DDRules Technical Documentation

As at Apsim Version 7.4 (revision 1472)

Scott Rains

18th April 2011

Table of Contents

[Introduction 5](#_Toc299098834)

[Grazing Management Implementation / Model 5](#_Toc299098835)

[Apsim Event Handlers 5](#_Toc299098836)

[OnPrepare 5](#_Toc299098837)

[OnProcess 5](#_Toc299098838)

[Paddock pasture masses/covers 6](#_Toc299098839)

[Pasture Quality 6](#_Toc299098840)

[Allocation of grazing paddocks 7](#_Toc299098841)

[Grazing Residual 7](#_Toc299098842)

[Grazing Rotation Length 8](#_Toc299098843)

[Removing Grazing Paddocks 8](#_Toc299098844)

[Grazing Management: DDRules vs. Industry “Best Practice” 8](#_Toc299098845)

[Alternative Option [LUDF] 9](#_Toc299098846)

[Introduction 9](#_Toc299098847)

[Parameters 9](#_Toc299098848)

[Related Model Outputs 10](#_Toc299098849)

[Allocation Process (weekly) 10](#_Toc299098850)

[Pasture Conservation 10](#_Toc299098851)

[Silage cutting rule parameters 10](#_Toc299098852)

[Key points 10](#_Toc299098853)

[Potential improvements 11](#_Toc299098854)

[Special Case: Surplus Removal from Feed Wedge 11](#_Toc299098855)

[Supplementary Feeding 11](#_Toc299098856)

[Dairy Cow Model 13](#_Toc299098857)

[Introduction 13](#_Toc299098858)

[Key Concepts 13](#_Toc299098859)

[Energy Requirements 13](#_Toc299098860)

[Animal Intake / Grazing 14](#_Toc299098861)

[Removal: Set stocking 14](#_Toc299098862)

[Removal: Break feeding 14](#_Toc299098863)

[Example 14](#_Toc299098864)

[Nutrient Partitioning 15](#_Toc299098865)

[Summary 15](#_Toc299098866)

[Nitrogen to excreta 15](#_Toc299098867)

[Dung 15](#_Toc299098868)

[Urine 16](#_Toc299098869)

[Nutrient Returns 16](#_Toc299098870)

[Dung 16](#_Toc299098871)

[Urine 16](#_Toc299098872)

[Assumptions 16](#_Toc299098873)

[References 17](#_Toc299098874)

[Additional Dairy Farm Operation 18](#_Toc299098875)

[Introduction 18](#_Toc299098876)

[Dairyshed Effluent Return (WIP) 18](#_Toc299098877)

[Settable values 18](#_Toc299098878)

[Operation 18](#_Toc299098879)

[Return to Named Paddock [EffluentPaddocks] 18](#_Toc299098880)

[Return to Percentage of Farm [EffluentPaddocksPercentage] 18](#_Toc299098881)

[Important Implementation Notes 18](#_Toc299098882)

[Laneways 18](#_Toc299098883)

[Settable values 18](#_Toc299098884)

[Operation 19](#_Toc299098885)

[Notes 19](#_Toc299098886)

[Whole Farm Irrigation 19](#_Toc299098887)

[Settable values 19](#_Toc299098888)

[Example Event 19](#_Toc299098889)

[Related Outputs 19](#_Toc299098890)

[Whole Farm Fertiliser 19](#_Toc299098891)

[Settable values 19](#_Toc299098892)

[Example Event Call 19](#_Toc299098893)

[Related Outputs 20](#_Toc299098894)

[Appendix 20](#_Toc299098895)

[Core Model/Code Structure 20](#_Toc299098896)

[Core classes 20](#_Toc299098897)

[Paddock 21](#_Toc299098898)

[Farm 21](#_Toc299098899)

[Biomass 21](#_Toc299098900)

[FeedStore 21](#_Toc299098901)

[SimpleCow / SimpleHerd 21](#_Toc299098902)

[Grazing Paddock Allocation Process 22](#_Toc299098903)

[Common Terminology 23](#_Toc299098904)

# Introduction

DDRules is an Apsim module that allows users to graze a multi-paddock Apsim simulation as a dairy farm. It implements a number of management options to allow the simulation of a “typical” New Zealand dairy farm. Its core purpose is to manage the grazing of pasture plants in a multi-paddock Apsim simulation. Additional functionality is provided for managing feed demand and supply through additional feed, pasture conservation and on farm demand.

A very simplistic dairy cow model is also provided. This model’s core function is to generate feed demand, pasture removal and nutrient return.

# Grazing Management Implementation / Model

DDRules implements a rules based approach to the grazing management or a typical New Zealand dairy farm as proposed by Dawn Dally of DairyNZ. At its centre this model uses two key parameters to control the grazing of the milking platform, grazing residual and rotation length. These two parameters are commonly understood and used by dairy farmers and herd managers in the New Zealand dairy industry.

In addition to this a number of additional controls are implemented to reflect the other key aspects of dairy farm management. These relate to management of pasture deficit (supplementary feeding etc.), pasture surplus / conservation and herd demand (wintering off, stocking rate etc.)

Core Concepts;

* Paddocks ranked by available dry matter.
* Paddocks arranged in a queue, with greatest mass first.
* Queued paddocks grazed sequentially.
* Paddocks re-ranked when queue is empty (complete rotation) or grazing rules are changed.
* Grazing rules based on post-grazing residual and rotation length.
* Paddocks can be break fed.
* Paddocks can be closed and cut for silage.
* Silage or purchased feed can be used to meet animal demand.
* Dry cows can be wintered off farm.
* Only milking platform is modelled i.e. no calves or heifers grazed on farm.

## Apsim Event Handlers

### **OnPrepare**

1. Reset all temporary output data (paddocks and animals)

### **OnProcess**

1. Retrieve paddock covers/pasture masses
2. Calculate animal/herd demand for the day
3. If no paddocks in the rotation then allocate paddocks
4. Graze animals / Pasture removal
5. Post grazing;
   1. Feed silage and/or supplements (if required)
   2. Partition nitrogen
   3. Return nutrients
6. Do pasture conservation

## Paddock pasture masses/covers

DDRules currently has a partial reliance on interacting with paddocks containing AgPasture although it can function correctly with all Apsim plant types that are able to return green and senescent leaf and stem.

The plant/pasture related variables that DDRules relies on are;

Basic plant biomass variables required;

* leafgreenwt
* stemgreenwt
* leafsenescewt
* stemsenescewt

These values are summed, converted (from g/m2 to kg/ha) by the Biomass class. These values are used to calculate total above ground pasture mass and pasture quality.

AgPasture specific variables;

* AboveGroundNPct (used calculate nitrogen content of diet)
* HerbageGrowthWt (used to predict pasture growth over grazing interval)

In paddocks not containing AgPasture a default nitrogen content is assumed (currently 3.5%) and no prediction of pasture growth is used when calculating the grazing rotation. The default nitrogen concentration is currently specified under the assumption that a pasture type plant is being grazed. At this stage DDRules does not sanity check what plant species it is being asked to graze and therefore will allow grazing of ANY of the Apsim plant types under the assumption they are pasture e.g. cotton, rice or even sugar cane will be grazed without regard to the true physical makeup of the species.

### Pasture Quality

Two quality related measures are used by the model, metabolizable energy (ME/kgDM) and digestibility. The former dictates the amount of energy available to the grazing animal, the latter the amount of consumed dry matter returned as excreta.

In cases where no ME value can be obtained from the plant species ME is calculated using a plant component based calculation as implemented in the QGraze model (reference required) e.g.

MEtotal = DMGreen Leaf x 12 + DMGreen Stem x 10.5 + DMDead x 9

MEkg/DM = METotal / DMTotal

If no value for digestibility can be obtained, a default value of 68% is used (reference required)

## Allocation of grazing paddocks

At its simplest level, DDRules ranks paddocks by total above ground biomass and then grazes paddocks with the greatest mass first. This process is managed via the queuing system in order to graze all paddocks in the simulation where possible. To manage this the queue/allocation is reconstructed when either;

1. A full farm rotation has been made i.e. the available paddock queue is empty or
2. The grazing rules have been changed e.g. either the rotation length or grazing residual parameters have been modified.

Allocation process;

1. All paddocks are sorted by total cover (highest mass first)
2. Paddocks are added to a queue if they are not currently closed for conservation (refer x.x) and are considered grazable (refer x.x)
3. A “number of days to graze” paddock level variable is calculated based on rotation length (refer x.x)

Key assumptions;

* All paddocks are grazed to the same residual i.e. paddocks with the highest mass are to be grazed with highest priority. This assumption holds true for an all pasture system, but will need to be revisited when forage crops etc. are added to the system.
* Paddocks not included in the queue i.e. either close for conservation or removed from rotation at the time of allocation will not be added back into the rotation until the next allocation process. The assumption will hold true until forage crops etc. are added or when the user would like to “clean up” paddock that have been conserved.

### Grazing Residual

The grazing residual parameter sets the post-grazing pasture residual i.e. the pasture cover that animals will graze down too. This is used to calculate the quantity of dry matter in the paddock that is available to the herd for grazing.

e.g.

Pre-graze cover 3500kgDM/ha (above ground dry matter (leaf and stem))

Less Grazing Residual 1500kgDM/ha

Available Feed = 2000kgDM /ha

### Grazing Rotation Length

The grazing rotation length is used to control how long animals take to graze the entire farm. Where possible all paddocks available for grazing will be grazed during a given rotation. This assumes that the growth potential of all paddocks is relatively equal and therefore best managed using the same return period.

E.g. 20 paddock farm;

* 10 day rotation = 0.5 days per paddock (or 2 paddocks per day)
* 20 day rotation = 1 day per paddock
* 40 day rotation = 2 days per paddock

### Removing Grazing Paddocks

DDrules has the ability to take paddocks out of the current grazing rotation to allow them to be used for another purpose e.g. forage cropping.

Syntax

DDRules.PaddockGrazable([Paddock Number]) = [Switch]

Where

Paddock Number = Position of paddock in the simulation (zero based)

Switch = Integer representation of a Boolean e.g. 0 = false, 1 = true

Examples

DDRules.PaddockGrazable(2) = 0 - set paddock 2 as not being available for grazing

DDRules.PaddockGrazable(2) = 1 - set paddock 2 as being available for grazing

### Grazing Management: DDRules vs. Industry “Best Practice”

Reference: DairyNZ Farmfact: Pasture feed wedges (1-14) & 1-12, 1-13a, 1-13b Using the Spring Rotation Planner How to Allocate Feed.

The New Zealand dairy industry relies heavily on good pasture management. The ability to balance feed supply and demand is further complicated by the need to maintain pasture quality. A number of techniques are used to help achieve this. Two which are commonly used are the feed wedge and average farm cover (at a given date). Full coverage of these techniques is beyond the scope of this document but in summary they can be simplified to;

Feed Wedge: Two pasture target values are used: Pre-graze cover and post-graze residual. Paddocks are ranked accord to cover and a “target line” is super imposed over this. Paddocks with covers above the line show a feed surplus, and those below highlight periods of deficit. This information then drives management decisions such as increase/decrease in rotation length and/or pasture conservation.

Average Farm Cover: Is more complex than the simple feed wedge. Target farm covers are set for specific dates in the future. This is used in combination with target pre-grazing cover, post-grazing residual and area per animal on order to predict farm cover at the selected future date. The difference between target and predicated cover indicates feed surplus or deficits. This information then drives management decisions.

The default grazing management implemented in DDRules is similar to the feed wedge approach, but differs in a couple of important ways;

1. Only the post-grazing residual target is set by the user/default
2. A “fixed” rotation length drives pasture availability by default
3. Future prediction of pasture surplus is not used to trigger pasture conservation
4. Pasture deficits are met (by silage/supplements) as they occur

While this is the “default” management strategy implemented by DDRules, mechanisms to allow the user to implement these techniques (via management script or module) are available.

## Alternative Option [LUDF]

### Introduction

In order to use the Lincoln University dairy farm data an alternative grazing management system based on the “feed wedge” approach is used. This approach uses a grazing residual and rotation length to calculate average and pre-grazing farm covers required to meet current animal demand. Excess pasture (i.e. farm in pasture surplus) is harvested on a weekly basis. When in deficit animals are feed supplement (silage or purchased feed) to meet demand.

Refer DairyNZ farmFact 1-14 Pasture feed wedges for further information.

### Parameters

* AllocationType = 1 to enable this option
* GrazingResidual [kgDM/ha] - desired post grazing pasture residual
* GrazingInterval [days] - number of days required to complete a full grazing rotation

DMpre = (SR x Intake x GrazingInterval) + GrazingResidual

DMavg = (DMpre + GrazingResidual) /2

Where;

DMpre - Pre-grazing cover target [kgDM/ha] = IdealPreGrazingCover

DMavg - Average farm cover target [kgDM/ha] = IdealAvgGrazingCover

SR - Stocking Rate [cows/ha] = StockingRate

Intake - Cows intake to meet requirements [kgDM/day] = ME\_Demand\_Cow / Pasture ME



### Related Model Outputs

* IdealPreGrazingCover [kgDM/ha] - Pre-grazing cover target
* IdealAvgGrazingCover [kgDM/ha] - Average pasture cover target
* FeedSituation [kgDM/ha] - Amount of dry matter on farm in relation to the amount required to meet animal demands (+ve = surplus, -ve deficit)

### Allocation Process (weekly)

1. Add paddock/s not completely grazed from last week to head of grazing queue
2. Sort remaining paddocks by cover
3. Calculate area required to meet rotation length (A)
4. Add paddocks to grazing queue until area allocated >= A

# Pasture Conservation

### Silage cutting rule parameters

* Start date [ConservationStart]
* Finish date [ConservationFinish]
* Trigger mass [ConservationTrigger]
* Cutting residual [ConservationResidual]
* Store on farm for later use [SilageStoreEnable]

### Key points

* Silage can be stored on farm for later use [SilageStoreEnable].
* Fixed / user specified ME and nitrogen content [SilageME, SilageN].
* A proportion of the cut mass can be returned to the surface organic matter pool at the time of cutting [SilageCutWastage].
* Paddock masses checked daily during conservation window
* Closed paddocks harvested weekly
* Closed paddocks are removed from grazing rotation until post harvest
* Paddocks being grazed are not closed for conservation. This rule can cause unexpected results in simulations with either a small number of paddocks or very low stocking rates e.g. no silage cut and pasture cover getting high in paddock where growth rate is greater than removal rate.

### Potential improvements

* Model the silage/baleage making process
  + ME and N content could be modelled using pasture quality at time of cutting
  + How to handle feeding silage of differing quality?
  + Will that level of realism ever be required?

### Special Case: Surplus Removal from Feed Wedge

Pasture conservation is triggered when the farm is in surplus by a given percentage.

When this happens pasture is removed/conserved in order to “reshape” the feed wedge. This is done by:

1. Calculating each paddock’s position in the feed wedge/rotation
2. Calculating an ideal pasture cover for the paddock
3. Cutting paddock to ideal cover if current cover exceeds ideal

This approach is used to remove surplus pasture during times of high growth and/or feed demand. In reality a number of decisions could be made as to how to deal with the feed surplus e.g. altering the rotation length, removing strategic paddocks from the rotation for conservation or forage planting, bring on grazers etc. The approach used here does not reflect any of these approaches accurately but remove surplus without the decision requiring a temporal aspect to the harvesting condition/reaction.

#### Pros

* Simple implementation / trigger rule
* Maintains the shape of the feed wedge

#### Cons

* Not entirely realistic
* Can place the farm in to feed deficit (resulting in cut pasture being fed back to cows the next week)

# Supplementary Feeding

Due to the almost limitless number of possible feeding scenarios the simplistic approach outlined in the original documentation has been implemented as a starting point. In summary this is:

* Any feed deficit (at the end of the grazing day) is met by supplement
* If silage is available it is fed with priority
* Any remaining demand is met with bought in feed

Silage made/stored on farm

Purchased feed i.e. grain, PKE

Feeding to meet feed deficit

* Can feed very small amounts in order to totally met animal demand

SilageWastage

SilageStore

# Dairy Cow Model

## Introduction

DDRules currently implements a highly simplistic animal model that goes some way toward providing realistic animal energy demand, nutrient return and milk production. Data of a representative dairy cow has been provided by DairyNZ. This data has been derived from their whole farm systems model and is a representative dairy cow derived from the whole herd data. The data points provide are live weight, body condition score and milk solids production and are based on a June calving dairy herd.

### Key Concepts

* Base units: metabolisable energy (ME/kg, ME/ha).
* Base scale: per hectare.
* Point values supplied for live weight, body condition and milk production.
* Energy input via grazing and/or supplementary feed.
* Energy partitioned into maintenance, live weight change, pregnancy and walking.
* Nutrients partitioned and returned as dung and urine.
* Single animal multiplied to herd scale.

## **Energy Requirements**

The dataset above only provides information relating to cow condition and production. DDRules distils this data down to the energy required to meet animal demand (MEtotal).

MEtotal is defined as;

MEtotal = Σ(MEmaint, MElact, MEpreg, MELWt, MEwalk)

where

MEmaint = [0.6, 0.55] x LWt0.75 [lactating, dry]

MElact = 65 ME/kgMS

MELWt = 46, 39, 32 [dry, lactating, loosing] x ∆LWt [kg LWt]

MEpreg = 5, 10, 20, 30 ME/day

MEwalk = 1 ME/km

Assumptions

* Live weight and body condition score are decoupled due to live weight including foetus weight.
* Live weight drives maintenance requirements, but includes foetus weight.
* Body condition score change drives energy requirements for live weight change
* 1 body condition score = 35 kgLWt
* Animals assumed to not walk. The cost of walking is very low and the distance walked is driven by farm size, number of milkings per day and length of lactation.
* Calculations assume a diet ME of 11ME/kgDM consumed (as per DairyNZ publications)
* Only demand for the milking platform is considered e.g. no calves or heifers grazed on farm.

References

DairyNZ Farmfact 1-7 (2006) Feed requirement of milking cows

DairyNZ Farmfact 1-8 (2006) Feed requirement of dry cows

DairyNZ (2010) Facts and figures for New Zealand dairy farmers (under review)

## **Animal Intake / Grazing**

An enormous number of possible grazing feeding combinations could be implemented from within DDRules, with almost as many assumptions attached to them. For simplicity two methods of pasture removal have been implemented.

Note: how these interact with the supplementary feeding options will be discuss in a later section.

The two methods of controlling pasture removal are currently implemented in DDRules. For the sake of argument they will be referred to as “set stocking” and “break feeding”. The method the model uses during can be set in the models “ini”. Both methods remove pasture using the Apsim “remove\_crop\_biomass” event.

### Removal: Set stocking

* Animals enter the grazing paddock with a feed demand
* Whenever possible the model will attempt to meet this feed demand with pasture
* No limits are placed on availability of dry matter above the grazing residual
* Remaining feed demand will be met by supplements

### Removal: Break feeding

* Animals enter the grazing paddock with a fed demand and will attempt to meet this feed demand with pasture
* Potential animal restriction by the number of days the herd will spend in the paddock
* Remaining feed demand will be met by supplements

### Example

* paddock contains 1500kgDM available to the herd
* herd demand is 1000kgDM/day 3 days
* *No pasture growth predicted*

**Daily outcome under set stocking;**

1. 1500 kgDM available, 1000 kgDM removed, zero kgDM supplements fed
2. 500 kgDM available, 500 kgDM removed, 500kgDM supplements fed
3. Zero kgDM available, Zero kgDM removed, 1000kgDM supplements fed

**Daily outcome under break feeding;**

1. 500kgDM available, 500kgDM removed, 500kgDM supplements fed
2. 500kgDM available, 500kgDM removed, 500kgDM supplements fed
3. 500kgDM available, 500kgDM removed, 500kgDM supplements fed

Note: The difference between these two methods becomes apparent in simulations with a small number of paddocks or when a very long grazing rotation is set.

Assumptions

* Near perfect pasture management control i.e. animals will graze down to the given pasture residual and no further.
* Animals will be returned to paddocks in cases where the given grazing residual has not been met. Often this results in animals being returned to a paddock to remove a (potentially) unrealistically small amount of dry matter e.g. 50kgHa. As the model is currently being developed to replicate the management practices of the LUDF (where often animals are returned to paddocks for very short periods of time) this is of little consequence. Note: care should be taken if the model is to be used to examine more “typical” farm management practice.
* Very small amounts of supplementary feed will be fed to animal in cases where demand is almost met by pasture e.g. 0.2kgDM/head. This behaviour is to ensure proper balance between animal intake and production due the stochastic nature of the current animal model i.e. production and demand are not dynamic, therefore demand must be met to ensure correct energy balance. For longer term simulations the effect this has on model output is very small.

## **Nutrient Partitioning**

### Summary

The animal partitions consumed nitrogen to milk production and body weigh change, with the remainder being partitioned into dung and urine. The excreta nutrients are returned the grazed areas (or whole farm in special cases) using the standard fertiliser and surface organic matter events. Nitrogen balance is also calculated to ensure no nutrient inputs and outputs are in balance at all times.

### Nitrogen to excreta

Nexcreta = Neaten - Nmilk - Nbc

where;

Nmilk = Nitrogen partitioned to milk production (0.07 x kgMS [kgN/day])

Nbc = Nitrogen partitioned to change in body condition (0.02 x ∆LWt [kgLWt/day])

### Dung

Ndung = 0.4 x Nexcreta

DMdung = 1 - DMdig

where;

Ndung = Nitrogen returned as dung [kgN/day]

DMdung = Dry mater returned as dung [kgDM/day]

DMdig = Dry mater digestibility [%]

### Urine

Nurine = Nexcreta - Ndung

Where:

Nurine = Nitrogen returned as urine [kgN/day]

## **Nutrient Returns**

Excreta produced by the model is returned to the paddock/s using standard Apsim event calls. Because of this each paddock is required to contain fertiliser, irrigation and surface organic matter components.

### Dung

* Returned to the surface organic matter pool using the “BiomassRemoved” event
* Crop\_type : “RuminantDung\_PastureFed”
* Phosphorus is also returned [McDowell 2005]

### Urine

DDRules will return urine using the UrinePatch model if it is present in the paddock definition. If it is not then urine is returned using fertiliser and irrigation events.

Urine patching call

* Stock type : “DairyCow”
* Amount of nitrogen returned [kgHa]
* Stocking density [su/ha/24 hours]

Otherwise

* Fertiliser event
  + Amount [KgN/ha]
  + Application depth = 300mm [Source? Rogierio’s modelling?]
  + Fertiliser type = “urea\_N”
* Irrigation event
  + Amount = Nurine / 0.08 / 10000 [mm]

### Assumptions

* Urine nitrogen concentration = 8% [source?]
* No nutrient transfer between paddocks i.e. rumen passage rate assumed to be less than 24 hours.
* Nutrients are returned to the grazed paddocks in proportion to the ME removed from them
* All excreta returned to grazing paddock i.e. no loss to laneways or dairy shed.
* Special case: When no paddocks have been grazed nutrients are returned evenly across the entire farm.

### References

McDowell and Stewart (2005) Phosphorus in Fresh and Dry Dung of Grazing Dairy Cattle, Deer, and Sheep, J. Environ. Qual. 34:598-607 (2005). Table 1.

# Additional Dairy Farm Operation

## Introduction

DDRules has the ability to model two additional dairy farm operational issues;

1. Return of dairy shed effluent to paddocks
2. Deposition of nutrients to laneways

## Dairyshed Effluent Return (WIP)

### Settable values

HoursInDairyShed - hours per day in the shed [0-24]

EffluentPaddocks - List of paddock name to return effluent to

EffluentPaddocksPercentage - Proportion of farm effluent is returned to [0-1]

EffluentPondVolume - effluent storage capability [?]

### Operation

A very simplistic representation of the cow / dairy shed interaction can be modelled. Under the assumption that cow nutrient returns occur evenly throughout the day. The amount of nutrient deposited in the shed is correlated directly with the proportion of the day the cows spend in the shed (i.e. not on pasture or laneways. This nutrient is collected and returned evenly over a proportion of the farm. Storage is included to allow future development of nutrient return rules (e.g. if soil moisture is above a given level then store if possible).

### Return to Named Paddock [EffluentPaddocks]

Effluent paddocks can be set via a comma delimited list of paddock names

### Return to Percentage of Farm [EffluentPaddocksPercentage]

The paddocks to which nutrients are returned will remain same throughout the simulation. Effluent will be returned to the “first” paddocks in the simulation tree e.g. in a 20 paddock simulation and EffluentPaddocksPercentage = 0.1, then effluent is only return to the first two paddocks in the simulation.

### Important Implementation Notes

* Set paddocks by name is given priority over any percentage setting
* Effluent will be returned evenly to all paddocks in the simulation if no values are set
* Time spent in the dairy shed has no impact grazing time e.g. if set to 24 hours the cows will still graze the paddocks

## Laneways

### Settable values

ProportionOfFarmInLaneWays - area allocated to lanes [0-1]

HoursOnLaneWays - hours per day on laneways [0-24]

### Operation

Similar to the implementation of the dairy shed, effluent collection laneways are assumed to receive a proportion of the excreta. This is applied to a “special” paddock in the simulation which will be given the name “Laneways”.

This paddock is assumed to not be grazed and as such this area is not included in the farm’s effective area e.g. for a 100ha farm with ProportionOfFarmInLaneWays = 0.05 the “Laneway” will be 5ha in area and not included in the effective farm area i.e. actual farm area would be effective farm area + laneways + shed area etc.

### Notes

* Laneway paddocks will not be included in the grazing rotation
* Time spent on laneways will not affect animal ability to graze
* There is not sanity check between time in laneways and time in dairy shed

## Whole Farm Irrigation

DDRules has a simple irrigation “centre pivot” irrigation component. This can be used exactly as an irrigation event at paddock level would be used, as it takes the same data. The only difference is that it will irrigate the entire farm. N.B this component will be externalised at some time in the future.

### Settable values

irrigation\_efficiency [0-1] - proportion of applied water that reaches the soil surface

### Example Event

ApplyIrrigation amount = 6 (mm)

<EventHandler()> Public Sub OnApply(ByVal amount As IrrigationApplicationType)

This event call will apply 6mm of irrigation across the whole farm in a single day. If irrigation\_efficiency is set to 1 [default] all 6mm will reach the soil surface, if set to 0.5, only 3mm will reach the soil surface i.e.

Water reaching soil surface [mm] = amount applied [mm] x irrigation\_efficiency [0-1]

### Related Outputs

Irrigation [mm] - quantity of irrigation water applied to the whole farm on a given day

## Whole Farm Fertiliser

A fertiliser event is also made available by DDRules as per the irrigation event above.

### Settable values

ApplyFertToEffluentPdks [0, 1] - 1 = Fertiliser applied to whole farm, 1 = apply only to non-effluent paddocks.

### Example Event Call

#### Old Manager

ApplyFertiliser amount = 10 (kg/ha), depth = 0 (mm), type = urea\_N()

#### New Manager

<EventHandler()> Public Sub OnApply(ByVal amount As FertiliserApplicationType)

### Related Outputs

Fertiliser [kgN/ha] - amount of fertiliser applied to the whole farm i.e.

Fertiliser applied [kgN/ha] = Application amount [kgN/ha] x Area of application [ha] / Farm Area [ha]

# Appendix

## Core Model/Code Structure

### Core classes

DDRules

* Interface between Apsim and the DDRules model
* Responds to Init2 (model setup), OnPrepare (data clearing) and OnProcess (model time step)
* Set model parameters (grazing rules, feeding rules etc.)
* Supplies output data to Apsim

Farm

* Drives core model functionality
* Maintain paddock lists
* Manages animal pasture interactions

Paddock

* Interfaces with the underlying Apsim paddock
* Adds additional data and convenience functions

Biomass

* Stores all pasture related information (DM, ME, N and digestibility)

SimpleCow

* Model code for a single dairy cow

SimpleHerd

* Manages extrapolation from single cow to dairy herd

FeedStore

* Used to store feed produced on farm
* Also used to hold information about supplementary feed used

### Paddock

**Paddock class (Paddock.vb)**

DDRules uses a wrapper type class to add the functionality to the basic Apsim paddock instance. The wrapper adds convenience functions and related data in order to simplify interaction with the underlying Apsim paddock and related event calls. A local reference to the underlying Apsim paddock allows direct interaction with the simulation without reliance on higher level classes e.g. in an OnPrepare event the farm requests a paddock in the model to retrieve its pasture mass data from the simulation, without needing to know which Apsim paddock the instance represents. All parameters passed to methods in this class take values with units of either kilograms or kilograms per hectare. All conversion to and from the standard Apsim units is managed here.

Core functionality;

* Pasture cover data retrieval (mass, nitrogen content etc.)
* Pasture removal (grazing and cutting)
* Nutrient return (dung and urine)
* Track paddock status (grazing, not grazable, closed for conservation etc.)
* Paddock area/s
* Output data storage e.g. pasture removed and nutrient returns

### Farm

**Farm class (Farm.vb)**

This class contains the core functionality of a dairy farm and is where any management code is to be found. The class contains a list of all paddocks in the simulation as well as a reference to the dairy herd, feed stores etc.

### Biomass

**Biomass class (Biomas.vb)**

This class is used to present an amount of biomass in the simulation. This can be from a single plant species or a mixed ration. It is used from a representation of biomass in the system e.g. animal intake, pasture removal and supplement supply. It acts as the core interface between the biomass representation used by DDRules and the lower level biomass data types as used by the various Apsim modules. Convenience functions to convert between the various mass related data types can also be found here e.g. RemoveCropDmType as required for a "remove\_crop\_biomass" event.

### FeedStore

**FeedStore class (FeedStore.vb)**

This class is used to store/track both feed conserved on farm and supplements fed out on farm. Additionally it tracks daily flow in and out of the various pools for reporting purposes.

### SimpleCow / SimpleHerd

**SimpleCow & SimpleHerd classes (SimpleCow.vb & SimpleHerd.vb)**

These classes contain the model of a single dairy cow (SimpleCow) and manage the scaling of the single animal up to a whole dairy herd. Both these classes only model milking/dry cow. These is no representation of animals from birth through to first lactation i.e. no calves or heifers.

### Grazing Paddock Allocation Process

Notes:

Paddocks currently closed for conservation not added

## Common Terminology

Feed: dry matter available from pasture

Supplements: Alternative feed source used to supplement intake from pasture (most often silage, grain or PKE.

Concentrate: High energy density feed source used for supplementation e.g. barley, maize and soy.

Surplus: Periods where available pasture mass is greater than animal demand. This is often combated by speeding up the rotation or by closing paddocks for conservation.

Deficit: Periods where available pasture mass is less than animal demand. This is often combated by slowing the rotation and/or feeding supplements.

Pre-grazing cover: Pasture mass at which animals enter the grazing paddock. Too high has quality impacts, too low limits pasture growth potential.

Post-grazing residual: Pasture mass at which animals are taken out of a paddock.

Clicks: Measure of pasture dry matter. 1 “click” equals approximately 140kgDM

Grazing Rotation Length: The number of days it takes the herd to graze all paddocks on the farm/milking platform. Also referred to as a “Return Period” because this is the length of time between successive grazing of an individual paddock e.g. 30 day rotation length, animals will not return to graze a paddock until 30 days after initial grazing.