`.

Compute water stress index:

```
wsi = 1
if(pott > 0) wsi = ttrans / pott
```

and `0.0` $\leq$ `wsi` $\leq$ `1.0`.

Compute minimum stress factor:

```
reg = 1
if(tsi < reg) reg = tsi
if(wsi < reg) reg = wsi
```

and `0.0` $\leq$ `reg` $\leq$ `1.0`.

- **laileaf** computes LAI using major functions from EPIC.

Compute daily heat units:
`hu = (tmax + tmin) / 2.0 - base`

Compute sum of daily heat units:
`ahu = ahu + amax1(hu, 0)`

Compute heat unit index (eq 2.191 of EPIC):
`hui = ahu / phu`

Compute leaf growth if `hui < plai`:
Heat unit factor (eq 2.198 of EPIC):

```
huf = hui / (hui + exp(ah(1) - ah(2)) * hui)
dhuf = huf - hufp
hufp = huf
```

LAI (eq 2.197 of EPIC):

Eqn 2.197 originally stated that

dlai = dhuf * laimax * (1.0-exp(5.0*(laip-laimax)))* sqrt(reg)

This function NEVER allows lai to achieve laimax under no stress conditions due to the exponential term. This term was removed. Therefore, lai development is governed by the S-Curve, Max LAI, and stress factors only.

```
dlai = dhuf * laimax * sqrt(reg)
lai = lai + dlai
laip = lai
maxlai = mxlailai
maxlai = amax1(maxlai, lai)
mxlailai = maxlai
```

Compute leaf growth if `hui` $\geq$ `plai .and. hui` $\leq$ `1`:

Leaf senesence (eq 2.199 of EPIC):

```
lai = mxlailai * ( (1 - hui)/(1 - plai) ) ** ad
```

`lai` $\geq$ `0.`

Compute frost stress index:

```
if(tmin < deathlai(3)) then
  deathlai(6) = deathlai(6) + 1.0
else
  deathlai(6) = 0.0
endif
```

NB `deathlai(3) = frostlai(1)`

When the number of sequential frost events (`deathlai(6)`) exceed a user-defined index, reduce lai by a rate function (power function based on number of sequential days).

```
if(deathlai(6) > deathlai(4)) then
  dlaif = lai * (deathlai(6)**deathlai(5))
  dlaif = lai * deathlai(5)
else
  dlaif = 0.0
endif
```

Compute senescence soil moisture trigger (`swatslai`):

Summing `swd` to depth `swatslai(2)`:

```
swd = ∑ (fc(i) - sw(i))
```

```
if(swd < swatslai(1)) then
  swatslai(5) = swatslai(5) + 1
else
  swatslai(5) = 0
endif
```

When the number of sequential soil water limiting days (`swatslai(6)`) exceed a user-defined index, reduce lai by a rate function (power function based on number of sequential days).

```
if(swatslai(5) > swatslai(3)) then
  dlais = lai * (swatslai(5)**swatslai(4))
  dlais = lai * swatslai(4)
else
  dlais = 0.0
endif

dlai = max(dlaif, dlais)
```

`0 ≤ dlai ≤ lai.`

```
if(hui ≥ plai  .and.  hui ≤ 1.0) mxlailai = mxlailai -dlai

laia = lai - dlai
```

- **laibiomass** computes biomass using EPIC functions.

Compute total leaf area (`sumlai`) of all species:

`sumlai` = $\sum$ `laia`

Compute intercepted radiation (`par`) for each species:

`par = 0.5 * rad * (lai/sumlai) * (1.0 - exp(-.65*lai))`
assuming 50% solar radiation, extinction coefficient of 0.65 and radiation is partitioned by LAI.

Compute daylength factor (`dhrlt`):

```
dhrlt = daylen() - hrltp
if(hrltp < 0.01) dhrlt = 0
hrltp = daylen()
```

Compute biomass accumulation (eq 2.193 of EPIC):

```
drym = drymlai
dryplus = reg * par * rue * (1.0 + dhrlt)**3.0
drymadd = drymadd + dryplus
drym = drym + dryplus
drymlai  = drym
```

- **roots** computes linear root growth

```
dwe = dwe + ddwe
```

$0 \leq$ `dwe` $\leq$ `dwemax`.

8 Type 10 / Native Pasture Model

natMat

It reads crop parameters for model ET:PAN from `matcrop`.

Line	Variable	Description
2	iflag	Internal flag (do not change)
3	grtmin	Minimum temperature (grtmin) - originally 0
4	grtmax	Maximum temperature (grtmax) - originally 30
5	grtopt	Optimum temperature (grtopt) - originally 12
6	grtemn	Temperature n (grtemn) [0 1]
7	grpnet	Saturated canopy radiation Pnet (MJ/m^2/day) (grpnet)
8	grslar	Specific leaf area (grslar)
9	grlexc	Light extinction coefficient (grlexc)
10	grcow1	Decay coefficient Alphaf between live and dead (grcow1)
11	grcow4	Decay coefficient Alphad between dead and litter (grcow4)
12	grgeff	Growth efficicency coefficient [0-1] (grgeff) (
13	grleaf	Growth partition to leaf [0-1] (grleaf) - remainder to stem
14	grilwt	Initial live dry weight (t/ha) (grilwt)
15	gridwt	Initial dead dry weight (t/ha) (gridwt)
16	dmtmin	Minimum live dry weight to calculate GAI (t/ha) (dmtmin)
17	grfert	Fertility [0-1]
18	crplng	Erosion flow length (m)
19	srdses	Stocking rate (dse)
20	igrdor(ipastr)	Pasture dormancy (0=not summer dormant 1=summer dormant)
21	igrphs(ipastr)	Pasture phase (1=vernalising 2=vegetative 3=reproductive 4=flowering 5=post-reproductive)
22	shootnt(ipastr-1)	Initial shoot mass (kg/ha) Immature
23	shootnt(ipastr-2)	Initial shoot mass (kg/ha) Mature
24	shootnt(ipastr-3)	Initial shoot mass (kg/ha) Senescence
25	stemsnt(ipastr-1)	Initial stem mass (kg/ha) Immature

26	stemsnt(ipastr-2)	Initial stem mass (kg/ha) Mature
27	stemsnt(ipastr-3)	Initial stem mass (kg/ha) Senescence
28	deadlnt(ipastr)	Initial dead/litter mass (kg/ha)
29	rootsnt(ipastr)	Initial root mass (kg/ha)
30	mulchnt(ipastr)	Initial mulch mass (kg/ha)
31	allocnt(ipastr-1-1)	Vernalising Ashoot
32	allocnt(ipastr-1-2)	Vernalising Astem
33	allocnt(ipastr-1-3)	Vernalising Aroot
34	allocnt(ipastr-1-4)	Vernalising Aseed
35	allocnt(ipastr-2-1)	Vegetative Ashoot
36	allocnt(ipastr-2-2)	Vegetative Astem
37	allocnt(ipastr-2-3)	Vegetative Aroot
38	allocnt(ipastr-2-4)	Vegetative Aseed
39	allocnt(ipastr-3-1)	Reproductive Ashoot
40	allocnt(ipastr-3-2)	Reproductive Astem
41	allocnt(ipastr-3-3)	Reproductive Aroot
42	allocnt(ipastr-3-4)	Reproductive Aseed
43	allocnt(ipastr-4-1)	Flowering Ashoot
44	allocnt(ipastr-4-2)	Flowering Astem
45	allocnt(ipastr-4-3)	Flowering Aroot
46	allocnt(ipastr-4-4)	Flowering Aseed
47	allocnt(ipastr-5-1)	Post-Reproductive Ashoot
48	allocnt(ipastr-5-2)	Post-Reproductive Astem
49	allocnt(ipastr-5-3)	Post-Reproductive Aroot
50	allocnt(ipastr-5-4)	Post-Reproductive Aseed
	root growth, max root depth & light extinction coeff	
51	psgwth(ipastr)	Daily root growth (mm) (psgwth() -> ddwe)
52	psdpth(ipastr)	Maximum root depth (psdpth() -> dwemax)
53	igrrfn(ipastr)	Root function 0=default 1=Ritchie 2=power law 3=proportional depth (should be same as Kr3 below)
54	pslexc(ipastr)	Light extinction coefficient (pslexc() -> grlexc)
	species decay parameters	
55	dcay1nt(ipastr)	Decay rate 1 (dcay1nt()) Immature -> Mature
56	dcay2nt(ipastr)	Decay rate 2 (dcay2nt()) Mature -> Senescence
57	dcay3nt(ipastr)	Decay rate 3 (dcay3nt()) Senescence -> Dead
58	dcay4nt(ipastr)	Decay rate 4 (dcay4nt()) Dead -> Null
	triggers to advance onset of senescence	
59	sencwnt(ipastr)	Soil water deficit (mm) re:onset of senescence (immature and mature -> senescence) At onset force isumdr=1

60	sencdnt(ipastr)	Depth (mm) to calculate onset of senescence (immature and mature -> senescence)
61	isencdy(ipastr)	Number of days at trigger temperature (immature and mature -> senescence)
62	senctnt(ipastr)	Trigger temperature (immature and mature -> senescence)
63	sencfnt(ipastr)	Multiplier for decay rate 3 during triggered senescence when isenc=1
64	partfnt(ipastr)	Shoot:stem partion ratio
65	basalar	NPP basal area coefficient

Coefficients

66	igrcod	Pasture type (1=perennial 2=annual) - igrtyp
67	igrcyc	Trigger (0 adopts day length trigger based on Kv4 whereas 1 adopts degree-day control based on Kv5)
68	Kv1	Kv1 - Vernalisation rate at 0oC (Phenology)
69	Kv2	Kv2 - Effect of temperature on vernalisation rate oC-1 (Phenology)
70	Kv3	Kv3 - Base temp for degree-day computations (Phenology)
71	Kv4	Kv4 - Daylength (h) for commencement of reproductive phenostage (Phenology)
72	Kv5	Kv5 - Degree-day sum for commencement of reproductive phenostage (Phenology)
73	Kv6	Kv6 - Degree-day sum for commencement of flowering - Annuals(Phenology)
74	Kv7	Kv7 - Maximum length of flowering period - Annuals (Phenology)
75	Kv8	Kv8 - Effect of soil moisture stress on flowering duration (days)-Annuals(Phenology)
76	Kv9	Kv9 - Degree-day sum beyond which reproductive phenostage can end (Phenology)
77	Kv10	Kv10 - Available water threshold below which reprod phenostage can end (Phenology)
78	Kv11	Kv11 - Tlag temperature below which summer dorm. phenostage can end(Phenology)
79	Kv12	Kv12 - ASW threshold above which summer dorm. phenostage can end (Phenology)
80	Kv13	Kv13 - Initial duration of cool moist cond's required to break summer dorm.(Phenology)

81	Kv14	Kv14 - Days required for summer dormancy to reduce to zero - summer dorm.(Phenology)
82	Kv15	Kv15 - Glfsoil [0-1] for commencement of reproductive phenostage (re:Kv4 Kv5) (Phenology)
83	Kv16	Kv16 - Number of days Kv15 for commencement of reproductive phenostage (re:Kv4 Kv5) (Phenology)
84	Kd1	Kd1 - Death rate between immature and mature pools during frosts (day-1)(Death)
85	Kd2	Kd2 - Temperature parameter for 5% mortality at the first frost (oC)(Death)
86	Kd3	Kd3 - Temperature parameter for 95% mortality at the first frost (oC) (Death)
87	Kd4	Kd4 - Frost hardening factor(Death)
88	Kf1l	Kf1l - Rate of fall of standing dead due to precipitation(Death *)
89	Kf2l	Kf2l - Curvature of relationship between fall rate and precipitation(Death *)
90	Kf3l	Kf3l - Specific rate of fall of standing dead due to trampling(Death *)
91	Ka1	Ka1 - Target root:shoot ratio during vegetative growth (not used when allocation table=1) (Allocation)
92	Ka2	Ka2 - Target root:shoot ratio after flowering(Allocation)
93	Ka3	Ka3 - Maximum allocation to reproductive structure (not used when allocation table=1) (Allocation)
94	Ka4	Ka4 - Maximum relative growth rate of shoots during reproductive and flowering (Allocation)
95	Ka5	Ka5 - Maximum relative growth rate of shoots during post-reproductive (Allocation)
96	Ka6	Ka6 - Maximum relative growth rate of roots during summer dormancy (not used when allocation table=1) (Allocation)

97	Kr1	Kr1 - Specific root loss rate at 10oC [0-1] approx equilivant to daily baseline fraction reduction (Allocation)
98	Kr2	Kr2 - Q10 for root loss [0-1] (Allocation)
99	Kr3	Kr3 - Root distribution funtion irootfn (0=default 1=Ritchie 2=power 3=proportional) (Roots)
100	Kr4	Kr4 - Root distribution power function beta value when Kr3=2 (Roots)
101	Kt1	Kt1 - Lower temperature for 5% of maximum growth (average daily) (Allocation)
102	Kt2	Kt2 - Lower temperature for 95% of maximum growth (Allocation)
103	Kt3	Kt3 - Upper temperature for 95% of maximum growth(Allocation)
104	Kt4	Kt4 - Upper temperature for 5% of maximum growth(Allocation)
105	Kw1	Kw1 - Average soil moisture ASW threshold for water use(Allocation)
106	Kw2	Kw2 - Average soil moisture ASW threshold for growth(Allocation)
107	Kw3	Kw3 - WFPS(water filled pore space) threshold for waterlogging(Allocation)
108	Kw4	Kw4 - The curvature of growth limitation by waterlogging(Allocation)
109	Ku1	Ku1 - Biomass threshold for remobilisation for underground reserves (kg/ha) (Remobilisation)
110	Ku2	Ku2 - Relative rate of remobilisation (day-1)(Remobilisation)
111	Ku3	Ku3 - Growth limiting factor (soil water) for remobilisation for underground reserves (Remobilisation)
112	Ku4	Ku4 - Growth limiting factor (temp) for remobilisation for underground reserves (Remobilisation)
113	Ki1	Ki1 - Ratio of green area index to leaf weight(Danth = 0.008)(Transpiration)
114	Ki3	Ki3 - Radiation use effiiency at a referenced solar gMJ-1(Transpiration)
115	Ki4	Ki4 - Radiation intensity on radiation use efficiency(Transpiration)

116	Ki5	Ki5 - Ratio of area index to stem weight(Transpiration)
117	Ki6	Ki6 - Ratio of area index to dead and litter weight (0.026) [NOT USED](Transpiration)
118	Kt5	Kt5 - Minimum temperature (grtmin) - originally 0 (Decay function)
119	Kt6	Kt6 - Optimum temperature (grtopt) - originally 12 (Decay function)
120	Kt7	Kt7 - Temperature n (grtemn) [0 1] (Decay function)
121	Kt8	Kt8 - Glftemp multiplication factor used when t>topt (Decay function)
122	Kx1	Kx1 - Minimum available percent soil water [0-1] (Glfsoil function)
123	Kx2	Kx2 - Optimum available percent soil water [0-1] (Glfsoil function)
124	Kx3	Kx3 - Available soil water shape parameter [0 1] (Glfsoil function)
125	Kx4	Kx4 - Decay soil moisture multiplier [must be > 1.0 0.0 = turned off] (Glfsoil function)
126	Kx5	Kx5 - Multiplication/weighting factor (Glfsoil function)
127	Kb1	Kb1 - Basal area coefficient (Basal area function)
128	Kb2	Kb2 - Basal area exponent (Basal area function)
129	Kc1	Kc1 - Conversion factor for cover and LAI (t/ha) (Cover and lai functions)

Table 9: Crop parameters for native read in from `matcrop`.

Some of the above parameters are constrainted/changed by the following:

- `dmtmin` $\geq 10^{-5}$
- `grfert` $\geq 10^{-5}$
- `srdses` ≥ 0.0
- Convert minimum dry weight (`dmtmin`) from t/ha to g/m^2
  `dmtmin = dmtmin * 1000.0 / 10.0`
- Convert shoot (`shootnt`) and stem (`stemnt`) mass from kg/ha to g/m^2
  `shootnt(1-3) = shootnt(1-3) / 10.0`,
  `stemnt(1-3) = stemnt(1-3) / 10.0`
  `deadlnt = deadlnt / 10.0`,
  `rootsnt = rootsnt / 10.0` and
  `mulchnt = mulchnt / 10.0`.
- `dwemax = psdpth`

- `irootfn = igrrfn`
- Shoot:stem partition ratio: `0.0` ≤ `partfnt` ≤ `1.0`
- NPP basal area: `0.0` ≤ `basalar` ≤ `1.0` and `basalnt = basalar`.
- Set harvest/cut store (`cutdmnt`), maximum temperature index (`phentx`), `uroot` and `delroot` to zero.
- Frost decay parameter: `0.0` ≤ `grxkd1` ≤ `1.0`
- Root mobilisation parameter 1: `0.0` ≤ `grxkr1` ≤ `1.0`
- Root mobilisation parameter 2: `0.0` ≤ `grxkr2` ≤ `1.0`
- Decay soil multiplier parameter (`grxkx4`) set to 0.0 or 2.0 if it is less than 10^{-6} or 1.0 respectively.
- Convert `grxku1` from kg/ha to g/m^2 - `grxku1 = grxku1 / 10.0`.
- Radiation use efficiency

```
if(grxki3(ipastr) > 25.0) then
  xnpp = grxki3(ipastr)/10.0
  val1 = (1.67 + grxki4(ipastr))/((20.0/12.0) + grxki4(ipastr))
  xki3 = xnpp/(val1*20.0)
  grxki3(ipastr) = xki3
endif
```

- Calculate basal area coefficient (`basalnt`)

```
basalar = basalnt(ipastr)
phxkb1 = grxkb1(igrcol(ipastr))
phxkb2 = grxkb2(igrcol(ipastr))
if(phxkb1 > 0.01 .and. phxkb2 > 0.01) then
  basalnt(ipastr) = phxkb1 * (basalar**phxkb2)
else
  basalnt(ipastr) = (1.0 - exp(-12.0 * basalar))**4.0
endif
```

- Set `ivernl, ivegst, ireprd, iflowr, isumdr, isencr, igermn, iherbm, iforcm` to zero.
- Set initial phenostage:
- Set phenology arrays (`daydor, dayrep, daysfl, dayssd, temlag, phenmv, phendd, phenfl, phensd, phenfh, phengi, phensi, glfsnt`) to zero.
- `if(igrcyc == 0)`, use day length trigger based on Kv4 (`grxkv4`).
- `if(igrcyc == 1)`, use degree day control based on Kv5 (`grxkv5`).
- Initialse curvature (`xn = grtemn`).

Call `rootl`, `rootd`, `erosnls`.

```
psdpth(ipastr) = dwemax
psdwes(ipastr) = 0.0
```

- **rootl** set maximum root length `dwemax`
- **rootd** calculates maximum root depth using `smxxx` of last soil layer. Calls `soilchara`, `soilextn`, `soilcharb`, `initialsw`.
- **erosnls** calculates erosion flow length.

nattempfn

This subrountine seems to do nothing useful.

natinital

```
grw3 = grw2 = grw1 = grilwt * 1000.0 / (10000.0 * 3.0)
grw4 = grilwt * 1000.0 / 10000.0

grpnet = grpnet / 1000.0
xnet = grpnet
reflight = 30.0
reftempr = 20.0
xlai = grslar * (grw1 + grw2 + grw3)
wgp = grw1 + grw2 + grw3 + grw4
drywght = wgp * 10.0
wghtliv = (grw1 + grw2 + grw3) * 10.0
wghtkil = grw4 * 10.0
grwth = 0.0
gract = 0.0
gwthrate = 0.0
gwthlive = 0.0
gwthdead = 0.0
glftemp = 0.0
glfsoil = 0.0
temp = reftempr

prvtemp = 0.0
prvdayl = 48.0

aroots=1.0
ashoot=0.0
aseeds=0.0
aleafs=0.0
```

```
grw1 = ∑ shootnt(i)
grw2 = ∑ stemsnt(i)
grw3 = deadlnt
grw4 = rootsnt

orgshoot(1-3) = shootnt(1-3)
orgstems(1-3) = stemsnt(1-3)
orgdeadl(1-3) = deadlnt(1-3)
orgroots(1-3) = rootsnt(1-3)
```

natives

Call natdaytmp, nattransp, natlimits, natharvst, rootf, natphenol, roots, natgrowth, natallocs, natremobs, natdigest, nattissue.

```
fallowphase = .false.
cropphase = .false.
dotillage = .false.
plantwheat = .false.
plantsunflower = .false.
plantlai = .false.
plantetpan = .false.
plantgrass = .false.
plantgrasg = .false.
plantgrasn = .false.
planttrees = .false.
```

- **natdaytmp** calculates daily temperature and other climate data.

```
      pi = 4.0 * atan(1.0)

      xlamda =  - 36.0 * 2.0 * pi/360.0

!     calculate declination of the sun

      sundec = (23.5 * pi/180) * cos(2.0 * pi * (dayno - 173)/365)

!     calculate daylength

      xval1 = cos(109.0 * pi/216)
      xval2 = sin(xlamda) * sin(sundec)
      xval3 = cos(xlamda) * cos(sundec)
      daylng = 24.0 * acos((xval1 - xval2)/xval3)/pi

!     calculate mean temperature during daylight hours

      xval1 = (daylng + 3.52)/daylng
```

```
xval2 = cos(0.17 * pi/(daylng + 3.52))
xval3 = cos(((daylng - 0.17) * pi)/(daylng + 3.52))
xval4 = (xval2 - xval3)/pi
tday = tmin + ((tmax - tmin) * xval1 * xval4)
```

- **nattransp** computes transpiration for pasture model.

Estimate evaporative demand: `evap0 = 0.8 * pan`

Initialise transpiration array (`trans(i) = 0`) and computes `mcfc(i) = sw(i) / fc(i)` and 0 ≤ `mcfc(i)` ≤ 1.

`pstran(1) = 0.0` and `wsi = 1.0`.

Compute % green active cover (`gcov`), total crop cover (`ccov`) and litter and mulch cover (`covm`)

```
gcov=0.0
ccov=0.0
covm=0.0
do ipastr=1,npastr
  phxki6=grxki6(igrcol(ipastr))
  phxkc1=grxkc1(igrcol(ipastr))

  xgaii=psgaie(ipastr)
  lai=pslais(ipastr)

  gcov=gcov+(xgaii*phxkc1)
  ccov=ccov+(lai*phxkc1)
  covm=covm+(mulchnt(ipastr)*phxki6)
enddo
```

0 ≤ `gcov` ≤ 1.0, 0 ≤ `ccov` ≤ 1.0, 0 ≤ `covm` ≤ 1.0.

Compute potential transpiration

Sum of product of GAI and a composite Kc1 coefficient.

Compute composite LAI (`complai`) and Kc1 (`compkc1`)

```
    xsumshoot = ∑ shootnt(i)
    xsumstems = ∑ stemsnt(i)
    xsumdeadl = deadlnt
xsumtotal = xsumshoot + xsumstems + xsumdeadl
    specshoot = ∑ shootnt(i)
    specstems = ∑ stemsnt(i)
    specdeadl = deadlnt
spectotal = specshoot + specstems + specdeadl
if(xsumtotal > 10^-4) then
   specfracs = spectotal / xsumtotal
else
   specfracs = 0.0
```

```
endif
lai = pslais
complai = specfracs * lai
phxkc1 = grxkc1
compkc1 = ∑ specfracs * phxkc1
```

xsumtotal ≥ dmtmin, 0.0 ≤ specfracs ≤ 1.0.

Redefine LAI

```
lai = complai
```

sumgai

```
    sumgai = ∑ psgaie
acov = sumgai * compkc1
```

0.0 ≤ acov ≤ 1.0.

Calculate potential transpiration based on green active cover

```
pott = amin1(acov * pan, pan - se)
```

0.0 ≥ pott.

Calculate root penetration per layer (dfac)

```
do j=1, npastr
  dwe = psdwes(j)
  dwemax = psdpth(j)
  irootfn = igrrfn(j)
  phxkr3 = grxkr3(igrcol(j))
  phxkr4 = grxkr4(igrcol(j))
  irootfn = nint(phxkr3)
  rootspw = phxkr4
  call rootf
  do i=1, ndeps
    rden(j,i) = dfac(i)
  enddo
enddo
```

- **natlimits** computes plant stage (pstage).
- **natharvst** computes plant stage (pstage).
- **rootf** computes plant stage (pstage).
- **natphenol** computes plant stage (pstage).

- **roots** calculates linear root growth.

  ```
  dwe = dwe + ddwe
  ```

  $0 \leq$ `dwe` $\leq$ `dwemax`

- **natgrowth** computes plant stage (`pstage`).
- **natallocs** computes plant stage (`pstage`).
- **natremobs** computes plant stage (`pstage`).
- **natdigest** computes plant stage (`pstage`).
- **nattissue** calculates transpiration for ET:PAN model.

References

Beverly, C., 2007. Technical Manual - Models of the Catchment Analysis Tool. Victoria Department of Sustainability and Environment. FILE.

Feikema, P., Morris, J., Beverley, C., Baker, T. and Lane, P., 2010. Description of the 3PG+ forest growth model. Department of Forest and Ecosystem Science, The University of Melbourne. FILE.

Hammer, G.L., Woodruff, D.R. and Robinson, J.B., 1987. Effects of climatic variability and possible climatic change on reliability of wheat cropping - a modelling approach, *Agriculture, Forestry & Meteor.*, **41**, 123-142. FILE.

Littleboy, M., Silburn, D.M., Freebairn, D.M., Woodruff, D.R. and Hammer, G.L., 1989. PERFECT - A Computer Model of Productivity Erosion Runoff Functions to Evaluate Conservation Techniques, *Queensland Department of Primary Industries, Brisbane, Australia*. FILE.

Littleboy, M., Freebairn, D.M., Silburn, D.M., Woodruff, D.R. and Hammer, G.L., 1999. PERFECT Version 3.0, 1-52. FILE

Neitsch, S.L., Arnold, J.G., Kiniry, J.R., Williams, J.R. and King, K.W., 2002. SWAT2000 Theoretical Documentation. FILE.

O'Leary, G.J. and Connor, D.J., 1996. A simulation model of the wheat crop in response to water and nitrogen supply: I. Model construction, *Agricultural Systems*, **52**, 1-29. FILE.

Porter, J.R. and Gawith, M., 1999. Temperatures and the growth and development of wheat: a review, *European Journal of Agronomy*, **10**, 23-36. FILE.

Stockle, C.O., Donatelli, M. and Nelson, R., 2003. CROPSYST, a cropping systems simulation model, *Eur. J. Agron.*, **18**, 289-307. FILE.

Williams, J.R., Jones, C.A., Kiniry, J.R. and Spanel, D.A., 1989. The EPIC crop growth model, *Transactions - American Society of Agricultural Engineers*, **32**, 497-511. FILE.

9 Appendix

Hammer and Woodruff

Littleboy et al.